University of Camerino

School of Science and Technology

Master of Science in Computer Science

Penetration Testing Methodologies

Case Study: A small bank of the Marche

Candidate:

Elisa Martini

Advisor:

Prof. Fausto Marcantoni

Year 2016-2017

Ringraziamenti

Si conclude con queste ultime righe il mio percorso di studi e di conseguenza si conclude una fase della vita per me molto importante. Sono tante le persone che ho incontrato in questi anni a Camerino e tutte, nel loro piccolo, mi hanno aiutato a raggiungere questo traguardo così importante.

Ringrazio il mio relatore, il Prof. Fausto Marcantoni, che mi ha permesso di affrontare con lui questo periodo di tesi. La sua disponibilità, i suoi preziosi consigli e la sua sempre presente supervisione hanno contraddistinto la sua personalità, rendendolo un relatore eccezionale come pochi.

Vorrei ringraziare la mia famiglia, che mi è sempre stata accanto e mi ha permesso di affrontare al meglio questa bellissima esperienza. Mi hanno sempre sostenuto ed appoggiato in tutte le mie scelte e consigliato nelle decisioni più difficili da prendere. Mi rendo conto che questo non sia stato facile e di quanto la loro fiducia è stata per me necessaria. Senza di loro non sarei mai diventata la persona che sono.

Vorrei ringraziare il mio ragazzo Vincenzo, che in questi ultimi anni ha sempre creduto in me, anche quando forse non era la cosa più giusta da fare. Mi è stato accanto nei momenti belli e soprattutto in quelli brutti facendomi capire che nella vita bisogna sempre lottare e contare sulle proprie forze. Lo ringrazio per la pazienza con cui mi ha sopportato e per la grande forza che mi ha dato e ancora mi da nella vita di tutti i giorni. È sempre stato un punto di riferimento, ha sempre avuto il consiglio più opportuno ed è anche grazie a lui se sono arrivata fin qui.

Infine un grandissimo grazie va a tutti i miei amici. A tutti gli amici che ho incontrato e conosciuto a Camerino, che mi hanno accompagnato in questo periodo di studi e che hanno trascorso con me lunghe giornate in questo piccolo paese. A tutti gli amici che mi hanno sostenuto a distanza e che conosco da anni con i quali ho passato attimi indimenticabili.

Abstract

Il penetration testing è una delle tecniche più comuni per valutare l'affidabilità di sicurezza di un sistema. Il pentest tenta quindi, in via del tutto legale, di entrare all'interno del sistema, proprio come farebbe un hacker, cercando quindi di andare a scoprire e scovare quali sono le falle e le vulnerabilità che il sistema possiede.

Il penetration tester tenta di individuare potenziali vulnerabilità esistenti nel sistema per poi trovare soluzioni per affrontare tali carenze; aumentando la sicurezza dell'ambiente nel suo complesso.

I processi di penetration testing sono aiutati da una vasta gamma di tool automatici, ma vista la loro quantità, si potrebbero incontrare difficoltà nella scelta dello strumento più idoneo da utilizzare. Esistono diverse metodologie di approccio e di studio, ognuna usa tecniche differenti con strumenti differenti. Di conseguenza, questa tesi si propone in primo luogo di fornire e confrontare le più importanti metodologie open source di penetration testing, aiutando la comunità di sicurezza nella scelta del percorso migliore per il penetration testing.

Il termine open source fa riferimento a tutti quei software liberi, di cui l'autore rende il pubblico il codice sorgente, permettendo a tutti il libero studio e modifiche.

Le più importanti metodologie vengono rilevate dalle più privilegiate e comuni certificazioni per il pentester. È proprio lo scopo di questa tesi infatti analizzare metodologie come OSSTMM[1], OWASP[2], ISSAF[3], domandandosi così l'efficacia di una particolare metodologia rispetto ad un'altra.

Contents

List of Figures

List of Tables

1 Introduction

1.1 Background and Motivation

L'espansione e l'evoluzione delle tecnologie di computer, internet e web hanno reso la società più dipendenti da servizi di rete di computer che mai. Per questa regione, le società di oggi vanno sempre di più incontro a rischi e minacce all'interno della rete. Attacchi e violazioni alla sicurezza sono sempre più numerosi e possono provocare perdite negli affari e nella produttività, il tempo e il lavoro coinvolti in ridistribuzione sistemi infetti pone una spesa significativa.

Questi attacchi o violazioni direttamente o indirettamente, danneggiano la reputazione di un'organizzazione e provocano non conformità con le leggi di protezione della privacy del cliente. Le minacce alla sicurezza si sono evolute in maniera significativa in quanto coinvolge tutte le attività che l'organizzazione, le imprese, le istituzioni nel tentativo di proteggere il valore dei beni, l'integrità e la continuità delle operazioni. C'è stata una sfida nel fornire un ambiente sicuro; un'efficace strategia di sicurezza che aiuta a identificare le minacce e selezionare un set più efficace di strumenti, in modo tale che ogni organizzazione è in grado di ridurre il rischio di incidenti e perdita di dati risultante. Ad oggi, le notizie di violazione della sicurezza sono sempre più comuni e da un rapporto del Clusit[4] (associazione italiana sulla sicurezza informatica) sono molti gli attacchi effettuati.

Nel Clusit vediamo l'attacco ad Anthem (compagnia di assicurazione sanitaria) iniziato ad aprile 2014 ma scoperto solo a gennaio 2015 che ha provocando il furto di circa 80 milioni di record contenenti i dati personali dei clienti e degli impiegati.

L'attacco a Ashley Madison[5], servizio web di incontri per adulti espressamente dedicato alle avventure extraconiugali. È stata minacciata la sua casa madre (Avid Life Media) di pubblicarlo qualora il sito non fosse stato messo offline. L'azienda ha rifiutato, pertanto l'intero database è stato reso pubblico, causando un notevole scandalo ed importanti disagi per milioni di persone.

La più grande cyber-rapina del 2015, e probabilmente di tutti i tempi, è stata compiuta ai danni di oltre 100 istituti bancari appartenenti a più di 30 paesi del mondo, Italia inclusa, con un danno stimato di almeno 1 miliardo di dollari. Fin dalla fine del 2013 i cyber criminali responsabili dell'operazione hanno infiltrato con tecniche di phishing diverse organizzazioni finanziarie, infettandole con malware realizzato ad-hoc.

1.2 Objectives of the Thesis

Con tanti strumenti e metodologie a disposizione per il penetration testing professional la scelta del metodo più efficace e professionale diventa sempre più difficile. C'è poco studio e poca ricerca per informare al meglio i pentester su quali strumenti e tecniche possono essere più efficaci rispetto alle loro esigenze. Il primo obiettivo di questa tesi è quindi quello di offrire un'idea globale di quali sono gli strumenti più utilizzati e più efficaci in termini di prestazioni, documentazioni e costi. Vengono presi in considerazione quindi un numero limitato di strumenti, descritti ed analizzati al meglio. Dopo aver dato una visione globale di quali sono i tool più utilizzati ci soffermiamo ad analizzare e confrontare le metodologie per il penetraton testing, prendendo quindi anche in considerazione le più professionali certificazioni per questo ambito. L'ultimo obiettivo, ma non meno importante, della tesi è quello di illustrare ed evidenziare il mio caso studio di vulnerability assessment, non un penetration testing vero e proprio, effettuato per una banca delle Marche.

Nella tesi si farà sempre riferimento alla terminologia Pentration Testing, in quanto c'è una sottile differenza fra le due e nell'uso comune il penetration testing risulta essere più utilizzato e diffuso.

La differenza tra le due attività viene dettagliatamente spiegata in un capitolo della tesi, ma possiamo anticipare dicendo che il Penetration Testing è l'attività che si sviluppa quando, una volta riscontrata una vulnerabilità la si sfrutta per cercare di entrare nella mvacchina obiettivo.

Il Vulnerabilty Assessment, invece, è l'attività che ci porta alla scoperta di tutte le vulnerabilità esistenti nel sistema. Queste vulnerabilità non vengono sfruttate e di conseguenza il sistema non viene compromesso.

Una vulnerabilità è una falla, un difetto che il sistema in quel momento possiede e fa sì che risulti vulnerabile.

2 Concepts and definitions of Penetration Testing

Nonostante il fatto che gli attacchi informatici sono sempre più frequenti in questo secolo di informazioni, molte aziende e organizzazioni non testano ancora la loro infrastruttura per identificare il loro stato di sicurezza. Una volta collegati ad internet, i sistemi delle imprese possono essere scansionati ed attaccati e quindi ogni organizzazione ha bisogno di proteggersi da accessi non autorizzati. Secondo la maggior parte dei professionisti della sicurezza, le aziende dovrebbero difendersi dalle minacce con strategie di sicurezza, per esempio, con periodici controlli per valutare i rischi. Invece di aspettare che si verifichino attacchi, che è ovviamente pericoloso, non controllato, e inefficiente, gli imprenditori dovrebbero esaminare i loro sistemi regolarmente per rivelare qualsiasi difetto esistente nella rete o sito web che può essere sfruttato per compromettere tutto il sistema.

Simile al noto detto, “la miglior difesa è l'attacco”, possiamo dire che il modo migliore per determinare la protezione di un sistema è tentare di penetrarlo legalmente - conosciuto come penetration test.

2.1 What is Penetration Testing?

Il penetration testing è quindi un’attività che serve a scoprire ed individuare le problematiche di un determinato sistema. È quindi un’analisi effettuata dall’interno o dall’esterno del sistema che ci permette di rilevare le vulnerabilità che questa area analizzata possiede e sfruttare le sue debolezze. Vengono valutati così tutti i componenti dell’infrastruttura IT, le applicazioni, i dispositivi di rete e la sicurezza fisica. Si può anche pensare al penetration testing cercando di immedesimarsi nella mente di un hacker. Si utilizzano tecniche che questo personaggio malevolo potrebbe a sua volta sfruttare simulando i suoi comportamenti. Vengono infatti definiti hacker, nella terminologia più comune, tutti coloro che senza alcuna autorizzazione hanno l’accesso ad una

infrastruttura IT. È infatti questa la grande e maggiore differenza fra un pentester ed un hacker, entrambi sfruttano le stesse metodologie e tecniche per intervenire in un sistema, i pentester però sono persone autorizzate che non possono permettersi di distruggere l'infrastruttura del proprio cliente. Differenza anche essa molto importante tra pentester e hacker è che il primo deve trovare tutte le vulnerabilità presenti in un sistema, non solo quelle che gli permettono di avere accesso privilegiato al sistema.

Andiamo a parlare più dettagliatamente del penetration testing e come prima cosa vediamo che ci sono due tipologie alla base di approccio a questa attività : black-box testing e white-box testing. La differenza principale tra queste due tipologie di approccio è la quantità di informazioni che un penetration tester possiede sui sistemi da testare.

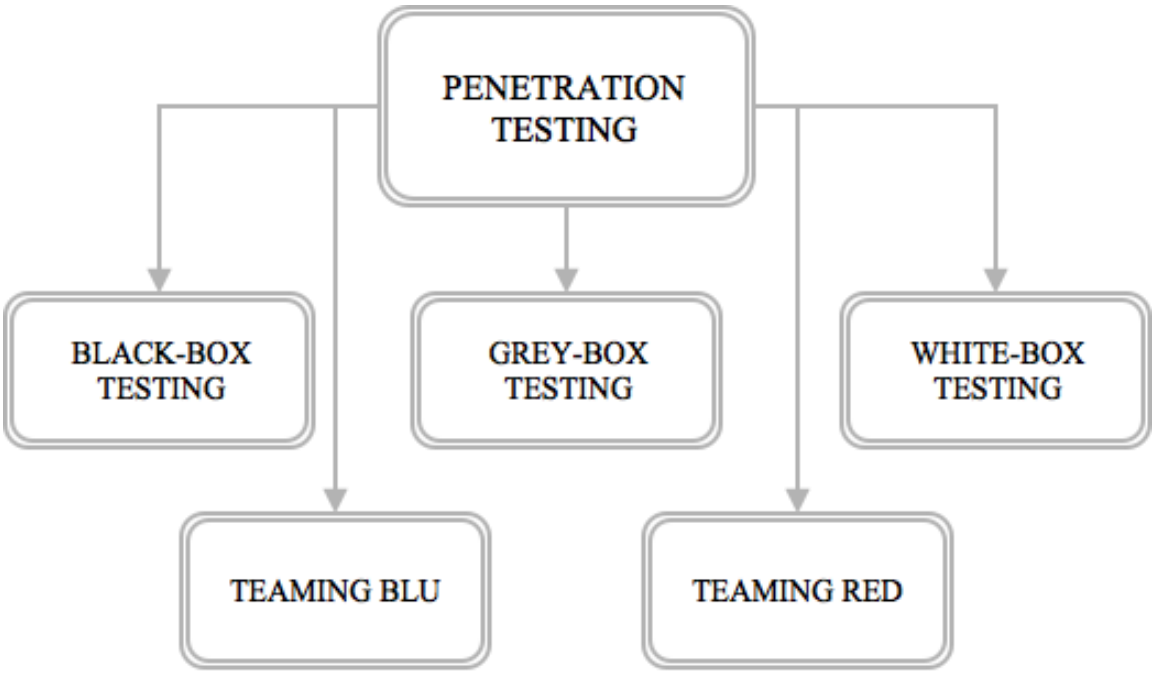

Figura 2.1: Initial penetration testing division

2.1.1 Black-box testing.

Tipologia di testing in cui viene simulato un attacco reale dal mondo esterno, infatti viene anche chiamato "test esterno". Un tester, non avendo informazioni dettagliate dell'infrastruttura, simula ciò che un attaccante potrebbe fare sfruttando le vulnerabilità rilevate. Si determina così se l'accesso al sistema è possibile, e in caso affermativo, quanti dati possono essere compromessi.

2.1.2 White-box testing.

Conosciuto anche come “test interno”, i tester simulano un attacco avendo la completa conoscenza dell’infrastruttura da testare. In questo modo i tester con i proprietari possono concentrarsi su un particolare obiettivo, verificando così l’integrità delle organizzazioni.

La combinazione di questi due tipi di penetration testing ha come risultato il **Grey-box testing**. In questo approccio i tester sono forniti di qualche conoscenza, risparmiando tempo per scoprire informazioni pubblicamente disponibili.

I penetration test possono essere condotti come "teaming blu" cioè con la conoscenza e il consenso del personale IT dell'organizzazione o "teaming rosso" cioè solo con la conoscenza e l'autorizzazione della gestione superiore. Il teaming rosso è più costoso e complesso da gestire, ma può fornire una migliore indicazione della sicurezza quotidiana, poiché gli amministratori di sistema non ne sono a conoscenza.

2.2 Goals of Penetration Testing

Uno dei principali obiettivi del penetration testing è quello di verificare il grado di sicurezza di un sistema, essendo al giorno d’oggi sempre più fortemente collegati all’innovazione tecnologica e di conseguenza esposti a possibili intrusioni ed attacchi.

Si effettuano i penetration testing per identificare quelle lacune presenti nel sistema o punti di accesso al sistema che fanno si che un utente malintenzionato possa sfruttarle ed infiltrarsi. Bisogna però ricordare che lo scopo del penetration testing non è quello di aggredire o rompere il sistema IT di un’intera società, ma fornire indicazioni e consigli per le problematiche trovate.

Considerando quindi il penetration testing come un’operazione ciclica, che deve ripetersi nell’arco di un determinato tempo, perché i sistemi sono sempre in continuo aggiornamento ed evoluzione, il penetration testing in primo luogo ci fornisce un primo passo per comprendere l’attuale posizione di sicurezza di un’organizzazione individuandone i difetti.

Comprendere la sicurezza di un sistema significa anche assegnare una priorità ai rischi riscontrati ed effettuare una valutazione del loro impatto. Individuati i rischi, il

penetration test fornisce un quadro di sicurezza che permette di oltrepassare le perdite finanziare e portare quindi l'infrastruttura al minimo livello di rischio.

2.3 Types of Penetration Testing

Oltre alla classificazione tra white-box e black-box i penetration test di differenziano anche a seconda della tipologia di penetration testing. Esistono infatti i network penetration testing, web application penetration testing e system penetration testing. La differenza sostanziale tra le varie tipologie di penetration testing sono gli oggetti che si vogliono esaminare.

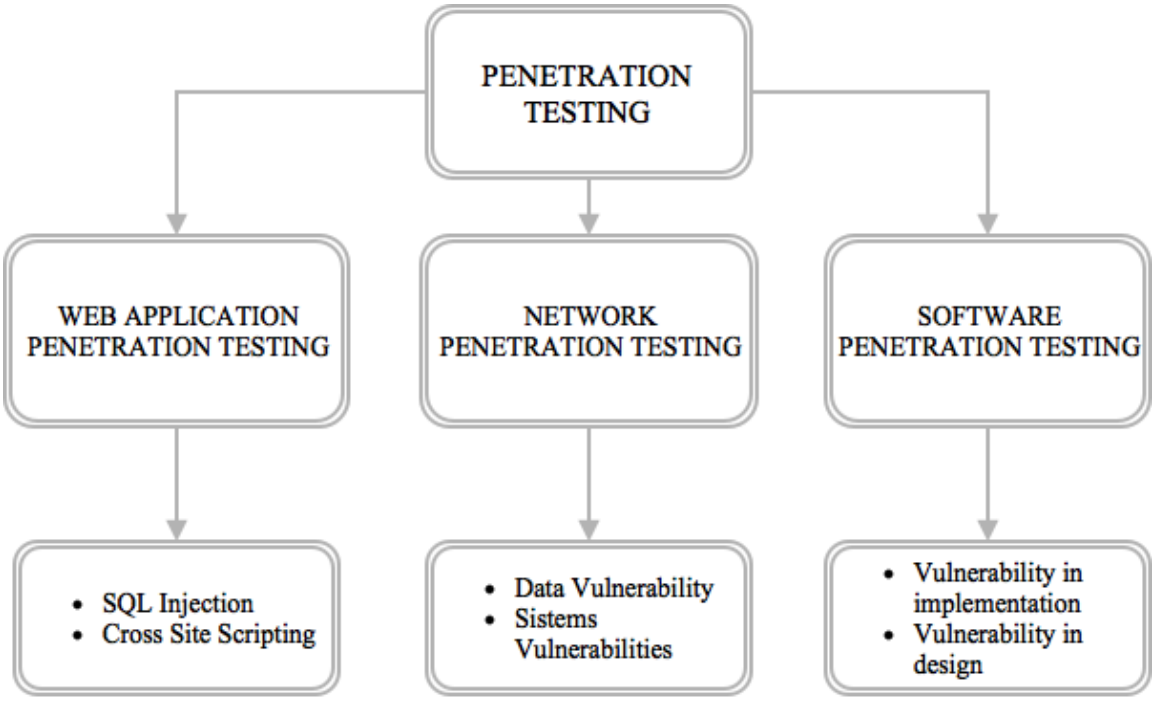

Figura 2.2: Types of Penetration Testing

2.3.1 Web application penetration testing

Spesso le applicazioni web compongono la maggior parte della superficie di attacco esposta verso internet. Le applicazioni web girano sui server web, quindi testare se un server web è sicuro da un attacco interno o esterno è fondamentale. Trovandosi il server web sulla porta 80/TCP, nota per i numerosi exploit che esistono, è importante prestare

molta attenzione e apportare al server numerose protezioni. L'exploit è una tipologia di script che permette l'esecuzione di codice malevolo sfruttando una specifica vulnerabilità.

In questo ambito, dobbiamo prestare molta cura ai meccanismi di autenticazione scadenti, ai difetti logici, alla divulgazione involontaria dei contenuti e soprattutto a tenere sempre aggiornato il nostro server web, perché la maggior parte delle vulnerabilità più importanti vengono date da sistemi operativi non aggiornati o dall'installazione di applicazioni.

Sono SQL injection e Cross-site-Scripting (XSS) le vulnerabilità più importanti per le applicazioni web.

La SQL injection è quindi una tecnica che mira a mandare del codice malevolo, sfruttando il mancato controllo sui dati ricevuti in input, ad una web app che fa uso di un database di tipo sql. Questo tipo di vulnerabilità ci permette ad esempio di autenticarci ai massimi livelli in aree riservate del sito senza essere in possesso delle credenziali di accesso e di visualizzare o alterare dati sensibili.

Per sfruttare la SQL injection bisogna prima identificare il punto di iniezione (injection point) e poi si può costruire il payload che permette di turbare una query costruita dinamicamente. Identificare l'injection point significa testare ogni input utente (parametri GET, POST, header HTTP…) che viene usato dall'applicazione web.

Trovare un injection point significa quindi provare ad iniettare:

- Terminatori di stringa: ‘ e “
- Comandi SQL: SELECT, UNION e altri
- Commenti SQL: # o - -

e verificare il comportamento dell'applicazione. L'applicazione risulterà quindi essere vulnerabile alle SQL injection se durante questo test la pagina, con l'inserimento dei comandi sopra elencati, si comporta in modo differente.

Una volta identificato l'injection point si passa all'utilizzo del payload. Un esempio potrebbe essere il Boolean based SQLi, che mira a trasformare la query originale in una condizione di vero/falso il cui risultato si rifletta in qualche modo sull'output dell'applicazione. Dopo aver trovato il modo migliore per sfruttare una SQLi

manualmente, è possibile automatizzare efficientemente l'exploitation usando dei tool automatici.

Cross Site Scripting è una famiglia di vulnerabilità che permettono a un attaccante di prendere il controllo del contenuto di un'applicazione web.

Il Cross-Site Scripting avviene quando dei dati provenienti da fonti non fidate sono inclusi in una pagina web senza essere prima sanitizzati e/o codificati adeguatamente. Per esempio, se un utente fornisce il proprio username per effettuare il login e l'applicazione mostra quello stesso username senza sanitizzarlo e/o codificarlo, cosa potrebbe succedere se lo username contenesse caratteri HTML? Il browser web non sarebbe in grado di distinguere tra lo username dell'utente e il codice HTML valido della pagina. Dei dati (lo username) sono mischiati con del codice (l'HTML della pagina)! Questo potrebbe permettere a un utente di autenticarsi con uno username contenente codice JavaScript malevolo e farlo eseguire nel browser all'interno del contesto dell' applicazione web. È opportuno quindi sanitizzare lo username prima di utilizzarlo.

Per esempio, se gli utenti potessero avere solo caratteri alfanumerici nei loro username, allora si potrebbe forzare questo limite con la sanitizzazione dell'input. Si può utilizzare una whitelist! Confrontare gli username contro una lista di termini fidati invece che con una di termini da evitare.

Per trovare un XSS si deve controllare ogni input utente e verificare se viene mostrato in qualche modo sull'output (punto di riflessione) dell'applicazione web. Dopo aver trovato il punto di riflessione (reflection point), bisogna capire se è possibile iniettare del codice HTML e vedere se arriva in qualche modo all'output. Questo permette di prendere il controllo della pagina di output.

Anche i cookie sono un'altra fonte di vulnerabilità delle applicazioni web e vulnerabilità come l'XSS consento agli hacker di rubare i cookie che possono essere usati per presentarsi al sistema come un utente lecito e quindi già registrato.

La maggior parte delle applicazioni web vengono analizzate utilizzando l'analisi black-box. Il pentester quindi esamina l'applicazione dalla prospettiva di un hacker provando a comprometterla con l'utilizzo di un gran numero di tools esistenti per il web application penetration testing.

2.3.2 Network penetration testing

Un network penetration test ha come scopo quello di identificare le vulnerabilità che si trovano nelle aree di maggior interesse per il business di un azienda.

Questa tipologia di test, infatti, ci permette di esaminare l'intera rete e alcune infrastrutture di rete critiche come firewall, server di database o server web.

L'attività di Network Penetration Test, aiuta quindi ad ottenere una definita sicurezza all'interno di un'azienda utilizzando determinati strumenti che permettono al meglio di identificare quali sono le problematiche e le falle individuate all'interno della rete utilizzando strumenti e processi che permetto di individuare tutte le vulnerabilità che potrebbero essere sfruttate da un eventuale attaccante.

In questo ambito quando si parla di vulnerabilità e di punti deboli individuati ci si può riferire sia a dati, sistemi compromessi o alterati da Exploit, Virus, Trojan, attacchi Denial of Service ecc… Si può far riferimento, però, anche ad altre tipologie di vulnerabilità che possono essere introdotte da patch e aggiornamenti o da errori sui Server, Router e Firewall.

Possiamo dire che il network penetration testing segue questi principali passi:

1. Ricerca del numero più vasto possibile di vulnerabilità nei sistemi maggiormente esposti.
2. Sfruttare le vulnerabilità rilevate per cercare di violare il sistema interessato.
3. Soffermarsi dettagliatamente sui sistemi interni alla ricerca di altre vulnerabilità che permettono di ottenere ulteriori accessi
4. Ripetere questo processo finché possibile

Troviamo una spiegazione dettagliata delle fasi nel network penetration testing nel capitolo 3.3.

2.3.3 Software penetration testing

I penetration testing software sono considerati abbastanza difficili da gestire, in quanto i software possono avere delle problematiche sia a livello di implementazione, sia a livello di progettazione.

Possiamo infatti individuare in questo ambito due tipologie di test:

- i test funzionali, che verificano che una determinata funzionalità esegua correttamente una specifica attività
- i test di sicurezza, che devono sondare direttamente e profondamente nei rischi di sicurezza per determinare come si comporta il sistema sotto attacco.

I test di sicurezza pongono una sfida molto più elevata rispetto ai test funzionali. Questo è dovuto perchè, essendo i test di sicurezza effettuati per vedere se il sistema commette errori, essi vengono pianificati per portare il sistema a produrre l'errore. Vengono alla fine sagnalate se e in quali circostanze si verifica l'errore. Se questi test non rilevano errori, abbiamo dimostrato che non esistono errori in determinate condizioni di prova, ma non che non esistono in nessun modo difetti.

Il penetration testing è un'attività molto comune e di norma viene utilizzata come test finale nel ciclo di vita (lifecycle) di un'applicazione. Questo però potrebbe essere un difetto per il software, in quanto, come abbiamo detto la maggior parte dei difetti vengono riscontrati a livello di progettazione e di implementazione.

Un modello tipico per il software penetration testing consiste normalmente in quattro passaggi:

- costruire un modello di minaccia a seconda delle classi di minaccia (spoofing di identità, manomissione dei dati, elevazione del privilegio…)
- creare un buon ambiente di test
- eseguire i casi di test
- reporting problem.

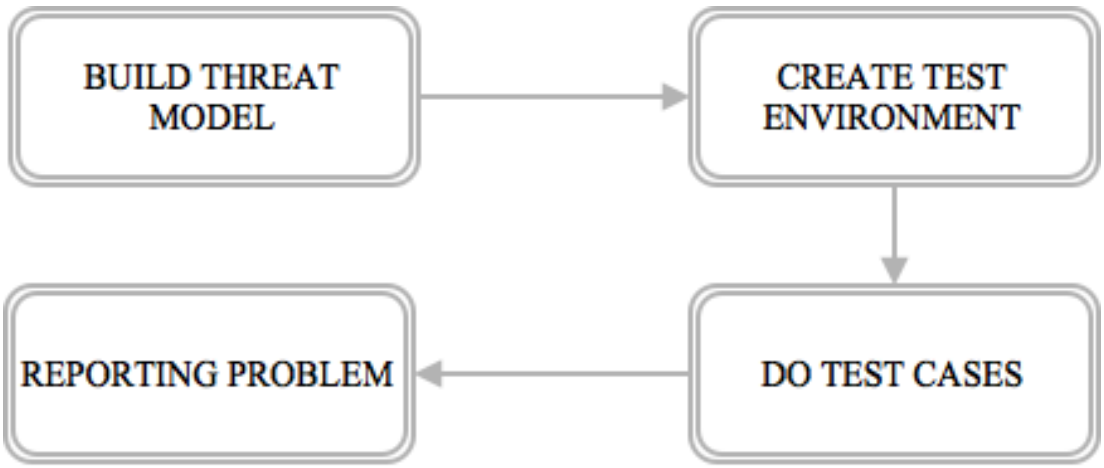

Figura 2.3: Software Penetration Testing Model

Questo approccio viene migliorato e il penetration testing software si basa sulle attività di test già effettuate dall'inizio del ciclo di vita del software. In tale approccio, i tools dovrebbero essere parte del test di penetrazione, in cui i tools di analisi statica possono utilizzare il codice software per identificare gli errori di implementazione comuni, mentre i tools di analisi dinamica possono osservare un sistema per individuare i guasti. I tools quindi riportano i guasti al tester per ulteriori analisi.

Molti fornitori di software hanno ora spinto controlli di garanzia della qualità prima nel ciclo di vita del software, piuttosto che alla fine. Le grandi aziende come Microsoft [21] o IBM [22] hanno iniziato a spingere la sicurezza in tutte le fasi del ciclo di vita dello sviluppo.

3 Penetration Testing Models and Methodologies

3.1 Pentration Testing Models

Molti modelli differenti sono stati adottati per il penetration testing. I più conosciuti e i più usati sono conosciuti come flaw hypothesis and attack tree.

3.1.1 Flaw hypothesis

La metodologia del flaw hypothesis la vediamo composta da 5 fasi:

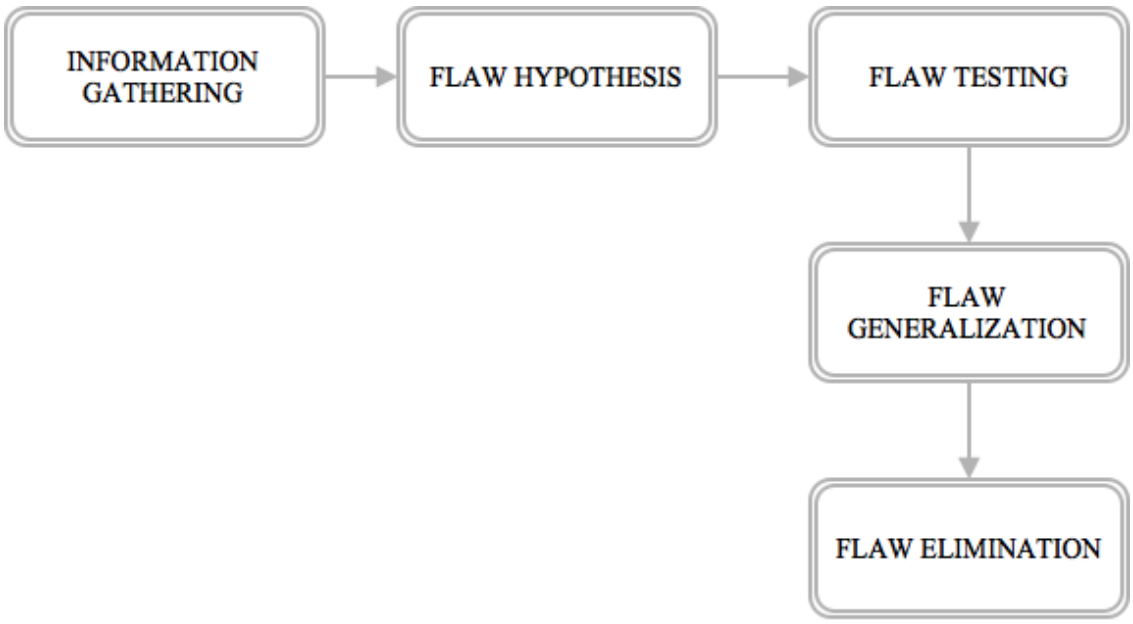

Figura 3.1: Phases of Flaw Hypothesis

1. Information Gathering:

In questa fase il penetration tester cerca di raccogliere più informazioni possibili riguardo al sistema che ha di fronte. In primis quindi prova a capire il suo disegno, capire il suo funzionamento e il suo utilizzo.

I tester in questa fase quindi elaborano un modello del sistema ed analizzano ogni aspetto del disegno del sistema per individuare le ipotesi errate e i potenziali difetti.

2. Flaw Hypothesis

Sulla base delle conoscenze acquisite nel primo passo e sulle vulnerabilità conosciute in altri sistemi i tester ipotizzano difetti del sistema.

Per individuare le vulnerabilità presenti nel sistema i pentester esaminano anche le policies e le procedure utilizzate per il mantenimento del sistema, perché anche se il sistema non rileverà debolezze, un funzionamento sbagliato o un installazione mal effettuata ha avuto come risultato dei sistemi lesionati.

Anche i problemi di implementazione portano a difetti di sicurezza. I penetration tester cercano sempre di oltrepassare ciò che i manuali di vulnerabilità indicano sull'individuazioni di esse, tentano sempre di omettere o oltrepassare passaggi e questa strategia fa si, nella maggior parte dei casi, di rivelare difetti di sicurezza.

Possiamo infine far notare che questa fase risulta essere abbastanza critica per un penetration tester, ma essi utilizzano la loro esperienza e i penetration testing effettuati in precedenza così da ricercare nuove vulnerabilità, ma anche provare ad individuare quelle riscontrate in precedenza.

3. Flaw Testing:

In questa fase i penetration tester testano i loro difetti ipotizzati. Qui le vulnerabilità vengono anche classificate a seconda della loro importanza, individuata a seconda degli obiettivi che sono stati prefissi prima dell'inizio del test. Se, ad esempio, è stato chiesto al penetration tester di individuare i principali difetti di progettazione o implementazione, questa tipologia di difetti avrà una priorità nella classificazione. Se invece lo scopo del test è quello di individuare le vulnerabilità del sistema sfruttabili da possibili attacchi esterni, i difetti relativi ai protocolli e ai programmi di accesso avranno una priorità molto alta, mentre quelli che interessano solo l'uso interno saranno classificati più in basso.

4. Flaw Generalization:

Una volta che un difetto è stato sfruttato con successo i tester tentano di generalizzarlo per trovare i tipi di difetti che esistono nel sistema e tentano anche di individuare altri casi simili ad esso.

5. Flaw Elimination:

In questa fase finale i tester suggeriscono metodi per eliminare il difetto o procedure di controllo per ottimizzarlo. La fase vera e propria dell'eliminazione del difetto a volte viene omessa perché non è compito del penetration tester andare a correggere le problematiche. I difetti però devono essere corretti e una corretta correzione richiede una comprensione del contesto del difetto, dei dettagli del difetto e del suo sfruttamento. È compito del penetration tester quindi cercare di far capire e di dare una spiegazione più dettagliata possibile di quali vulnerabilità sono state scoperte, il loro ambito e come forse potrebbero essere sfruttate da un utente malintenzionato.

3.1.2 Attack tree

Questo modello di penetration testing viene utilizzato nella maggior parte dei casi quando abbiamo meno informazioni di base del sistema che stiamo esaminando.

Assume infatti le sembianze di un vero e proprio attacco, simulando i comportamenti di un utente malevolo che quindi si trova davanti ad un sistema che non conosce bene, o meglio conosce poco.

Considerando la classica struttura ad albero composta da radice, nodi padri e nodi figli (foglie), il metodo di attack tree prende come obiettivo il nodo radice attraversando però in diversi modi i nodi figli che ci permettono di raggiungere l'obiettivo.

Dal basso verso l'alto, quindi, i nodi figli sono condizioni che devono essere soddisfatte per far si che il nodo padre diretto diventi vero; quando la radice è soddisfatta, l'attacco è completo.

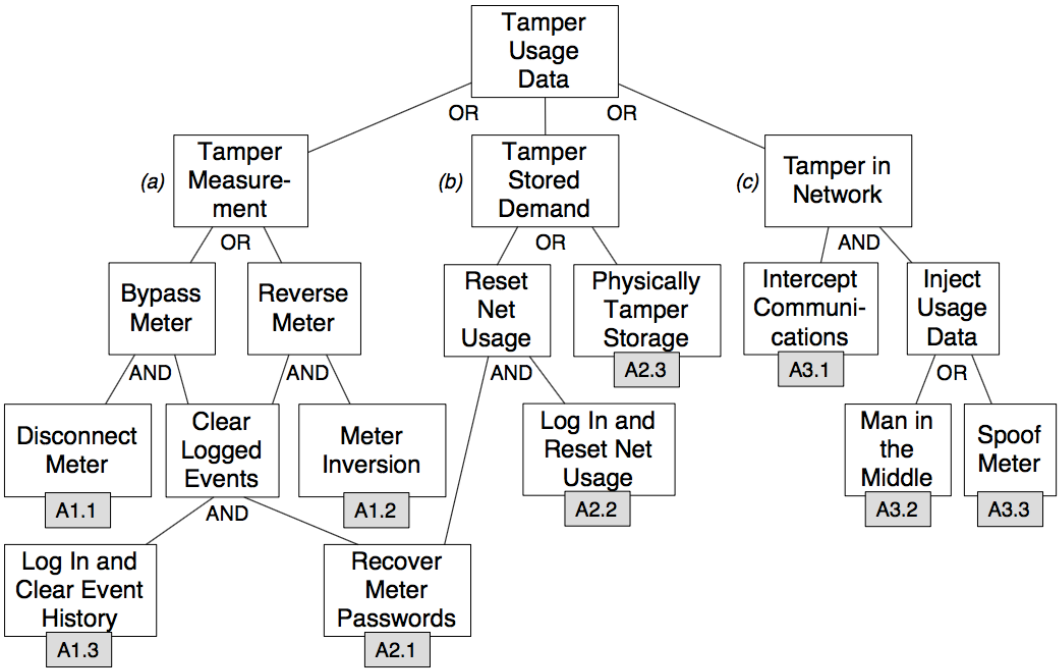

Figure 3.2: Example energy fraud attack tree.[6]

Prendiamo in considerazione la Figura 3.2, esempio in cui si usa un attack tree per commettere una frode in ambito energetico. La radice quindi specifica l'obiettivo finale, i nodi interni (quelli con genitori e figli) descrivono le diverse combinazioni che devono essere soddisfatte per commettere frodi. Le foglie dell'albero sono gli attacchi necessari

per le frodi di energia. L'attributo finale dell'albero sono le congiunzioni (AND / OR) tra ogni livello di nodi figlio che specificano se tutti o solo uno dei rami figlio devono essere seguiti per raggiungere l'obiettivo nel nodo padre.

Possiamo notare che gli attacchi alle foglie dell'albero sono abbastanza generali e sembrano applicabili alla maggior parte dei sistemi intelligenti di misurazione, ciò significa che questo tipo di albero è uno strumento abbastanza applicabile. Non avendo quindi dettagli su alcun sistema specifico, la sua utilità è limitata nel trovare vulnerabilità concrete, così il passo successivo è effettuare un secondo albero, più concreto e specifico per il sistema che si ha di fronte. La struttura di questo secondo albero è quella di prendere ogni foglia del albero "generale" che rappresenta un attacco e costruirla con più dettagli a seconda del sistema specifico (Figura 3.3). I sottogruppi nell'albero concreto definiscono quindi le condizioni esatte in cui è possibile raggiungere l'obiettivo finale.

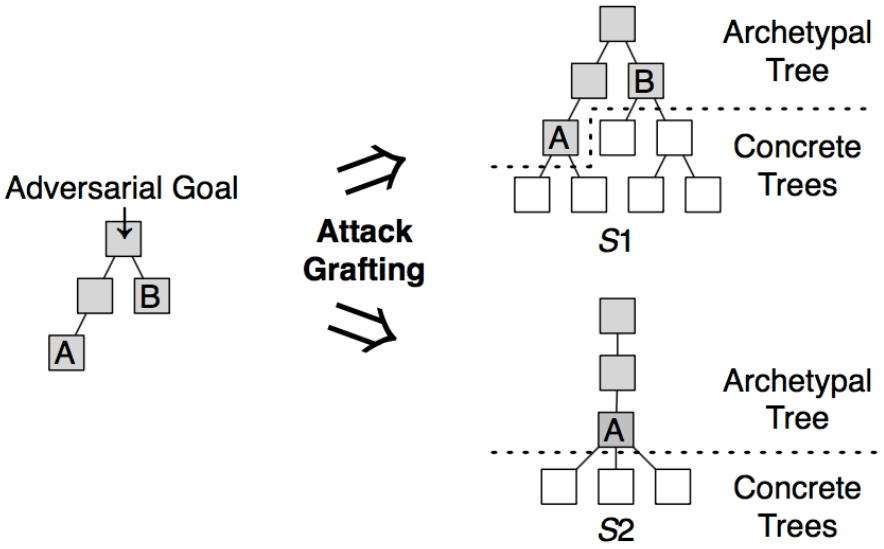

Figure 3.3: Concrete trees for two different systems (S1 and S2) onto an archetypal attack tree for a specific adversarial goal.[6]

3.2 Penetration Testing Phases

Il processo di penetration testing può essere suddiviso in più fasi o step, che in linea di massima si identificano in tutte le tipologie di penetration testing. Alcune metodologie hanno utilizzato nomenclature differenti per le varie fasi, ma tutte portano allo stesso obiettivo. La panoramica completa della metodologia di penetration testing quindi ci porta a dividere questa attività in 3 fasi principali chiamate: Pre-Attack Phase, Attack Phase e Post-Attack Phase.

Come possiamo vedere dalla figura 3.4 queste fasi seguono in sequenza e le attività di ciascuna fase dipende nello specifico dalle regole decise nell'ingaggio dell'attività di penetration testing.

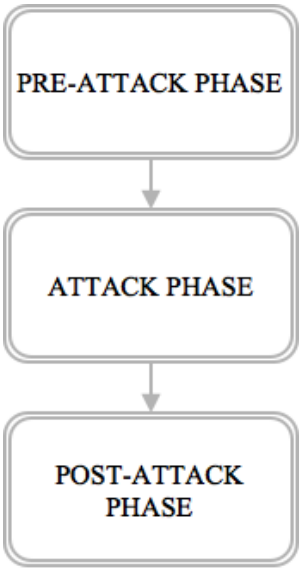

Figure 3.4: The Three phases in a Penetration Tests

3.2.1 Pre-attack Phase

La fase di pre-attacco è l'inizio di un penetration testing. In questa fase infatti si tenta di scoprire ed individuare più informazioni possibili riguardo il sistema che si sta cercando di analizzare.

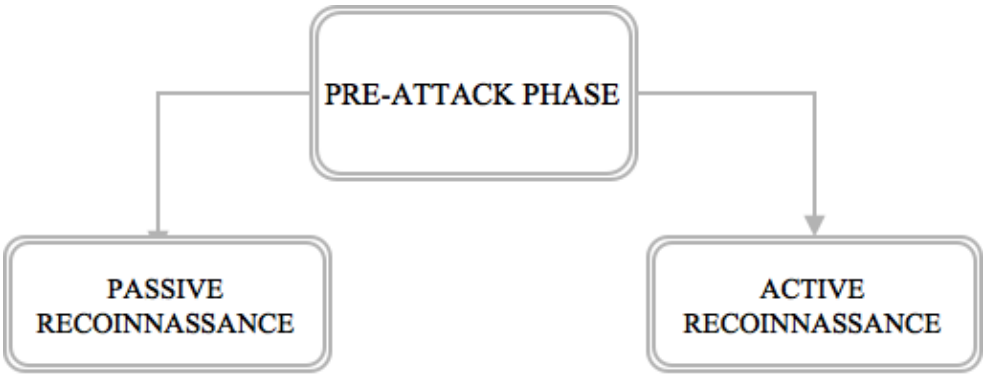

Figure 3.5: The Pre-Attack Phase in a Penetration Test

In particolare coinvolge la reconnaissance, che come possiamo vedere dalla figura 3.5 si può dividere in passiva ed attiva, ma entrambe sfruttano la potenza di internet. La reconnaissance passiva utilizza le informazioni disponibili sul web e coinvolge attività come l'ottenimento di informazioni di registrazione, prodotti, servizi ecc.. Nella reconnaissance attiva invece, si tenta di mappare il profilo internet dell'obiettivo. Infatti in questa attività, a differenza dell'altra, l'obiettivo è in grado di rintracciare i passi del penetration tester perché si parla di attività come il fingerprinting, port scanning, network mapping, perimeter mapping and web profiling.

3.2.2 Attack Phase

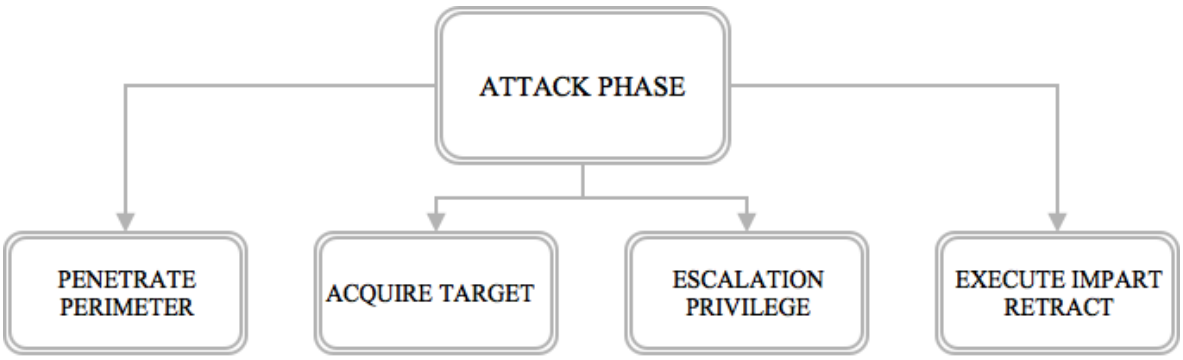

Figure 3.6: The Attack Phase in a Penetration Test

Questa è la fase principale del penetration testing dove si può ottenere il completo controllo del sistema. In questa fase si utilizzano tutte le tecniche e le possibilità per riuscire a sfruttare le vulnerabilità che si sono individuate nella fase precedente, perché è proprio sfruttando esse che si riesce ad ottenere il controllo della macchina.

Una volta che l'obiettivo è stato acquisito si tenta di escalare i privilegi, sfruttando l'obiettivo e installando una o più applicazioni che ci permettono di mantenere l'accesso nella macchina target, si tenta quindi il più possibile di sfruttare il sistema che è stato compromesso e quindi cercare anche di estendere il proprio controllo ad altri sistemi della rete.

Per privilege escalation si parla di una fase successiva allo sfruttamento della vulnerabilità. Significa quindi acquisire il controllo delle risorse di una macchina normalmente precluse a un utente. Un utente con maggiori autorizzazioni di quelle di un semplice utente viene chiamato utente root e riesce ad avere accessi privilegiati rispetto agli altri.

In questo caso vengono utilizzate tecniche come il brute force per ottenere lo stato di autenticazione, l'uso di buffer overflow, exploit, Protocol Analyzers, sono mezzi che ci permettono l'escalation dei privilegi nel sistema obiettivo e si farà di tutto per sfruttarli al massimo e sfruttare al massimo le loro potenzialità.

Le attività che normalmente si effettuano sempre in questa fase sono:

- ✓ Controlli per vedere come l'obiettivo risponde a risposte di errore
- ✓ Rispondere alle risposte con pacchetti creati per testare le liste di controllo degli accessi
- ✓ Test per misurare la soglia degli attacchi di denial of service inviando diverse variazioni di connessione sia TCP che UDP
- ✓ Test per vedere quali filtri di protocollo sono in atto cercando di connettersi con i protocolli più utilizzati (ad esempio SSH, FTP e Telnet).
- ✓ Test per verificare se l'Intrusion Detection System consente di contenere contenuti dannosi e scansionando l'obiettivo in molti modi vedere se l'Intrusion Detection System cattura traffico anormale.
- ✓ Verificare se i sistemi come il web server rispondono alla scansione del server web eseguendo diversi metodi come POST, CANC e COPY.

3.2.3 Post-Attack Phase

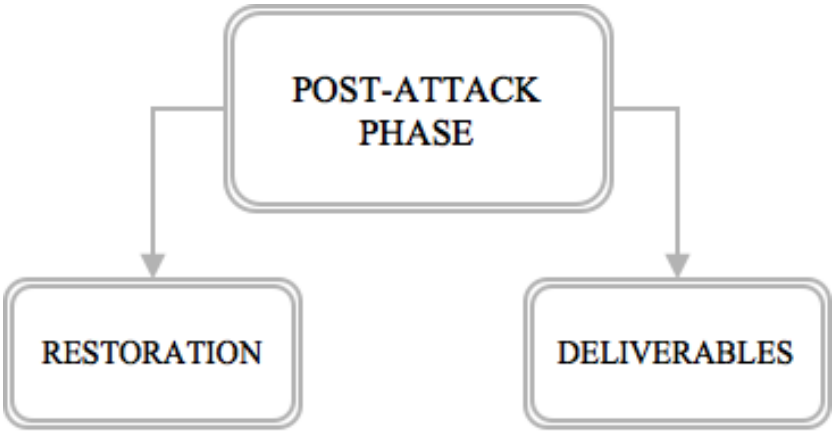

Figure 3.7: The Post-Attack Phase in a Penetration Test

La fase di post-attacco, come mostrato nella Figura 3.7, prevede prima di tutto il ripristino dello stato iniziale del sistema che si è andati ad analizzare.

Vengono quindi ripristinati i sistemi al loro stato di pre-test originale, vengono quindi rimossi dei file di root installati o dei programmi backdoor, ripristinati i dispositivi di rete e l'infrastruttura di rete, effettuata la pulizia delle voci di registro aggiunte durante lo sfruttamento e rimosse infine le condivisioni e le connessioni stabilite durante la fase di accesso.

Viene poi effettuato un report in cui vengono mostrati tutti i risultati del penetration test. Il report deve essere un documento molto dettagliato in cui si deve chiaramente spiegare e analizzare ogni singolo passo che è stato effettuato durante la fase di test. Si aggiungono inoltre i suggerimenti ritenuti più opportuni per la correzione degli errori riscontrati.

3.3 Network Penetration Testing Phase

Dopo aver descritto le tre fasi generali che si utilizzano per tutti i tipi di penetration testing, andiamo a vedere nello specifico e in maniera dettagliata le fasi che maggiormente vengono utilizzate per il network penetration testing.

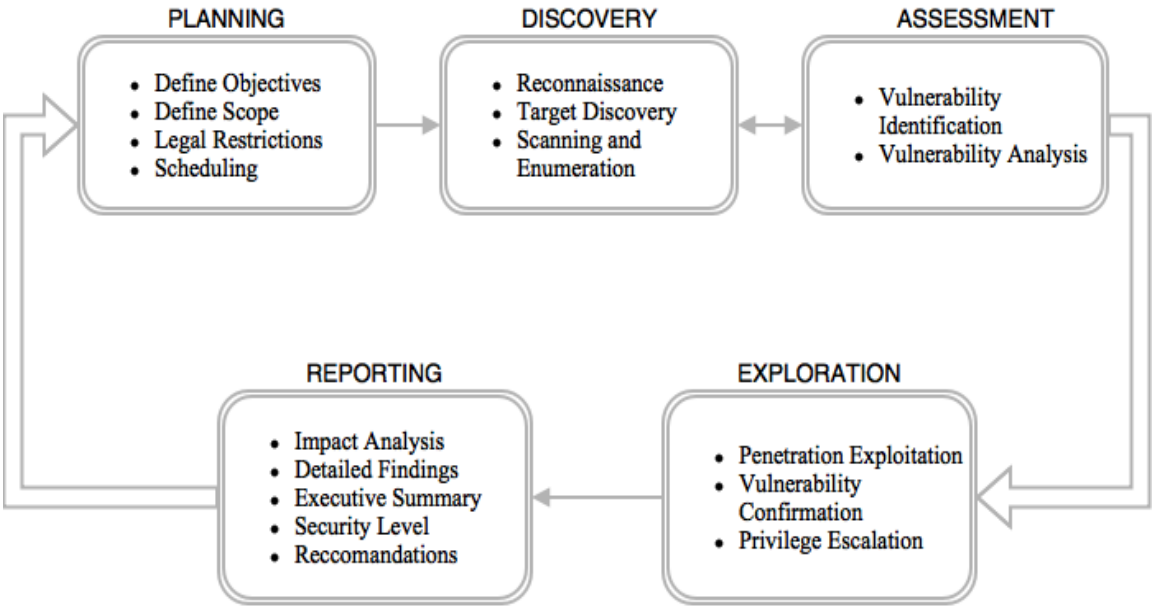

Figure 3.8: Network Penetration Testing Methodology

Come vediamo dalla figura 3.8 l'attività è composta da 5 fasi ed ognuna di essa è altrettanto suddivisa in operazioni più dettagliate.

3.3.1 Planning Phase

Questa è la fase di inizio, la fase di ingaggio dell'attività di penetration testing e anche se sembra una fase poco importante rispetto all'attività principale del test, non è così perché è proprio grazie ad una buona pianificazione e preparazione che si riesce ad ottenere un ottimo risultato.

In questa fase vengono definiti e formulati gli obiettivi, l'ambito, le restrizioni legali e la pianificazione dell'attività. Il penetration tester redige anche un "patto di responsabilità" con l'azienda, in cui scrive di non assumersi le responabilità in merito a danni o compromissioni che si potrebbero verificare durante l'attività. Essendo un'attività quella del penetration testing molto invasiva, anche un pentester con molta esperienza non può mai anticipare cosa potrebbe accadere e in quali situazioni si potrebbe trovare, essendo ogni volta una situazione e un ambiente differente.

In questa fase viene pianificato ciò che verrà attaccato, quando e come, viene discusso di quali sono le particolari restrizioni legali al fine di non andare a compromettere il sistema in maniera sbagliata. Anche le attività amministrative come ad esempio la raccolta di documenti e le attrezzature riservate rientrano in questa fase.

Si può definitivamente iniziare il penetration testing però quando l'azienda rilascia un documento in cui c'è l'effettiva autorizzazione per l'inizio di questa attività. Questo permesso, spesso called the rules of engagement (ROE), should include:

- Gli indirizzi ip o gli intervalli da testare
- Eventuali restrizioni su particolari host, cioè host, sistemi, sottoreti che non bisogna analizzare
- Una lista di tecniche, come ad esempio DoS (Denial of Service), e tecniche come cracker di password, sniffer di rete ecc… che sono accettate dall'azienda

3.3.2 Discovery Phase

Dopo aver definito gli obiettivi e la pianificazione della nostra attività, passiamo alla seconda fase del nostro test, ovvero la fase di discovery.

Questa parte può essere ulteriormente divisa in:

1. Reconnaissance and Target discovery
2. Scanning and Enumeration

3.3.2.1 Reconnaissance and Target discovery

Lo scopo principale di questa fase è quello di riuscire a catturare e scoprire più informazioni possibili riguardanti il sistema che si ha in esame. Tutte le informazioni disponibili sia di carattere tecnico che non, parlando quindi sia di scoprire il sistema operativo, le aree aperte all'attacco che informazioni riguardanti l'azienda.

La reconnaissance può essere effettuata in due tipi, a livello passivo e attivo. Nella prima vengono eseguite varie tipologie di ricerche, come quelle relative alla rete in esame, ai sistemi che sono direttamente connessi alla rete, l'attività aziendale ecc… Queste informazioni vengono poi verificate utilizzando la reconnaissance attiva, risulta quindi essere come un test per appunto chiarire se le informazioni prima identificate risultano effettivamente vere con metodi attivi.

Gli strumenti e le tecniche più comuni utilizzati per questa fase di reconnaissance sono:

- Social Engineering. La Social Engineering basa la sua natura esclusivamente sulla psicologia umana. In questo caso il pentester entra direttamente nell'azienda che richiede il servizio e con tecniche persuasive cerca di estrapolare più informazioni possibili dai suoi dipendenti. È quindi con la corruzione, con l'inganno che in questo caso si cerca di ottenere più informazioni specifiche possibili su uno specifico individuo o su tutto il bersaglio. Il tester deve essere bravo nel manipolare la psicologia umana, perchè nella maggior parte dei casi si sta cercando di corrompere personale fidato.

- Garbage Picking. È proprio nel posto più improbabile che a volte facciamo le scoperte più vantaggiose. Il cassonetto dei rifiuti può infatti fornire al pentester informazioni sensibili, sia a livello hardware che software. Documenti che vengono classificati come meno importanti, come bozze di lettere, documenti di posta, fogli aziendali vengono gettati senza alcuna considerazione. Questi documenti però possono fornire informazioni importanti per un pentester, come nomi, indirizzi, numeri di telefono e id del dipendete.

- Internet Footprinting. È il metodo di reconnaissance e target discovery che restituisce i risultati più importanti e significativi per lo svoglimento dell'attività. È un metodo di identificazione pulito, legale e sicuro.

  Vengono identificati 4 metodi di footprinting:

  - Web presence. Navigando sulle pagine web della società il pentester può raccogliere informazioni sul target dell'azienda come le informazioni sui dipendenti e documenti online della compagnia. Gli strumenti di ricerca è sicuramente il browser, i vari motori di ricerca, i siti web relativi alla sicurezza…

  - Network enumeration. È il processo che utilizza un particolare tool chiamato WHOIS. Il database WHOIS contiene informazioni relative all'assegnazione di indirizzi Internet, nomi di dominio e contratti individuali.

    Il risultato di WHOIS quindi include:

    - L'indirizzo
    - Nome del dominio
    - Informazioni di contatto amministrativo e tecnico, con nomi, numeri di telefono e indirizzo e-mail

- A list of Domain servers, with names and IP addresses
- Date and time of record creation
- Date and time when the record was last modified

- Domain Name System (DNS). Utilizza informazioni disponibili dai server riguardanti l'indirizzo IP. This method uses DNS lookup, DNS Zone Transfer tools like nslookup, dig, host.

- Network-based Reconnaissance. È il processo di identificazione dei computer e altri servizi attivi su una rete di destinazione tramite strumenti come ping, traceroute e netstat.

3.3.2.2 Scanning and Enumeration

La fase di scansione ci permette di identificare i sistemi presenti all'interno della rete target. Di questi sistemi possiamo poi vedere quali sono le porte aperte e quali quelle filtrate, i servizi che sono in esecuzione su queste porte, i dettagli del sistema operativo, i percorsi di rete ecc… La conoscenza di questi dettagli ci permette poi di individuare le vulnerabilità presenti nel sistema. Da questa fase in poi il tester deve essere cauto nel non sopraffare il sistema o la rete con un traffico eccessivo.

Alcuni degli strumenti più comuni utilizzati durante questa fase sono Nmap, Netcat, Superscan ecc…

3.3.3 Assessment Phase

Questa è la fase in cui le vulnerabilità ricoprono un ruolo molto importante, infatti è proprio in questo passaggio che vengono scoperte e poi analizzate. Questa fase è strettamente legata alla fase precedente, ovvero la fase di scoperta, perchè gli elementi rilevati nella discovery phase ora ci fanno da input per l'assessment phase.

3.3.3.1 Vulnerabilities Identification

In questa fase il pentester inizia a sondare i sistemi o le reti in maniera molto più approfondita rispetto alla discovery phase.

Tutti i dati riguardanti i sistemi operativi, gli indirizzi IP e i servizi analizzati, in questa fase vengono utilizzati per identificare le eventuali vulnerabilità e minacce. Le vulnerabi-

lità che costituiscono minacce in una rete includono bug software, errori di configurazione del sistema, account non protetti e servizi inutili.

Ci sono database di vulnerabilità come SecurityFocus[7], Exploit-Db[8] e PacketStrom[9] disponibili su Internet, che forniscono informazioni sulla vulnerabilità e le minacce.

3.3.3.2 Vulnerabilities Analysis

Tutte le vulnerabilità riscontrate vengono studiate ed analizzate. Il pentester deve comprendere lo stato di sicurezza che si trova all'interno di un sistema o di una rete e scoprire quali sono le vulnerabilità reali e quali sono falsi positivi.

Il tester per la scansione delle vulnerabilità può utilizzare sia strumenti automatizzati, che decidere di testare manualmente l'ambiente in analisi. Gli strumenti automatizzati dispongono di un proprio database che fornisce informazioni e dettagli sia sulle ultime che sulle vulnerabilità più vecchie.

3.3.4 Exploration Phase

Questa è la fase più impegnativa, ma anche affascinante di un penetration testing, perhcè è proprio qui che, se si è riusciti a trovare nelle fasi precedenti gli strumenti giusti, si riesce ad avere il pieno controllo del sistema in esame.

Una volta identificati quindi gli obiettivi giusti e scoperte ed analizzate su di loro le vulnerabilità presenti, è proprio grazie ad esse che si riesce a completare l'attacco. Se un attacco è riuscito, allora la vulnerabilità viene verificata e confermata e si prova ulteriormente ad ottenere privilegi più alti all'interno del sistema.

La fase di attacco infatti può essere suddivisa in:

1. Exploitation

2. Privilege Escalation

3.3.4.1 Exploitation

In questa fase vengono sfruttate le informazioni trovate per entrare nel sistema di destinazione. Questa fase deve essere eseguita con molta attenzione, perchè è proprio qui che tramite le vulnerabilità si riesce ad entrare nell'obiettivo, quindi devono essere considerati attentamente tutti i possibili effetti e soprattutto tutti i possibili exploit devono

essere provati in un ambiente controllato prima di eseguire procedure di test critiche, come l'utilizzo di exploit di buffer overflow.

L’utilizzo del framework di exploit ci permette di ridurre le tempistiche per il penetration. Metasploit è un framework open source che viene ampiamente utilizzato in questa fase.

3.3.4.2 Privilege Escalation

In questa fase il tester di penetrazione dovrebbe cercare modi per aumentarne l'accesso al sistema. Supponendo quindi di aver avuto accesso al sistema si dovrebbe poi cercare di ottenere il privilegio di root, se invece ha ottenuto accesso alla rete dovrebbe arrivare ad analizzare il traffico sulla rete per cercare di ottenere le informazioni sensibili.

Si può ricorrere all’utilizzo di rootkit o backdoor che aiutano a ottenere un livello di privilegi più elevato.

3.3.5 Reporting Phase

È proprio in questa fase che il tester è a conoscenza di tutte le debolezze che sono presenti nel sistema o nella rete. Sarà in grado infatti di dare informazioni dettagliate sulle vulnerabilità presenti nella fase di report.

I report devono contenere una valutazione delle vulnerabilità analizzando anche i possibili rischi che queste possono comportare e i possibili rimedi per eliminarle. In generale, il report finale serve appunto a far capire all’azienda qual’è la loro attuale sicurezza dei sistemi o della rete.

3.4 Penetration Testing Methodologies

Una metodologia deve specificare i vari procedimenti che devono essere eseguiti attraverso l'uso di tecniche specifiche. In questo caso una metodologia di penetration testing è una serie di regole o linee guida utilizzate per eseguire test di penetrazione su un sistema o rete di computer.

Una metodologia è quindi un aiuto, una sorta di linee guida che un pentester dovrebbe seguire per effettuare una buona valutazione, funziona come una tabella di marcia in cui ci sono procedimenti già collaudati utili per valutare correttamente la sicurezza di un sistema.

Ci sono diverse metodologie di penetration testing, non esiste una "metodologia giusta". Queste metodologie forniscono una valida documentazione che ci permette di seguire nel miglior modo, di fase in fase, l'attività di penetration testing, al fine di valutare con precisione la sicurezza di un sistema.

Esistono diverse metodologie anche per far si che il penetration tester riesca a scegliere quella più adeguata anche in base ai diversi contesti che si va ad analizzare.

3.4.1 OSSTMM

The Open Source Security Testing Methodology Manual (OSSTMM) [1] is absolutely standard for security testers. Descrive una completa tipologia di test, offrendo una buona guida e buoni strumenti per il penertation testing.

Questa metodologia si sviluppa per tre classi principali:

- PHYSSEC. Physical and Human Security Channels is:
    - Human: Comprises the human element of communication where interaction is either physical or psychological.
    - Physical: Physical security testing where the channel is both physical and non-electronic in nature. Comprises the tangible element of security where interaction requires physical effort.
- SPECSEC. full spectrum Wireless Security Channel is:

- Wireless: Comprises all electronic communications, signals, and emanations which take place over the known EM spectrum. This includes ELSEC as electronic communications, SIGSEC as signals, and EMSEC which are emanations untethered by cables.

• COMSEC. Telecommunications and Data Networks security Channels is:

- Telecommunications: Comprises all telecommunication networks, digital or analog, where interaction takes place over established telephone or telephone-like network lines.

- Data Networks: Comprises all electronic systems and data networks where interaction takes place over established cable and wired network lines.

Per scegliere la giusta tipologia di test è bene quindi capire come sono organizzati i moduli di questa metodologia.

Esistono 4 fasi differenti nell'esecuzione di questa metodologia, ognuna con una profondità diversa nell'analisi, ma nessuna singola fase è meno importante dell'altra nella sicurezza globale.

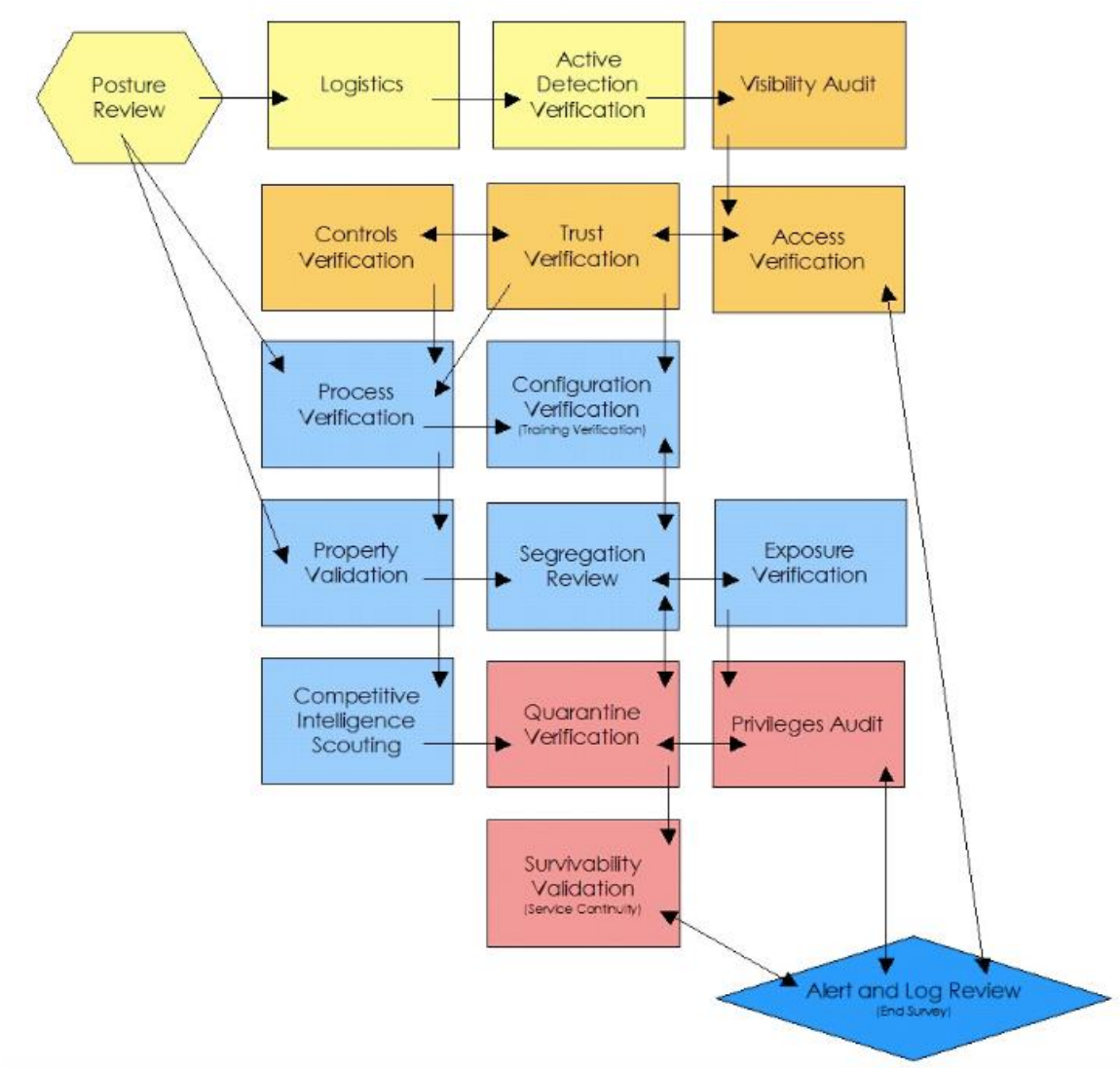

Figura 3.9: OSSTMM Methodology Phase [1]

3.4.1.1 INDUCTION PHASE

In questa fase il tester inizia ad avere una visione globale del campo di applicazione su cui andrà ad operare, è qui che viene definito infatti il tipo di test che andrà ad essere svolto.

- Posture Review: visione delle norme, delle regole, della legislazione dell'obiettivo.
- Logistics: misura dei vincoli come velocità e fallibilità per conoscere al meglio le limitazioni.

- Active Detection Verification: rilevazione delle interazioni, della risposta e della prevedibilità di risposta.

3.4.1.2 INTERACTION PHASE

In questa fase si mira a definire l'ambito del test. Definire l'ambito a volte è un processo molto lungo, perchè non sempre si riesce ad intuire subito cosa il tester dovrebbe cercare e quali tipi di test bisogna eseguire.

- Visibility Audit: si scopre quali sono i bersagli e come reagiscono con l'ambito. Un bersaglio morto o mancante è anche un obiettivo non rispondente.
- Access Verification: Il punto di accesso è il punto principale di qualsiasi asset di interazione. La verifica completa richiede conoscenza di tutto ciò che è necessario sapere sul punto di accesso.
- Trust Verification: La determinazione delle relazioni di fiducia da e tra gli obiettivi. Una relazione di trust esiste quando c'è interazione tra obiettivi nell'ambito.
- Control Verification: La misurazione dell'utilizzo e dell'efficacia dei controlli: riservatezza, privacy e integrità.

3.4.1.3 INQUEST PHASE

Fase in cui vengono valutate le i diversi tipi di informazioni che il tester scopre. Valutate quindi quanto sono svantaggiose e inadeguate le insicurezza che vengono rilevate.

- Process Verification: Determinazione dell'esistenza, dell'efficacia e della manutenzione che effettivamente il processo di sicurezza ha in atto.
- Configuration Verification / Training Verification: Verifica dello stato normale del target, ovvero identificare come funziona il target quando non è sotto "stress" (penetration testing).
- Property Validation: Valutazione dello stato di proprietà degli obiettivi.
- Segregation Review: Conoscenza dei diritti di privacy e classificazione delle informazioni personali.

- Exposure Verification: Informazioni sugli obiettivi e risorse ricornducibili da fonti pubbliche.
- Competitive Intelligence Scouting: Ricerca di informazioni liberamente disponibili che possono danneggiare il possessore del target

3.4.1.4 INTERVENTION PHASE

Descrive i test pratici riguardanti le informazioni precedentemente raccolte. Questi test quindi si concentrano sulle risorse del target e queste risorse possono essere modificate, sovraccaricate per causare penetrazioni o disturbi.

- Quarantine Verification: Determinare se risulta efficace l'utilizzo della quarantena per tutto l'accesso all'interno del bersaglio.
- Privileges Audit: La mappatura e la misurazione dell'uso improprio dei controlli, delle credenziali e dei privilegi.
- Survivability Validation / Service Continuity: La determinazione e la misura della resistenza del bersaglio a cambi eccessivi o negativi.
- Alert and Log Review / End Survey: Una revisione delle attività di ascolto svolte. Questa parte ci permette di sapere quali parti della revisione ha lasciato un percorso utile e affidabile. Vengono verificati i test precedenti che non hanno fornito alcuna interattività all'Analista. La maggior parte dei test di sicurezza che non includono questa fase può ancora avere bisogno di eseguire una revisione finale per chiarire eventuali anomalie.

3.4.2 OWASP

Open Source Open Web Application Security Project (OWASP) [2] è una metodologia prettamente utilizzata per testare la sicurezza delle applicazioni web. È una metodologia open source che fornisce un framework di test per le applicazioni web basate su http.

Di seguito, nell'analisi dettagliata delle varie fasi di Owasp, vedremo usare una specifica nomenclatura. Ad esempio, OWASP-IG-001, sta a significare che si parla della prima fase dell'information gathering (IG) di OWASP.

La guida di OWASP per le applicazioni web si compone principalmente dei seguenti passi:

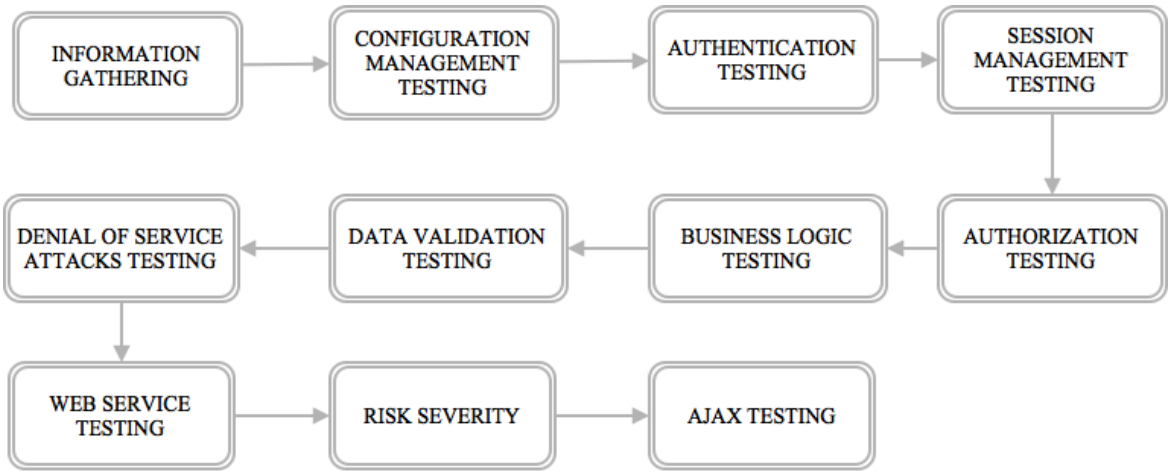

Figura 3.10: OWASP Methodolody Phase

3.4.2.1 Information gathering

La prima fase è rivolta alla raccolta delle informazioni dell'applicazione interessata, che è un passo necessario ed importante durante il penetration testing. Si può effettuare questa raccolta di informazioni utilizzando tools pubblici, scanner, inviando semplici richieste HTTP o richieste appositamente create.

Il dettaglio dei passi effettuati è:

- Spiders, Robots, and Crawlers (OWASP-IG-001)
- Search Engine Discovery/Reconnaissance (OWASP-IG-002)
- Identify application entry points (OWASP-IG-003)
- Testing Web Application Fingerprint (OWASP-IG-004)
- Application Discovery (OWASP-IG-005)
- Analysis of Error Codes (OWASP-IG-006)

3.4.2.2 Configuration management testing

Possiamo arrivare a sapere molte informazioni importanti con l'analisi dell'architettura dell'infrastruttura. Informazioni come il codice sorgente, i metodi HTTP usati, i metodi di autenticazione sono informazioni che ci servono per svolgere un buon penetration testing.

I passi effettuati in dettaglio:

- SSL/TLS Testing (OWASP-CM-001)
- DB Listener Testing (OWASP-CM-002)
- Infrastructure Configuration Management Testing (OWASP-CM-003)
- Application Configuration Management Testing (OWASP-CM-004)
- Testing for File Extensions Handling (OWASP-CM-005)
- Old, Backup and Unreferenced Files (OWASP-CM-006)
- Infrastructure and Application Admin Interfaces (OWASP-CM-007)
- Testing for HTTP Methods and XST (OWASP-CM-008)

3.4.2.3 Authentication testing

Letteralmente autenticazione significa risalire alla provenienza di ciò che si ha davanti, di ciò che si sta analizzando. Nella sicurezza informatica l'autenticazione è il processo, ad esempio il processo di accesso, che ci permette di identificare l'identità digitale del mittente di una comunicazione. Testare lo schema di autenticazione significa comprendere come funziona il processo di autenticazione e utilizzare queste informazioni per aggirare questo processo.

Authentication testing si suddivide in:

- Credentials transport over an encrypted channel (OWASP-AT-001)
- Testing for user enumeration (OWASP-AT-002)
- Testing for Guessable (Dictionary) User Account (OWASP-AT-003)
- Brute Force Testing (OWASP-AT-004)
- Testing for bypassing authentication schema (OWASP-AT-005)
- Testing for vulnerable remember password and pwd reset (OWASP-AT-006)
- Testing for Logout and Browser Cache Management (OWASP-AT-007)
- Testing for CAPTCHA (OWASP-AT-008)

- Testing Multiple Factors Authentication (OWASP-AT-009)
- Testing for Race Conditions (OWASP-AT-010)

3.4.2.4 Session management testing

La parte importante di ogni applicazione web è analizzare il modo in cui controlla l'iterazione dell'utente con il sito. Questa fase infatti gestisce tutti i controlli effettuati ad un utente, dal momento in cui effettua l'autenticazione fino a quando esce dall'applicazione.

Esistono diversi modi in cui l'applicazione interagisce che con un utente, che differiscono a seconda della natura del sito, della sicurezza e delle esigenze dell'applicazione.

Questa fase si struttura con i seguenti passi:

- Testing for Session Management Schema (OWASP-SM-001)
- Testing for Cookies attributes (OWASP-SM-002)
- Testing for Session Fixation (OWASP-SM-003)
- Testing for Exposed Session Variables (OWASP-SM-004)
- Testing for CSRF (OWASP-SM-005)

3.4.2.5 Authorization testing

L'autorizzazione consiste del suddividere determinati utenti a seconda delle risorse che sono portati ad usare.

In questa fase viene quindi capito come funziona il processo di autorizzazione e come vengono utilizzate queste informazioni per cercare di aggirare questo meccanismo. L'autorizzazione è un processo che viene dopo un'autenticazione riuscita e il tester deve identificare se quest'operazione è andata a buon fine e classificare bene quali privilegi appartengono a quell'utente. Viene poi verificato se è possibile bypassare lo schema di autorizzazione e si va alla ricerca di qualche vulnerabilità di percorso, per cercare di scalare i privilegi.

Authorization testing si suddivide in:

- Testing for Path Traversal (OWASP-AZ-001)

- Testing for bypassing authorization schema (OWASP-AZ-002)
- Testing for Privilege Escalation (OWASP-AZ-003)

3.4.2.6 Business logic testing

In questa fase viene testato se l'applicazione, in funzione della logica aziendale, funziona come dovrebbe. Se ad esempio, durante l'autenticazione si cerca di oltrepassare un passaggio di questo processo, dobbiamo vedere come l'applicazione risponde, cioè se ci da un messaggio di errore o se funziona ugualmente. È una vulnerabilità che i scanner di vulnerabilità non identificano e che si basa sulle competenze e sulla creatività del tester.

Gli attacchi sulla logica aziendale di un'applicazione sono pericolosi, difficili da individuare e sono di solito specifici per l'applicazione in fase di test.

3.4.2.7 Data validation testing

In questa fase vengono testate tutte le possibili forme di input, per capire se l'applicazione convalida i dati di input prima di utilizzarla.

I dati che provengono da un'entità esterna non devono essere mai attendibili, perchè possono essere alterati volutamente. Questo è un problema che molte applicazioni complesse hanno, in quanto, avendo molti punti di accesso, risulta difficile per uno sviluppatore applicare la regola di controllo.

Questa è una problematica molto comune nelle applicazioni web, dovuta dall'insuccesso di convalidare correttamente l'input che proviene da un'entità esterna. Si ricavano così tutte le principali vulnerabilità nelle applicazioni web, such as cross site scripting, SQL injection, interpreter injection, locale/Unicode attacks, file system attacks, and buffer overflows.

Abbiamo suddiviso Data validation testing nelle seguenti categorie:

Testing for Cross site scripting (XSS)

Si verifica se è possibile manipolare i parametri di input dell'applicazione in modo che genera un'uscita dannosa.Troviamo una vulnerabilità XSS quando l'applicazione non convalida il nostro input e crea un'output che è sotto il nostro controllo. Sfruttando questa vulnerabilità possiamo rubare informazioni

riservate (come i cookie di sessione) o assumere il controllo del browser della vittima.

I passi si suddividono in:

- Testing for Reflected Cross Site Scripting (OWASP-DV-001)
- Testing for Stored Cross Site Scripting (OWASP-DV-002)
- Testing for DOM based Cross Site Scripting (OWASP-DV-003)
- Testing for Cross Site Flashing (OWASP-DV004)
- SQL Injection (OWASP-DV-005):

  Qui verifichiamo se è possibile iniettare i dati nell'applicazione in modo che esegua una query SQL controllata dall'utente nel DB back-end. Troviamo una vulnerabilità se l'applicazione utilizza l'input utente per creare query SQL senza una corretta convalida dell'inserzione. Con questa vulnerabilità un utente non autorizzato piuò accedere o manipolare i dati nel database.

  Si suddivide all'interno in:

  - Oracle Testing
  - MySQL Testing
  - SQL Server Testing
  - MS Access Testing
  - Testing PostgreSQL
- LDAP Injection (OWASP-DV-006)
- ORM Injection (OWASP-DV-007)
- XML Injection (OWASP-DV-008)
- SSI Injection (OWASP-DV-009)
- XPath Injection (OWASP-DV-010)
- IMAP/SMTP Injection (OWASP-DV-011)
- Code Injection (OWASP-DV-012)

- OS Commanding (OWASP-DV-013)
- Buffer overflow (OWASP-DV-014):

  Vengono qui controllati diversi tipi di vulnerabilità del buffer overflow.

  Seguono questi passaggi:

  - Heap overflow
  - Stack overflow
  - Format string
- Incubated vulnerability Testing (OWASP-DV-015)
- Testing for HTTP Splting/Smuggling(OWASP-DV-016)

3.4.2.8 Denial of service attacks testing

Il concetto di attacco Denial of Service (DoS) di rete si presenta quando un utente dannoso cerca di mandare abbastanza traffico sulla macchina obiettivo e la macchina non è in grado di reggere le richieste ricevute. Questo è uno dei più comuni attacchi DoS che generalmente uno sviluppatore non può prevenire, ma si può solo attenuare tramite delle soluzioni nell'architettura di rete.

Esistono anche però delle vulnerabilità all'interno dell'applicazione che consentono ad un utente malintenzionato di eseguire determinate funzionalità, o rendere addirittura non disponibile il servizio.

Parlando di questo possiamo suddivide la fase in:

- Testing_for_SQL_Wildcard_Attacks (OWASP-DS-001)
- D Locking Customer Accounts (OWASP-DS-002)
- Buffer Overflows (OWASP-DS-003)
- User Specified Object Allocation (OWASP-DS-004)
- User Input as a Loop Counter (OWASP-DS-005)
- Writing User Provided Data to Disk (OWASP-DS-006)
- Failure to Release Resources (OWASP-DS-007)

- Storing too Much Data in Session (OWASP-DS-008)

3.4.2.9 Web services testing

Le applicazioni web sono sistemi sempre in continua evoluzione e crescita.

Il Web Services Framework utilizza il protocollo HTTP (come applicazione Web standard) in combinazione con le tecnologie XML, SOAP, WSDL e UDDI:

• La "Web Services Description Language" (WSDL) viene utilizzata per descrivere le interfacce di un servizio.

• "Simple Object Access Protocol" (SOAP) fornisce i mezzi per la comunicazione tra servizi web e applicazioni client con XML e HTTP.

• "Universal Description, Discovery and Integration" (UDDI) è utilizzata per registrare e pubblicare i servizi web e le loro caratteristiche in modo che possano essere trovati da potenziali clienti.

Le vulnerabilità dei servizi web sono simili a altre vulnerabilità, come l'iniezione SQL, la divulgazione di informazioni e le perdite, ma i servizi web dispongono anche di unica vulnerabilità legata a XML, che analizziamo con i seguenti passaggi:

- WS Information Gathering (OWASP-WS-001)
- Testing WSDL (OWASP-WS-002)
- XML Structural Testing (OWASP-WS-003)
- XML Content-level Testing (OWASP-WS-004)
- HTTP GET parameters/REST Testing (OWASP-WS-005)
- Naughty SOAP attachments (OWASP-WS-006)
- Replay Testing (OWASP-WS-007)

3.4.2.10 Risk severity

3.4.2.11 AJAX testing

AJAX, an acronym for Asynchronous JavaScript and XML, is a web development technique used to create more responsive web applications.

Utilizzando le tecniche AJAX possiamo avere molti vantaggi per le applicazioni web, mariscontriamo problemi nella sicurezza perchè queste applicazioni essendo eseguite sia lato client che server sono più complicate e quindi hanno una superficie di attacco maggiore rispetto alle normali applicazioni web.

Le applicazioni AJAX sono vulnerabili all'intera gamma di vulnerabilità delle applicazioni tradizionali.

Nelle applicazioni web tradizionali, i moduli HTML standard inviati tramite richieste GET o POST hanno un formato facilmente comprensibile, quindi è facile modificare o creare nuove richieste ben formate. Le applicazioni AJAX spesso utilizzano schemi di codifica diversi per inviare i dati POST rendendo più difficile la funzionalità dei tools di testing.

Affronteremo quindi questi argomenti:

- AJAX Vulnerabilities (OWASP-AJ-001)
- How to test AJAX (OWASP-AJ-002)

3.4.3 TOP 10 OWASP

Owasp tenta ogni anno di aumentare la sensibilità sulla sicurezza applicativa stilando una lista delle 10 vulnerabilità più rischiose di quell'anno, identificando quindi quali sono i principali rischi a cui le organizzazioni IT sono rivolte.

Questa lista è presa come riferimento da numerosi standard, libri, tool e organizzazioni come MITRE, PCI DSS, DISA, FTC, e molte altre. OWASP incoraggia l'uso della Top 10 affinchè le vostra organizzazioni inizino a considerare la sicurezza applicativa.

La top 10 dell'anno 2017:

1. Injection.

2. Broken Authentication and Session Management
3. Cross-Site Scripting (XSS)
4. Broken Access Control
5. Security Misconfiguration
6. Sensitive Data Exposure
7. Insufficient Attack Protection
8. Cross-Site Request Forgery (CSRF)
9. Using Components with Known Vulnerabilities
10. Underprotected APIs
11.

3.5 ISSAF

The Information Systems Security Assessment Framework (ISSAF) [3] è una metodologia di penetration testing sviluppata da OISS.org.

Questa metodologia è utilizzata per coprire una gran parte di tipologie di testing, infatti ci permette di controllare sistemi, intere reti e applicazioni.

Vediamo in dettaglio questa metodologia che è divisa in tre blocchi principali, fornendo però ulteriori dettagli per quanto riguarda la parte culmine dell'attività.

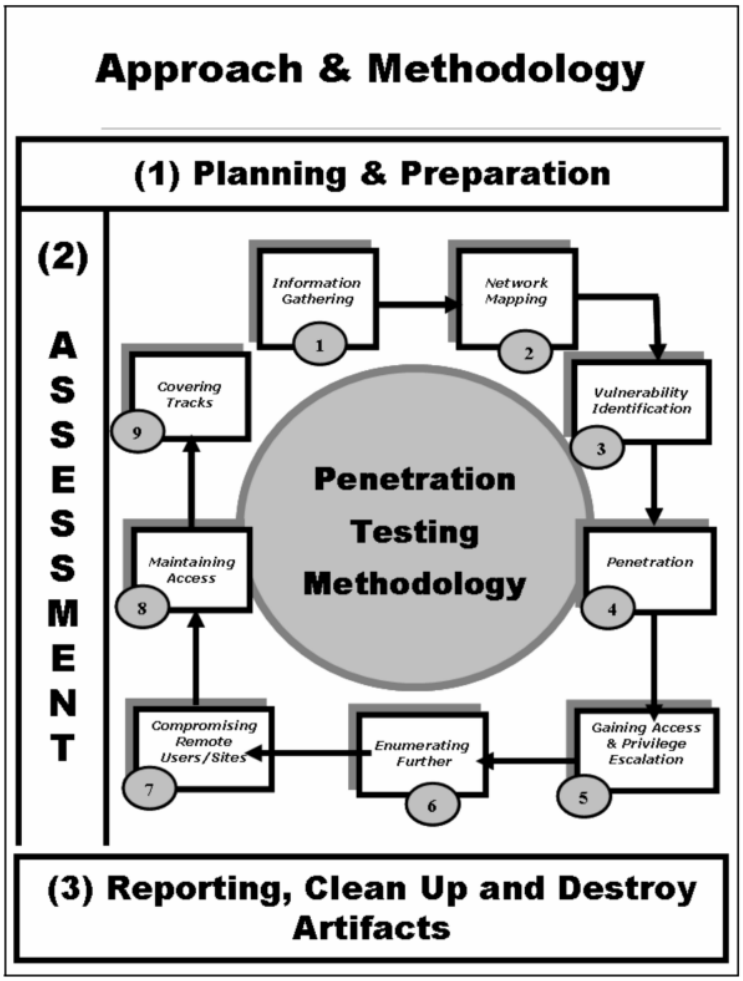

Figura 3.11: ISSAF Methodology Phase [23]

3.5.1 Planning and Preparation.

Questa fase si concentra esclusivamente sulla pianificazione e preparazioni di tutti gli apparati che successivamente andremo ad usare. Vengono stipulati qui gli accordi tra l'azienda e il team di pentration testing, vengono stipulati i contratti e visto tutto l'ambito legale dell'attività.

3.5.2 Assessment.

Questa è la fase più articolata del penetration testing, è qui che viene messo alla prova il sistema in esame. Essendo quindi la fase più importante tra le 3, vediamo in dettaglio i suoi passaggi:

3.5.2.1 Information Gathering.

Information Gathering consiste in una prima analisi del sistema che si ha in esame. Qui vengono raccolti tutti i possibili dettagli dell'obiettivo, tutte le informazioni che ci serviranno a capire come muoverci nel miglior modo per poter eseguire una valutazione il più dettagliata possibile. È una fase che alcune metodologie erroneamente tralasciano, ma abbiamo bisogno di conoscere il più possibile il nostro obiettivo, fornendo una panoramica completa del bersaglio per esplorare quindi ogni possibile via di attacco. Nella maggior parte dei casi la fonte principale di informazioni per questa fase è Internet.

3.5.2.2 Network Mapping.

Le informazioni specifiche di rete che sono appena state scoperte vengono ora utilizzate per ipotizzare una possibile mappa della rete del sistema. Si è in questa fase alla ricerca di informazioni tecniche sugli host e sulle reti coinvolte, informazioni come host attivi, firewall, sistemi operetivi coinvolti ecc… anche con l'aiuto di strumenti e applicazioni.

3.5.2.3 Vulnerability Identification.

Tutte le informazioni acquisite ci servono ora per avere un quadro generale di quali possono essere i possibili host o server vulnerabili. Verrà poi effettuata una valutazione di questi host, ovvero verranno scansionati e controllati di tutte le vulnerabilità esistenti. L'obiettivo di questa fase, infatti, è quello di utilizzare le informazioni raccolte in precedenza per effettuare una valutazione tecnica dell'esistenza di vulnerabilità. Queste vulnerabilità vengono cercate sia tramite l'utilizzo di tool automatici di vulnerability assessment, sia testando i servizi web per le vulnerabilità come l'iniezione di XSS e SQL.

3.5.2.4 Penetration.

Le vulnerabilità prima identificate vengono ora messe alla prova. Sfruttandole al massimo ci permettono di penetrare nel sistema o rete vulnerabile.

3.5.2.5 Gaining Access and Privilege Escalation.

Quando il tester è riuscito ad entrare nel suo bersaglio, può impegnarsi per riuscire ad ottenere tutti i permessi nella macchina e diventare quindi utente root.

3.5.2.6 Enumerating Further.

Ottenuto l’accesso e i massimi privilegi il penetration tester può effettuare:

- Attacchi di password.
- Sniffare il traffico e analizzarlo.
- Raccolta di cookie
- Raccolta di indirizzi e-mail
- Identificazione di percorsi e reti
- Mappatura di reti interne

3.5.2.7 Compromise Remote Users Sites.

Un solo accesso può compromettere tutta la rete perimetrale. Una volta dentro il penetration tester dovrebbe quindi cercare di espandersi all’interno della rete, compromettere gli utenti remoti o siti remoti di un’impresa.

3.5.2.8 Maintaining Access.

Dopo aver ottenuto un accesso iniziale, il penetration tester dovrebbe provare a far si che questo accesso rimanga più a lungo possibile. Anche se l'utilizzo di canali di copertura, back door installation and deployment of rootkits spesso non vengono eseguiti come parte di un test di penetrazione, a causa del rischio che si verifica se uno di questi rimane aperto durante o dopo il test e viene rilevato da un attaccante.

3.5.2.9 Covering Tracks.

Il penetration tester deve nascondere tutte le tracce che ha lasciato all’interno del sistema per manometterlo. L'obiettivo principale è quindi quello di nascondere gli strumenti/exploit usati durante il test.

3.5.3 Reporting, Clean-up and Destroy Artifacts.

La fase finale di questa attività consiste nel redigere un report che illustri tutta le operazioni effettuare durante il penetration testing. I tester devono anche far tornare il

sistema al punto di partenza e quindi distruggere tutti i cambiamenti effettuati durante l'assessment phase.

3.6 Comparison Analysis

Una buona metodologia, per far si che sia tale, deve considerare diversi aspetti che a volte vengono erroneamente tralasciati. Partendo dall'inizio del percorso di penetration testing, già dalla fase di ingaggio è importante adottare le disposizioni legali, osservare bene le condizioni relative alla gestione dell'azienda e dei dipendenti. Si deve tener conto del tempo disponibile per effettuare il test, includendo anche il fattore dei potenziali rischi che si possono riscontrare.

Una metodologia dovrebbe quindi prendersi molta cura del processo di selezione, determinando quali sono i costi e l'efficacia del test a livello ottimale. Non dovremmo quindi soffermarci sull'aspetto puramente tecnico dell'attività, ma concentrarci nella prima fase sull'aspetto manageriale.

Determinare la giusta strategia di valutazione dipende da diversi fattori, inclusi i dettagli tecnici forniti sull'ambiente di destinazione, sulla disponibilità di risorse, sulle conoscenze di PenTester, sugli obiettivi aziendali e sulle preoccupazioni normative [26].

Una metodologia di test di penetrazione è come una "mappa" con cui il tester può raggiungere la destinazione finale (cioè la fine di un test di successo) seguendo però una metodologia che non lo faccia "perdere" durante il tragitto. (Ossia test incompleto, spreco di tempo e fatica).

3.6.1 OSTMM Analysis

OSTMM è diventata una delle metodologie più utilizzate, in quanto descrive una serie di azioni e fasi molto nel dettaglio.

È la prima metodologia ad includere i "fattori umani", o meglio l'ingegneria sociale, rendendosi conto che a volte la vulnerabilità più importante la possiamo riscontrare all'interno dell'azienda stessa, capendo che a volte gli essere umani possono essere più pericolosi dei sistemi.

OSTMM risulta essere quindi una grande metodologia, però pecca in qualche punto. Essendo molto incentrata sull'analisi della comunicazione, sulla parte "non-pratica" dell'attività, tende a tralasciare alcuni concetti chiave come: control flow analysis, induced hypotheses, readable diagrams, data loss process on writing reports and traceable reports.

Control flow analysis non è molto presente, come anche induced hypotheses non vengono considerate, eliminando così una serie significativa di possibili percorsi di scoperta di vulnerabilità, utili per il tester.

Utilizzare poi una metodologia composta da 17 moduli, ha i suoi pro e i suoi contro. Certo non possiamo assolutamente dire che questa non è una metodologia dettagliata e molto spiegata per i punti a cui fa maggior riferimento, però diciamo anche che non è una metodologia molto intuitiva. Il tester dovrebbe aver bisogno di un flusso semplice e diretto, visto che il suo compito è quello di trovare vulnerabilità e non di interpretare le metodologie.OSSTMM non fornisce tale processo intuitivo, principalmente a causa del numero elevato di articoli e di molti cicli disordinati all'interno del flusso.

Anche il suggerimento che questa metodologia fornisce nella scrittura di un report non risulta abbastanza chiaro e facile da seguire. Sapendo che l'attività di penetration testing è un processo lungo e complicato, buona idea sarebbe la possibilità di scrivere il report passo dopo passo durante l'attività. La metodologia OSTMM offre modelli ben organizzati per riempire i report, molto dettagliati e che però seguono un processo strettamente lineare di lettura. Per conoscere quindi la sicurezza di un sistema, il lettore deve leggere l'intero report. Questo approccio può essere positivo perchè porta ad una maggiore comprensione, negativo perchè ci sono figure che devono trovare subito gli aspetti specifici dell'attività, come le commissioni tecniche.

3.6.2 OWASP Analysis

È più limitato rispetto agli altri standard, ma copre in dettaglio la sua area.

La Guida di Test di OWASP è un'ottima descrizione dei numerosi tipi di test che possono essere effettuati nella sua area, fornendo anche un'ampia selezione di strumenti da utilizzare nel processo penetration testing delle applicazioni web.

La metodologia OWASP, con la sua top 10 stilata ogni anno, prova a determinare quali sono i maggiori rischi per le organizzazioni e aumentare la consapevolezza della sicurezza.

La guida non si concentra sui programmi di sicurezza applicativi, ma fornisce una base necessaria per integrare la sicurezza attraverso principi e pratiche di codifica sicura. Classifica i rischi per la sicurezza delle applicazioni valutando i principali vettori di attacco e le debolezze di sicurezza in relazione al loro impatto tecnico e commerciale.

OWASP risulta essere una guida molto dettagliata e articolata, formata da numerose fasi e attività. Svolge molto bene il ruolo di guida per la sua specifica area che ricopre, l'unica pecca che ha è che risulta essere poco intuitiva e molto ramificata nei suoi dettagli.

3.6.3 ISSAF Analysis

ISSAF ha una struttura chiara e molto intuitiva che guida il tester attraverso i passi di valutazione complicati.

Segue un ordine molto chiaro e comprensibile, descrivendo il processo di penetration testing cercando di evitare gli errori più comuni, al fine di aiutare il tester ad eseguire un'attività completa e corretta.

ISSAF si concentra poco sul lato della ricerca delle informazioni, quelle informazioni che aiuterebbero il tester alla scoperta delle vulnerabilità.

È invece molto ben spiegata la fase della scoperta e della valutazione delle vulnerabilità, contrariamente alla fase di reporting. Non esiste infatti una linea guida ben definita e precisa che permetta di sviluppare al meglio i report finali.

Vengono anche forniti alcuni suggerimenti obsoleti, come ad esempio il suggerimento di distruggere e ripulire le tracce lasciate durante l'attività, anche se la pratica più attuale è quella di lasciare questi artefatti sul sistema sperimentato in modo da facilitare un'analisi ulteriore e più approfondita.

3.6.4 Conclusions

Possiamo quindi concludere dicendo che la fase di pianificazione dell'intera attività risulta essere molto importante e che essa ci permette di avere una buona visione di tutta

l’attività che andremo successivamente a svolgere. Con poche informazioni riguardo questa fase troveremo confusioni nelle fasi successive.

Risulta essere molto importante anche la flessibilità e adattabilità nella guida di una metodologia, una metodologia troppo lineare e dettagliata non sempre risulta essere un vantaggio per chi ad esempio può avere delle situazioni diverse da quelle descritte. Risulta essere quindi l'obiettivo più impegnativo: rendere la metodologia chiara e intuitiva nonostante la necessaria complessità. Una metodologia deve guidare il tester attraverso il processo di penetration testing, senza distrarla con oneri aggiuntivi.

Anche la stesura del report è un processo importante. Il penetration testing richiede sforzi significativi e spesso si estende per un lungo periodo di tempo. Per questo motivo, la segnalazione dei risultati di ciascuna fase prima di passare alla successiva è una buona pratica da rispettare, infatti una buona metodologia non dovrebbe costringere il tester a perdere tempo nel porre molti dettagli per uno specifico contesto.

Il report deve essere scritto per consentire una lettura bidirezionale: dall'inizio alla fine , o dalla fine all’inizio, a seconda del tipo di utilizzo che fa il lettore delle informazioni fornite.

Sulla base di queste considerazioni possiamo fornire la seguente tabella**:**

	ISSAF	OSSTMM	OWASP
Modeling			
Planning			
Flexibility			
Adaptation			
Guidance			
Reporting			

Table 3.1: Feature map of the security testing methodologies

4 Vulnerability Assessment VS Penetration Testing

4.1 What is Vulnerability Assessment?

L’argomento della sicurezza informatica è una questione molto trattata e risaputa negli ultimi anni, molte infatti sono le documentazioni presenti in questo ambito, ma nonostante questo risultano essere molte le debolezze in ambito di sicurezza delle piccole e grandi organizzazioni.

Le documentazioni presenti sono approfondimenti riguardanti queste vulnerabilità, come vengono valutate, come un’organizzazione ne può beneficiare, ma se si va nello specifico a ricercare quali sono le vulnerabilità più comuni che si possono riscontrare all’interno di un’impresa, si insorge in piccole difficoltà, perchè non risulta così semplice trovarle.

La letteratura, che si concentra sulla vulnerabilità delle reti, si è rapidamente sviluppata nel corso degli ultimi dieci anni. È compito importante che un’azienda deve svolgere, infatti, rimanere sempre informati ed aggiornati su quali sono le falle della sicurezza informatica, esaminare tutti i livelli di protezione dei dati per attenuare il rischio.

Una valutazione della vulnerabilità ci permette quindi di avere una panoramica della posizione attuale dell’azienda, documentando quali sono le politiche e i controlli di sicurezza che questa sta adottando.

Una valutazione di vulnerabilità è il processo di esecuzione di strumenti manuali e automatizzati contro un set definito di indirizzi IP o intervalli IP per identificare le vulnerabilità noti e potenziali in un ambiente IT.

Gli indirizzi IP sono tutti quei dispositivi attivi e connessi ad una rete che possono essere scansionati per controllare i servizi e i protocolli che girano su di essi.

Il software commerciale VA è disponibile fin dagli anni '90, ma non ha guadagnato popolarità fino all'inizio del XXI secolo. Il software di valutazione di vulnerabilità è stato sviluppato per aiutare a trovare ed individuare i problemi presenti nel software, cercando

le cause delle vulnerabilità come si possono trovare nel codice, nella progettazione che nell'architettura del sistema.

Un database nazionale di vulnerabilità gestito dalla CERT (Computer Emergency Response Team) Nazionale Italia [11] riporta almeno dieci nuove vulnerabilità al giorno. I crescenti incidenti di hacking e aumento dei requisiti di conformità alla sicurezza IT per le aziende hanno provocato l'evoluzione del processo Vulnerability Assessment.

Lo svolgimento di un vulnerabilty assessment segue diverse fasi, proprio come il pentration testing, anzi possiamo proprio dire che solitamente le prime fasi di un penetration testing sono le fasi che andranno a formare un vulnerability assessment.

In dettaglio, sono 4 le fasi che vengono affrontate nel vulnerability assessment.

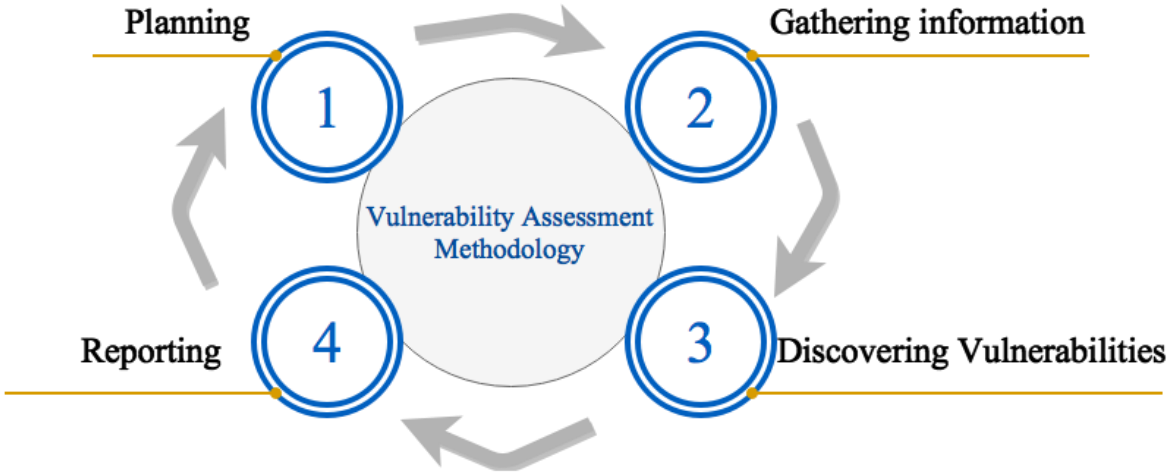

Figura 4.1: Vulnerability Assessment Phase

1. Planning. Lavorare con il cliente per definire e documentare chiaramente gli obiettivi di valutazione, l'ambito e le regole di ingaggio.
2. Information gathering. Vengono raccolte e analizzate le informazioni del sistema in esame.
3. Discovering vulnerabilities. Usate tecniche sia automatiche che manuali per la ricerca delle vulnerabilità.
4. Reporting. Fornita una relazione completa con analisi approfondite e raccomandazioni su come mitigare le vulnerabilità scoperte.

Vogliamo sottolineare che il Vulnerability Assessment, come il penetration testing, è un ciclo di fasi, ognuna molto importante, come risulta importante non limitarsi ad eseguire

questo tipo di attività solo una volta, ma ripeterla con frequenza e costanza all'interno dell'azienda.

Viene quindi classificata come un ciclo di fasi dato l'alto numero di nuove vulnerabilità che vengono scoperte ogni giorno ed è fondamentale svolgere un Vulnerability assessment con la giusta frequenza al fine di assicurarsi che le configurazioni dei sistemi siano corrette e le opportune patch di sicurezza applicate.

4.2 Vulnerability Assessment VS Penetration Testing

Dando quindi una definizione di vulnerabilità possiamo dire che sono delle debolezze riscontrate all'interno di un sistema, di una rete. Sono delle falle che possono essere sfruttate da un utente malevolo che vuole intrufolarsi nell'obiettivo.

Possiamo dire che esistono due tipologie differenti di vulnerabilità: vulnerabilità logica e vulnerabilità fisica.

Le vulnerabilità fisiche sono quelle vulnerabilità che si trovano effettivamente all'interno di un azienda: informazioni sensibili troppo reperibili facilmente nell'azienda oppure informazioni possibilmente ricavabili tramite i dipendenti della stessa azienda attraverso diversi approcci di ingegneria sociale.

Le vulnerabilità logiche sono invece associate ai computer aziendali, ai dispositivi di infrastruttura, al software e alle applicazioni.

C'è una confusione globale nella classificazione di questa attività perchè molti fornitori dicono di offrire determinati servizi, invece poi si rivelano differenti. Esistono però 3 livelli comuni di servizio chiamati port scanning, vulnerability assessment, and penetration testing.

Il port scanning viene effettuato tramite l'utilizzo di un software che ci permette di individuare le eventuali porte aperte, il vulnerability assessment si usa per rilevare le possibili minacce presenti nella rete o nel sistema.

Il penetration testing fa uso di tool che ti permettono di facilitare il processo di test per tentare di sfruttare il sistema e di solito si tenta di infiltrarsi dall'esterno dei sistemi, delle reti o delle applicazioni web tramite internet.

Il test di penetrazione interna, al contrario, fornisce informazioni su ciò che un hacker potrebbe fare al sistema una volta che le contromisure esterne sono violate con successo. I penetration testing sono differenti dai vulnerability assessment perchè i primi utilizzano tecniche manuali con l'integrazione di strumenti automatizzati per sfruttare il sistema, mentre i vulnerability assessment dipendono nella maggior parte dei casi da strumenti automatizzati che ci permettono di rilevare le debolezze del sistema.

Andando quindi infine a specificare quale delle due attività ci forniscono un miglior processo per valutare la sicurezza delle informazioni. Paul Paget of Core Security Technologies, and Ron Gula of Tenable Network Security, affrontano questo argomento.

Paget [12] afferma che il vulnerability assessment non ti risolve il problema di una probabile intrusione. Effettuando un vulnerability assessment devi poi andare ad analizzare queste vulnerabilità per capire se possono essere o no dannose per il sistema in esame. I penetration testing, invece, tentano effettivamente di intrufolarsi nel sistema usando le tecniche che un hacker potrebbe utilizzare. I risultati di un penetration testing producono molto di più rispetto ai risultati di un vulnerability assessment, aiutano così gli amministratori a individuare rapidamente le vulnerabilità reali, e ottenere approfondimenti sull'efficacia delle misure di sicurezza in atto.

Gula [13] sostiene che il vulnerability assessment è più adatto per un'efficace gestione delle vulnerabilità perchè la scansione delle vulnerabilità può essere automatizzata in contrapposizione ai test di penetrazione che devono necessariamente eseguire un team di esperti. il vulnerability assessment provoca un numero più ampio di vulnerabilità su più piattaforme rispetto agli strumenti di test di penetrazione tipici. Il vulnerability assessment garantisce una maggiore fedeltà delle informazioni, offrendo una gamma più vasta di vulnerabilità che aiuta il team di sicurezza a dare un quadro più ampio della situazione.

Possiamo concludere dicendo che entrambe le attività vengono scelte a seconda della richiesta del cliente.

Se la società desidera chiarire il numero di potenziali debolezze di sicurezza esistenti nel loro sistema, un vulnerability assessment è apparentemente una scelta adeguata. Se, invece, l'obiettivo è quello di verificare come le vulnerabilità influiscono nel sistema aziendale, un penetration testing ben programmato è ovviamente una decisione razionale.

	VULNERABILITY ASSESSMENT	PENETRATION TESTING
Purpose	Identify and rank vulnerabilities.	Identify ways to exploit vulnerabilities.
When	Quarterly or after significant changes.	Annually or after significant changes.
How	Automated tools are better set up with manual rules. This analysis can also be performed manually.	Automated tools are used manually by experts.
Report	Contains vulnerability ratings based on the risk of these vulnerabilities identified.	Described each vulnerability that was found and shown how it was exploited
Duration	Relatively short, several minutes per hosts.	Depends on size of systems and issues uncovered during testing, usually days to weeks
Cost	There are free vulnerability assessment tools, depending on choice of the tester.	Depends on the choice of the tester and the duration of the activity.

Table 4.1: Comparison between Vulnerability Assessment and Penetration Testing

5 Main tools for Penetration Testing

Ci sono diverse documentazioni orientate sull'uso degli strumenti di sicurezza. In questa parte andremo infatti a descrivere questi tool che possono essere utilizzati per condurre il penetration testing.

Possono essere suddivisi in:

5.1 Service and Network Mapping Tools

Service and Network Mapping Tools sono strumenti utilizzati per analizzare sistemi, reti, serivizi e porte aperte. Vengono analizzate le regole di firewall o le risposte sui diversi pacchetti IP. Di seguito andiamo ad analizzare i più importanti.

5.1.1 NMAP

Nmap [14] di Fyodor, è un'applicazione gratuita e open source importante e molto utilizzata dalla maggior parte dei professionisti della sicurezza. Con l'utilizzo di Nmap siamo in grado di individuare informazioni importanti come gli sono gli host disponibili in rete, i servizi che questi host offrono, i sistemi operativi che sono in esecuzione, filtri di pacchetti / firewall in uso, e molte altre caratteristiche.

L'output di Nmap varia a seconda delle richieste che si effettuano al servizio e quindi i risultati sono variabili, ma possiamo dire in generale che abbiamo sempre un elenco degli obiettivi scansionati con le varie informazioni a seconda delle opzioni utilizzate.

È la tabella di porta che ci restituisce le informazioni più importanti, come il numero di porta e il protocollo, il nome del servizio e lo stato. Lo stato viene classificato in 4 modi: aperto, filtrato, chiuso o non filtrato. Aperto significa che il servizio sull'host di destinazione sta ascoltando le connessioni/pacchetti su quella porta. Filtrato significa che un firewall, un filtro o un altro ostacolo di rete blocca la porta in modo che Nmap non possa sapere se è aperta o chiusa. Le porte chiuse non hanno alcuna applicazione che li ascolta, anche se potrebbero aprirsi

in qualsiasi momento. Le porte sono classificate come non filtrate quando rispondono alle sonde di Nmap, ma Nmap non può determinare se sono aperte o chiuse.

5.1.2 WIRESHARK

Wireshark [15] un software open source che si utilizza maggiormente per analizzare i pacchetti di rete, praticamente blocca i pacchetti da una rete e cerca di visualizzare il loro contenuto.

Wireshark può essere utilizzato per diversi scopi, come apprendere protocolli di rete, eseguire il debug di nuovi protocolli, analizzare i problemi di rete e individuare problemi di sicurezza.

Analizzare il traffico di una rete permette al penetration tester di riceve importanti informazioni che rivelano vulnerabilità di sicurezza o servono da base per diversi tipi di attacchi.

Wireshark può anche essere utilizzato per analizzare i protocolli utilizzati da diverse macchine in quanto comunicano attraverso la rete al fine di trovare le incongruenze che possono essere sfruttate.

5.1.3 HPING

Hping [16] è uno strumento che si espande sulle funzionalità di base di ping fornendo la possibilità di creare pacchetti IP personalizzati per l'ascolto ei test dei controlli di sicurezza.

Hping viene utilizzato per:

• Port Scanning

Si può effettuare il port scanning basilare con l'aggiunta di opzioni incrementali, come, ad esempio, ++ prima del numero di porta che fa si che si effettui la scansione di molte porte con pacchetti personalizzati e opzioni TCP.

• Access control and Firewall testing

Si possono testare le regole del firewall per garantire la loro integrità.

• Network protocol testing

Può essere utilizzato per creare qualsiasi pacchetto che ci permetta la verifica di come il sistema risponda a comunicazioni malformate.

5.2 Scanning and Vulnerability Assessment Tools

I tools di scanning and vulnerability assessment ci permettono di avere una visione globale della sicurezza presente nel sistema, una valutazione delle reti per determinare le adeguate misure di sicurezza.

Questi strumenti individuano le vulnerabilità presenti e forniscono anche una valutazione di questi difetti a seconda della loro importanza, prima di essere sfruttate da aggressori o da software dannosi. Sono strumenti che hanno al loro interno un database dei difetti, che vengono costantemente aggiornati per essere sempre al passo con le vulnerabilità scoperte giorno dopo giorno.

Sono diversi gli strumenti che svolgono questo ruolo, ma in questa tesi abbiamo preferito mostrare due dei più usati e famosi.

5.2.1 NESSUS

Nessus è uno scanner di vulnerabilità sviluppato da Tenable Network Security [17] . Questo scanner un tempo era open source, ora invece offre due versioni, una gratuita e l'altra a pagamento. La versione a pagamento offre una gamma molto più vasta di plugin rispetto all'altra, anche se quella gratuita è già molto completa.

Nessus è stato sviluppato con architettura client/server. Il server Nessus esegue l'attività di scansione effettiva, mentre il client è l'applicazione front-end del programma.

L'utente di Nessus durante la scansione può anche impostare parametri e variabili quali opzioni di scansione, credenziali, plugin e impostazioni avanzate. Nessus viene utilizzato per rilevare le potenziali vulnerabilitàe la debolezza della rete e dei sistemi come il controllo cracker remoto, le password predefinite, l'attacco DoS, gli aggiornamenti e le patch mancanti, utilizzando sempre il security

vulnerability database che contiene informazioni aggiornate di tutte le vulnerabilità note.

L'utilizzo di Nessus risulta molto facile ed intuitivo. Dopo aver effettuato l'accesso all'interfaccia web, si deve configurare le policy a seconda del nostro target. Migliaia di plugin possono essere utilizzati per trovare le vulnerabilità che forniscono l'intelligenza di valutazione. Fissato poi il target si può subito procedere con la scansione. Finita la scansione il risultato è un elenco di elementi scoperti che possono essere visualizzati per livello di gravità. Nessus classifica le sue vulnerabilità dividendole in critiche, alte, medie e basse. Per ogni vulnerabilità vengono fornite delle spiegazioni dettagliate e la possibilità di scaricare un report, in una vasta gamma di formati, che contiene tutti i dettagli dell'attività svolta.

5.2.2 OPENVAS

OpenVas [18] è uno strumento di scansione delle vulnerabilità rilasciato dalla versione libera di Nessus 2.2, dopo che Nessus non era più open source. Come Nessus, OpenVas esegue la scansione della rete per individuare potenziali vulnerabilità rilasciando un report alla fine. Andiamo a vedere però più dettagliatamente come è strutturato OpenVas.

Possiamo dire che OpenVas include i seguenti moduli:

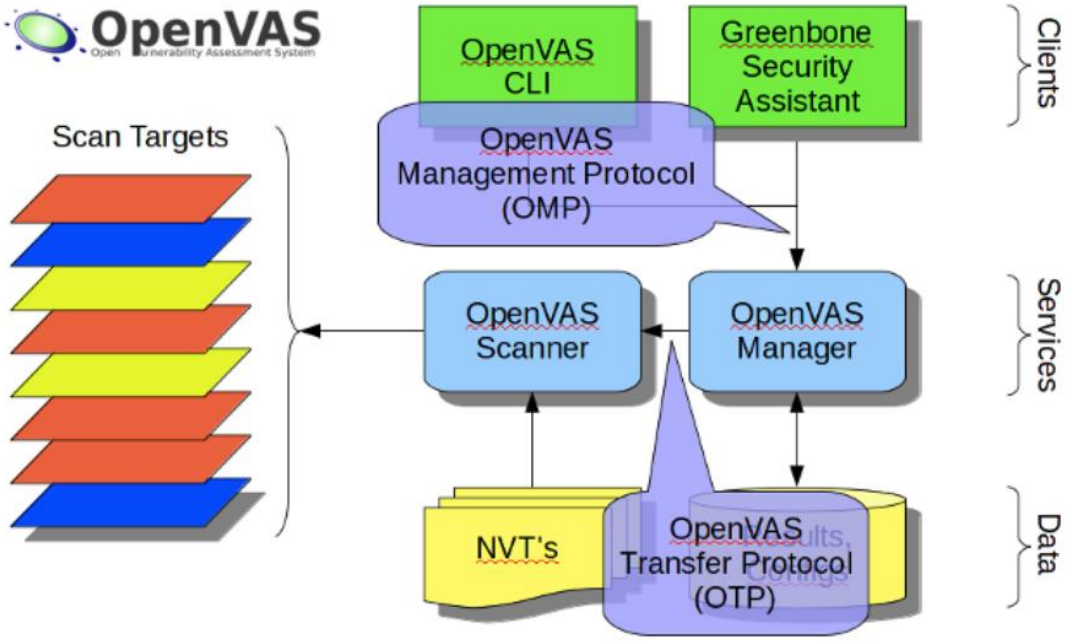

Figura 5.1: OpenVas Modules

- OpenVasManager: Servizio centrale che consolida la scansione delle vulnerabilità
- OpenVas Scanner: esegue i rivelatori di vulnerabilità della rete (NVTs) tramite il feed OpenVAS NVT
- OpenVas Administrator: Strumento di riga di comando che offre il protocollo OpenVAS Administration Protocol (OAP)
- Greenbone Security Assistant (GSA): servizio Web che offre un'interfaccia utente per i browser web
- Greenbone Security Desktop (GSD): client desktop per OpenVAS Management Protocol (OMP)
- Comman Line Interface (CLI): strumento di riga di comando che consente la creazione di processi di sbatch per guidare OpenVAS Manager
- Librerie: Funzionalità condivise aggregate
-

5.3 Penetration testing Tools

Molti penetration tester usano exploit per lo sfruttamento generale, combinati con strumenti come Core Impact, Canvas e Metasploit Framework, oltre alle proprie applicazioni e script.

Sono strumenti che aiutano molto il tester che ovviamente deve aggiungere la propria abilità di pari passo con l'utilizzo di questi framework per ottenere un buon risultato. Non tutte le applicazioni svolgono le medesime funzioni, quindi un bravo pentester dovrebbe anche provarle tutte e scegliere quelle che sono le più adatte per l'ambiente del progetto.

In questa tesi si fa riferimento al solo strumento Metasploit Framework essendo il più completo, e quindi utilizzato, nell'attività di penetration testing.

5.3.1 METASPLOIT

Sviluppato da Rapid7 [19], Metasploit Framework è stato creato inizialmente nel linguaggio di programmazione Perl, ma ultimamente è stato interamente riscritto nel linguaggio di programmazione Ruby.

Metasploit può svolgere molte attività come lo sviluppo dell'exploit, l'inserimento di payloads dannosi per attacchi su lato client ecc… in poche parole tutto ciò che un tester ha bisogno per svolgere la sua attività.

Metasloit Framework usa 5 passaggi principali per arrivare allo sfruttamento di un sistema, che sono:

1. Scegliere e configurare un exploit da mirare
2. Controllare se l'obiettivo è vulnerabile all'uso scelto.
3. Selezionare e configurare il payload che verrà utilizzato.
4. Scegliere e configurare lo schema di codifica per assicurarsi che il payload possa evadere all' Intrusion Detection Systems.
5. Esegui l'exploit.

Metasploit risulta anche molto utile per avere conferma riguardo ai report generati dai tools di vulnerability assessment dimostrando quindi che le vulnerabilità rilevate non sono dei falsi positivi.

Metasploit permette anche di testare i nuovi exploit che vengono dal web, aiutando così il tester a capire bene il loro utilizzo e funzionamento.

Ed infine Metasploit è un ottimo strumento per valutare l'efficacia del sistema di Intrusion Detection System applicando l'exploit e cercando di aggirarlo.

6 Case Study: A small bank of the Marche

6.1 A Bank of the Marche

L'obiettivo principale di questo lavoro di tesi era quello di indagare sugli strumenti e le tecniche utilizzate per il penetration testing, andare quindi ad analizzare le più importanti metodologie esistenti e capire come i tester possono sfruttare queste tecniche al fine di comprendere la sicurezza di un sistema e proteggerlo da eventuali attacchi.

A fronte di questi studi, viene qui mostrato il penetration testing effettuato per una banca delle Marche.

Questo test alla banca è stata un'attività suddivisa in più giornate e sono state effettuate diverse fasi per arrivare a raggiungere l'obiettivo interessato. È stata seguita la struttura generale che il penetratrion testing richiede, ovvero si è iniziata l'attività con la fase di pre-attacco, continuata con la fase di attacco e conclusa con la fase di post-attacco.

Andremo nei capitoli successivi a vedere nel dettaglio quali tecniche sono state utilizzati e quali informazioni sono state riscontrate.

6.1.1 Information gathering

Dopo aver discusso tutte le regole di ingaggio con la nostra azienda di riferimento, aver quindi compreso bene qual'è il nostro scopo, aver pianificato la nostra attività e capito quali sono le nostre restrizioni legali passiamo alla fase di information gathering.

Innanzitutto dobbiamo precisare che abbiamo iniziato la nostra attività con la tipologia black-box, in quanto abbiamo preferito essere all'oscuro di tutti i dettagli che riguardavo il sistema, proprio ipotizzando la stessa situazione in cui un attaccante si trova.

È stata quindi molto importante per noi la fase di information gathering in quanto è proprio in questa fase che abbiamo iniziato a capire la situazione che avevamo di fronte.

È stato essenziale qui comprendere il tipo e la quantità di informazioni necessarie per iniziare il test.

L'information gathering è stato effettuando sia raccogliendo informazioni in modo passivo, ovvero facendo riferimento al web e cercando tutte le informazioni che apertamente sono disponibili; sia in maniera attiva con l'utilizzo di tool che ci permetto di ricavare maggiori dettagli.

Con l'utilizzo di tool come Nmap, o Whois ed altri, siamo riusciti a ricavare informazioni riguardo le porte, i sistemi operativi presenti nei sistemi e quindi a riscostruire una visione completa di tutta la rete interessata. Lo strumento ampiamente utilizzato è stato Nmap perchè ha fornito molti dettagli nella definizione degli obiettivi.

6.1.1.1 Network surveying results

Il primo passo è stato quello di identificare gli hosts attivi nel segmento di rete. Con l'utilizzo di Nmap il comando più immediato da utilizzare ci permette di inviare pacchetti ICMP e vedere quali host rispondono a questi pacchetti.

```
Starting Nmap 7.30 ( https://nmap.org ) at 2017-06-09 10:06 ora legale Europa occidenta
NSE: Loaded 142 scripts for scanning.
NSE: Script Pre-scanning.
Initiating NSE at 10:06
Completed NSE at 10:06, 0.00s elapsed
Initiating NSE at 10:06
Completed NSE at 10:06, 0.00s elapsed
Initiating ARP Ping Scan at 10:06
Scanning 63 hosts [1 port/host]
Completed ARP Ping Scan at 10:06, 1.04s elapsed (63 total hosts)
Nmap scan report for 10.133.37.64 [host down]
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try us
Nmap scan report for 10.133.37.69 [host down]
```

Figure 6.1: Start scanning Nmap

```
Discovered open port 41964/tcp on 10.133.37.75
Discovered open port 41964/tcp on 10.133.37.70
Discovered open port 41964/tcp on 10.133.37.73
Discovered open port 49181/tcp on 10.133.37.73
Discovered open port 50001/tcp on 10.133.37.68
Discovered open port 50002/tcp on 10.133.37.68
Discovered open port 49155/tcp on 10.133.37.70
Discovered open port 4141/tcp on 10.133.37.67
Discovered open port 49152/tcp on 10.133.37.75
Discovered open port 49152/tcp on 10.133.37.70
Discovered open port 49152/tcp on 10.133.37.73
SYN Stealth Scan Timing: About 70.25% done; ETC: 10:09 (0:00:54 remaining)
Discovered open port 49154/tcp on 10.133.37.75
```

Figure 6.2: Active Nmap hosts

Da questa immagine possiamo vedere che iniziata la scansione, sono 63 gli host da analizzare e individuare quali sono aperti e quali no.

In alcuni scenari, a volte questo semplice comando non ci permette di avere informazioni perchè molte organizzazioni e società filtrano questi pacchetti ICMP sia dai loro host che dalla rete. Il port scanning viene quindi effettuato con l'utilizzo di tecniche e protocolli diversi, come TCP o UDP.

Queste ultime scansioni risultano essere molto più dettagliate, fornendo preziose informazioni sugli hosts e sui servizi.

6.1.1.2 Network scanning results

Una volta identificati gli host attivi e determinati i loro indirizzi IP è stata effettuata la scansione delle porte individuando il loro stato, i servizi che girano su di esse e i sistemi operativi.

Sono state eseguite sia tecniche TCP che UDP per la scansione delle porte e in questo particolare caso possiamo vedere che viene usato il SYN Stealth Scan che l'ACK Scan.

Una scansione SYN distingue quali porte erano in ascolto o non basandosi sulla risposta generata.

Nmap Scan Report - Scanned at Fri Feb 24 15:13:02 2017

Scan Summary

Nmap 7.30 was initiated at Fri Feb 24 15:13:02 2017 with these arguments:
nmap -A 10.133.36.110

Verbosity: 0; Debug level 0

10.133.36.110(online)

Address

- 10.133.36.110 - (ipv4)
- 00:1D:7D:47:9A:1C - Giga-byte Technology (mac)

Ports

The 980 ports scanned but not shown below are in state: **closed**

Port	State	Service	Reason	Product	Version	Extra info
53	tcp open	domain	syn-ack	Microsoft DNS		
88	tcp open	kerberos-sec	syn-ack	Microsoft Windows Kerberos		server time: 2017-02-24 14:13:13Z
135	tcp open	msrpc	syn-ack	Microsoft Windows RPC		
139	tcp open	netbios-ssn	syn-ack	Microsoft Windows netbios-ssn		
389	tcp open	ldap	syn-ack	Microsoft Windows Active Directory LDAP		Domain: tdata.servizi, Site: [redacted]
445	tcp open	microsoft-ds	syn-ack	Windows Server 2012 R2 Standard 9600 microsoft-ds		workgroup: [redacted]
464	tcp open	kpasswd5	syn-ack			
593	tcp open	ncacn_http	syn-ack	Microsoft Windows RPC over HTTP	1.0	
636	tcp open	ldap	syn-ack	Microsoft Windows Active Directory LDAP		Domain: tdata.servizi, Site: [redacted]
3268	tcp open	ldap	syn-ack	Microsoft Windows Active Directory LDAP		Domain: tdata.servizi, Site: [redacted]
3269	tcp open	ldap	syn-ack	Microsoft Windows Active Directory LDAP		Domain: tdata.servizi, Site: [redacted]
3389	tcp open	ms-wbt-server	syn-ack	Microsoft Terminal Service		
5666	tcp open	tcpwrapped	syn-ack			
49152	tcp open	msrpc	syn-ack	Microsoft Windows RPC		
49153	tcp open	msrpc	syn-ack	Microsoft Windows RPC		
49154	tcp open	msrpc	syn-ack	Microsoft Windows RPC		
49156	tcp open	msrpc	syn-ack	Microsoft Windows RPC		
49157	tcp open	ncacn_http	syn-ack	Microsoft Windows RPC over HTTP	1.0	
49158	tcp open	msrpc	syn-ack	Microsoft Windows RPC		
49160	tcp open	msrpc	syn-ack	Microsoft Windows RPC		

Remote Operating System Detection

- Used port: **53/tcp (open)**
- Used port: **1/tcp (closed)**
- Used port: **36832/udp (closed)**
- OS match: **Microsoft Windows Server 2012 R2 Update 1 (100%)**
- OS match: **Microsoft Windows 7, Windows Server 2012, or Windows 8.1 Update 1 (100%)**

Traceroute Information

- Traceroute data generated using port /

Hop	Rtt	IP	Host
1	0.12	10.133.36.110	

Misc Metrics

Metric	Value
Ping Results	
System Uptime	24455217 seconds (last reboot: Tue May 17 15:07:49 2016)
TCP Sequence Prediction	Difficulty=262 (Good luck!)
IP ID Sequence Generation	Incremental

Figure 6.3: Nmap host scan 10.133.36.110

La Figura 6.3 mostra l'output di una scansione ACK contro gli host della rete di destinazione utilizzando Nmap, in particolare si fa riferimento ad un singolo host preso come esempio.

In questa immagine possiamo vedere che siamo a conoscenza delle prime informazioni importanti che ci serviranno per procedere al passo successivo.

Possiamo vedere che siamo in grado di identificare quali porte sono aperte,chiuse o filtrate nel dispositivo in esame.

- Quale servizio interessa la porta corrispondente
- Quali sistemi operativi sono in esecuzione sul nostro obiettivo
- Questa specifica attività è stata effettuata per tutti gli host che sono risultati attivi nella rete.

L'information gathering ha fornito le basi per la prossima fase di scanning and vulnerability assessment. Sono stati individuati gli host attivi e ulteriormente analizzati per trovare porte aperte e servizi in esecuzione su tali porte.

Nmap è stato utilizzato principalmente per il port scanning, l'enumerazione di servizi e dei sistemi operativi. Quindi Nmap è stato uno degli strumenti versatili più utilizzati durante l'information gathering.

6.1.2 Scanning and Vulnerability Assessment

In questa fase tutte le informazioni precedentemente raccolte sono state sfruttate per completare la tecnica di valutazione e scansione delle vulnerabilità.

Di solito, in questa fase, si fa uso sia della tecnica manuale che automatizzata per la ricerca delle vulnerabilità. Si deve prendere in considerazione però che la tecnica manuale richiede molto più tempo per perfezionare la scansione e soprattutto risulta molto più lunga e poco efficiente se il nostro obiettivo fosse una rete con centinaia di sistemi.

Abbiamo infatti preferito utilizzare scanner automatizzati di vulnerability assessment come Nessus e OpenVas. Questi scanner ci hanno permesso di avere una controprova sulle informazioni precedentemente rilevate, come sistemi operativi e servizi in esecuzione negli hosts.

Questi due tools ci hanno anche permesso di rilevare quali vulnerabilità erano presenti nei nostri hosts attivi, vulnerabilità che verranno poi esaminate per verificare quali sono i possibili exploit da mandare nel nostro obiettivo nella fase di exploitation.

6.1.2.1 Vulnerability Assessment Nessus results

Nessus Home Feed edition è stata utilizzata per valutare le vulnerabilità degli hosts attivi nel nostro sistema. Tutti i plug-in sono stati installati e aggiornati prima della scansione.

10.133.36.110

Summary

Critical	High	Medium	Low	Info	Total
1	0	12	2	40	55

Details

Severity	Plugin Id	Name
Critical (10.0)	79638	MS14-066: Vulnerability in Schannel Could Allow Remote Code Execution (2992611) (uncredentialed check)
Medium (6.8)	90510	MS16-047: Security Update for SAM and LSAD Remote Protocols (3148527) (Badlock) (uncredentialed check)
Medium (6.4)	51192	SSL Certificate Cannot Be Trusted
Medium (6.4)	57582	SSL Self-Signed Certificate
Medium (5.1)	18405	Microsoft Windows Remote Desktop Protocol Server Man-in-the-Middle Weakness
Medium (5.0)	12217	DNS Server Cache Snooping Remote Information Disclosure
Medium (5.0)	20007	SSL Version 2 and 3 Protocol Detection
Medium (5.0)	42873	SSL Medium Strength Cipher Suites Supported
Medium (5.0)	94437	SSL 64-bit Block Size Cipher Suites Supported (SWEET32)
Medium (4.3)	57690	Terminal Services Encryption Level is Medium or Low
Medium (4.3)	58453	Terminal Services Doesn't Use Network Level Authentication (NLA) Only
Medium (4.3)	78479	SSLv3 Padding Oracle On Downgraded Legacy Encryption Vulnerability (POODLE)
Medium (4.0)	35291	SSL Certificate Signed Using Weak Hashing Algorithm
Low (2.6)	30218	Terminal Services Encryption Level is not FIPS-140 Compliant
Low (2.6)	65821	SSL RC4 Cipher Suites Supported (Bar Mitzvah)

Figure 6.4: Nessus host scan 10.133.36.110

Nell'esempio preso in considerazione possiamo vedere che vengono individuate 1 vulnerabilità critica, 12 vulnerabilità medie e 2 vulnerabilità basse.

Nella notazione di Nessus, ogni vulnerabilità è stata associata a un fattore di rischio o una gravità. Le vulnerabilità identificate sono state contrassegnate come critiche, alte, medie, basse. Queste informazioni sono state date in base al punteggio di una vulnerabilità (CVSS). CVSS is a vulnerability scoring system

designed to provide an open and standardized method for rating IT vulnerabilities. Questi fattori di rischio non sono classificati in tutti i tool nello stesso modo, quindi una gravità etichettata come "Critica" su Nessus potrebbe non avere lo stesso livello di severità usando altri scanner. Pertanto, i fattori di rischio dovrebbero essere considerati come linee guida o suggerimenti.

6.1.2.2 Vulnerability Assessment OpenVas results

Anche OpenVas, esattamente come Nessus, è stato utilizzato per valutare le vulnerabilità presenti negli hosts individuati. È stato infatti scelto di considerare entrambi i tools di vulnerability assessment in modo da poter meglio paragonare ed analizzare i risultati.

Host	High	Medium	Low	Log	False Positive
10.133.36.110 [redacted]	0	16	1	39	0
Total: 1	0	16	1	39	0

Figure 6.5: OpenVas host scan 10.133.36.110

Prendendo quindi come esempio lo stesso host attivo precedentemente visto con Nessus, possiamo vedere che in questo caso vengono riscontrate 16 vulnerabilità medie ed 1 vulnerabilità bassa.

Esattamente come i report di Nessus, i report di OpenVas includono una panoramica degli host interessati, una breve descrizione che fornisce la conoscenza della vulnerabilità, il livello di impatto che una determinata vulnerabilità può avere nella sua applicazione o sistema e mostrate le oppurtinità per risolvere o mitigare la vulnerabilità.

Anche in questo caso le vulnerabilità sono classificate a seconda di quanto può essere minacciosa per l’obiettivo interessato. Le vulnerabilità identificate sono state contrassegnate come High, Medium, Low, Log e False Positive in base al punteggio base del sistema di punteggio di vulnerabilità (CVSS).

I report di Nessus e OpenVas indicavano che gli host sulla rete erano vulnerabili all'esecuzione di codice in modalità remota, al buffer overflow, all'elevazione del privilegio, ma per riuscire a classificare veramente l’importanza di una vulnerabilità bisogna provare a sfruttarla.

È proprio nella fase successiva, infatti, che siamo in grado di capire se una vulnerabilità risulta essere pericolosa o no per per il sistema. Se una vulnerabilità, nella fase di attacco, ci permette di prendere il controllo del dispositivo allora possiamo classificarla come pericolosa, dal momento che è grazie a lei che siamo riusciti ad ottenere il controllo della macchina.

I tools Nessus e OpenVas ci hanno fornito una buona base per verificare la sicurezza locale dei sistemi e delle infrastrutture.

Diciamo che sarebbe molto utile avere questi due tools di ricerca delle vulnerabilità all'interno di ogni infrastruttura, sarebbe un grande passo che un'azienda dovrebbe fare per cercare sempre di non avere grandi problematiche nell'ambito della sicurezza. Tali scanner sono, infatti, un elemento di sicurezza IT se configurati correttamente e senza problemi. Nessus e OpenVAS sono stati i due scanner selezionati per questa tesi, ma altri scanner come Nexpose, Retina o Internet Security Systems possono anche essere utilizzati durante la fase di valutazione e scansione delle vulnerabilità.

6.1.2.3 Comparison of results

Andando quindi ad analizzare i CVE che sono stati riscontrati in entrambi i risultati vediamo fin da subito delle differenze. OpneVas risulta più dettagliato e specifico nella ricerca delle vulnerabilità rispetto a Nessus, ma Nessus ha un database di plug-in molto più grande rispetto ad OpenVas.

Infatti possiamo notare che la vulnerabilità classifica come critica in Nessus non viene proprio identificata in OpenVas.

Il consiglio è di usare entrambi i tools per analizzare e identificare al meglio le vulnerabilità presenti nella rete e nei sistemi.

6.1.2.3.1 Important Discovery

Avendo svolto la nostra attività in più giornate, abbiamo riscontrato noi stessi l'importanza del fatto che il penetration testing è un'attività ciclica, che deve ripetersi in un periodo di tempo non troppo lungo per far sì che il proprio sistema risulti sicuro da eventuali intrusioni.

Abbiamo appena parlato delle vulnerabilità riscontrate nell'obiettivo preso come esempio e questa analisi della vulnerabilità per quel determinato obiettivo è stata svolta nelle prime due giornate di test.

Nei giorni che hanno preceduto l'ultima giornata di test, gli esperti del sistema sono venuti a conoscenza di una vulnerabilità appena scoperta, la vulnerabilità chiamata MS17-010.

6.1.2.3.2 MS17-010 Wanna Cry

Wanna Cry [20] è il ransomware che ha bloccato l'accesso ai file su migliaia di computer in quasi 100 paesi, scoperto il 14/03/2017.

I maggiori interessati sono stati gli ospedali del Regno Unito, un operatore telefonico in Spagna, FedEx negli Stati Uniti, Deutsche Bahn in Germania e l'università Bicocca in Italia.

WannaCry cerca e crittografa 176 tipi diversi di file e aggiunge l'estensione .WCRY alla fine del nome dei file. Chiede agli utenti di pagare un riscatto di 300 dollari in bitcoin. La richiesta di riscatto indica che l'importo del pagamento verrà raddoppiato dopo tre giorni. Se il pagamento non viene effettuato entro sette giorni, i file crittografati verranno eliminati.

Figura 6.6: WannaCry Ransomware [21]

Individuata quindi questa vulnerabilità nella scansione degli ultimi dispositivi che dovevamo analizzare, ci siamo preoccupati di ripetere l'operazione per tutti gli hosts attivi.

Il risultato: gli host che precedentemente non risultavano vulnerabili a questo ransomware, perchè non ancora scoperto, ora risultano invece tutti affetti da questa vulnerabilità.

Anche il nostro dispostivo preso fin ora come esempio dimostra ciò che stiamo affermando.

```
msf > use auxiliary/scanner/smb/smb_ms17_010
msf auxiliary(smb_ms17_010) > set RHOSTS 10.133.36.1/25
RHOSTS => 10.133.36.1/25
msf auxiliary(smb_ms17_010) > set SMBDomain .
SMBDomain => .
msf auxiliary(smb_ms17_010) > set THREADS 24
THREADS => 24
msf auxiliary(smb_ms17_010) > set RPORT 445
RPORT => 445
msf auxiliary(smb_ms17_010) > run -j
[*] Auxiliary module running as background job
[*] 10.133.36.18:445      - Connected to \\10.133.36.18\IPC$ with TID = 2048
[*] 10.133.36.18:445      - Received STATUS_INSUFF_SERVER_RESOURCES with FID = 0
[!] 10.133.36.18:445      - Host is likely VULNERABLE to MS17-010!

[*] 10.133.36.110:445     - Connected to \\10.133.36.110\IPC$ with TID = 18437
[*] 10.133.36.110:445     - Received STATUS_INSUFF_SERVER_RESOURCES with FID = 0
[!] 10.133.36.110:445     - Host is likely VULNERABLE to MS17-010!
```

Figura 6.7: Host vulnerable to MS17-010

6.1.3 Exploitation

È in questa fase che vengono analizzate e riscontrate le vulnerabilità scoperte con Nessus e OpenVas. Viene quindi specificate se le lacune e le minaccie individuate nella fase di scansione rappresentano effetivamente reali problematiche per la sicurezza del sistema in esame.

Durante questa fase, le vulnerabilità sono state sfruttate utilizzando gli exploit pubblicamente disponibili. Metasploit è stato uno di questi framework di exploitation open source, che è risultato molto utile in questa fase, permettendoci di prendere possesso della macchina obiettivo.

6.1.3.1 Social Engineering Method

Come abbiamo già detto, le vulnerabilità di ingegneria sociale a volte risultano essere le più pericolose. Non bisogna mai fidarsi della persone che entrano nella nostra azienda per la prima volta, non bisogna dare a loro il permesso di accedere alla rete che è riconducibile ad informazioni importanti, e nemmeno dare la possibilità di inserire supporti rimovibili nei computer aziendali.

È stato proprio così che siamo riusciti ad intrufolarci in un pc dell'azienda, sfruttando la cortesia di un loro dipendente.

Con l'utilizzo di un apposito tool, abbiamo costruito un trojan.

Un trojan è un tipo di malware, è un programma maligno mascherato da qualcosa di benigno. Un trojan viene utilizzato per entrare nel computer della vittima senza farsi riconoscere, permette di accedere ai dati dello stesso.

A volte si tende ad inviare un trojan tramite allegato mail tentando di mascherarlo in un'altra forma che risulti non malevola per l'utente.

I trojan hanno lo stesso livello di privilegi dell'utente che lo avvia; quindi se sarà un utente normale, senza troppi privilegi, esso potrebbe causare pochi danni, ma se invece l'utente è amministratore esso può danneggiare notevolmente il dispositivo.

Abbiamo quindi immesso un trojan in un file eseguibile che abbiamo poi inserito in un dispositivo rimovibile, ed in maniera fraudolenta abbiamo consegnato questo dispotivo ad un dipendente, che inserendolo nel proprio computer ha fatto si che il sistema diventasse infetto dal nostro trojan.

Avevamo in quel momento quindi le stesse autorizzazioni che aveva quell'utente e siamo riusciti ad ottenere il controllo della macchina.

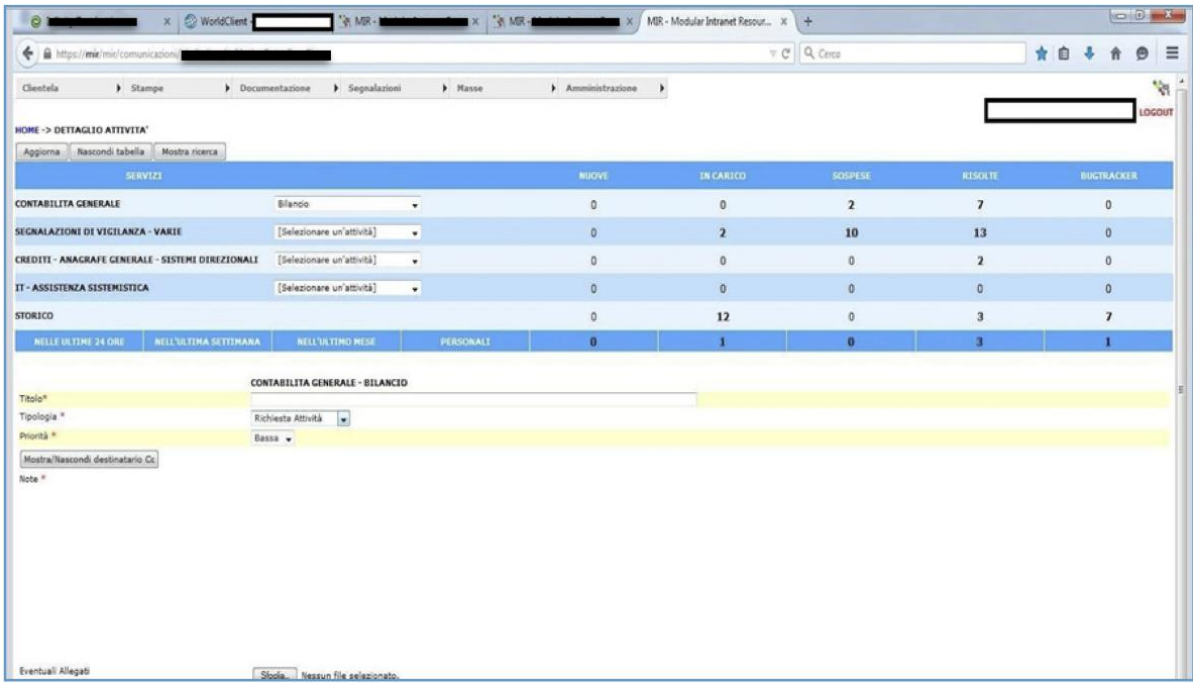

Figure 6.8: Accessing the device

In quel momento eravamo in grado di svolgere qualsiasi azione all'interno della macchina, come se fossimo noi in prima persona davanti al monitor. È stato possibile vedere in tempo reale tutte le operazioni che si svolgevano con il computer, ed infatti l'immagine sopra è appunto uno screenshot del monitor.

6.1.3.2 MS17-010 Method

Sul dispositivo utilizzato come esempio abbiamo rilevato questa vulnerabilità, e con l'utilizzo di Metasploit verificato se effettivamente questa vulnerabilità risultava dannosa per l'intero sistema.

L'exploit è stato eseguito utilizzando i seguenti passaggi:

1. **Lunching the Metasploit Framework**

   Msfconsole è stato il comando utilizzato per lanciare il framework di metasploit. Msfconsole è stato utilizzato per lanciare exploit, caricare i moduli ausiliari e cercare exploit.

2. **Searching for SMBv2 exploit**

   Il comando di ricerca è stato utilizzato per cercare l'exploit.

   Nessus ha indicato la vulnerabilità MS17-010, quindi la parola chiave 'ms17-010' è stata usata come parametro di ricerca. Abbiamo quindi selezionato il modulo più opportuno e viene poi utilizzato per eseguire l'exploit.

3. **Loading the exploit module**

   Viene utilizzando il comando *use* per caricare il modulo specifico per exploit e *show options* per elencare le opzioni del modulo.

4. **Setting required Options and Payload to compromise the host**

   Viene poi impostato l'indizzo ip della macchina da attaccare (RHOST) e quello della macchina che sta attaccando (LHOST).

   È stato selezionato un payload chiamato "Meterpreter reverse_tcp" che si collegava all'istanza Metasploit. Meterpreter è un tool che ci ha aiutato a ricavare più informazioni possibili, una volta che il sistema era già stato compromesso.

5. **Activation of the exploit**

Viene utilizzato il comando *exploit* per eseguire l'effettivo sfruttamento. Una volta che l'operazione è andata a buon fine eravamo in grado di muoverci all'interno della macchina, di andare così ad esplorare i file che erano all'interno, di effettuare gli screenshot del monitor e molte altre operazioni che risulterebbero molto dannose per un'azienda.

6.1.4 Post Exploitation Phase

Nel penetration testing i privilegi del livello di root non sono sempre raggiunti durante la fase di exploitation.

È proprio in questa fase, infatti, che abbiamo tentato di raggiungere i massimi privilegi all'interno di una macchina.

Se è stato ottenuto l'accesso utente, il tester dovrebbe eseguire diverse attività per poter raggiungere il livello root. Se è stato ottenuto l'accesso alla rete, il tester può sniffare il traffico di rete e ottenere informazioni sensibili.

6.1.5 Reporting Phase

Dopo aver completato tutte le fasi abbiamo scritto un report che contiene tutti i risultati dettagliati ottenuti nelle diverse fasi insieme ad alcune raccomandazioni da effettuare per migliorare il sistema.

In questa fase abbiamo anche effettuato la cancellazione di tutti gli artefatti che abbiamo precedentemente utilizzato per compromettere il nostro sistema.

Questa fase serve maggiormente come punto di riferimento per effettuare le migliorie al sistema in esame, soprattutto se si ha davanti un report ben scritto.

Il report che abbiamo effettuato contiente:

- Executive Summary

  Viene mostrato un breve scenario dell'attività che è stata svolta. In questa parte viene mostrato lo scopo del test e i risultati più importanti.

- Approach

  In questa fase viene descritto l'approccio utilizzato per eseguire questa attività

- Scope

  Spiegato l’ambito del test

- Finding

  Questa parte mostra il punto centrale del lavoro svolto. Qui vengono mostrati tutti i risultati individuati durante il test, quindi le vulnerabilità riscontrate e come esse sono state analizzate e sfruttate.

- Recommendations

  Vengono specificate le raccomandazioni e le modifiche da apportare per mitigare le vulnerabilità presenti, anche in base al loro rischio.

7 Conclusions

Un penetration testing risulta essere efficace, se innanzitutto è efficace la metologia che viene utilizzata per eseguirlo.

Uno degli obiettivi fissati per questa tesi è stato quello di andare ad analizzare e confrontare le principali metodologie che vengono seguite per affrontare il penetration testing. Abbiamo infatti visto come ognuna di esse sviluppa tecniche e approcci differenti nel relazionarsi a questa attività.

Abbiamo mostrato nei dettagli tutti i passi che le metologie OSSTMM, OWASP e ISSAF consigliano di intraprendere per effettuare un buon penetration testing e spiegati poi i tools più importanti che vengono utilizzati per questa attività.

La sezione della tesi che si occupa dell'analisi di questi tools (Nmap, Nessus, OpenVas e Metaspolit Framework), viene ben definita a seconda del loro utilizzo. Viene infatti affrontata la fase di information gathering, che consiste nella visione globale del sistema in considerazione. Conoscere quindi più informazioni possibili riguardo agli host attivi nell'ambiente, il loro sistema operativo, e i servizi che girano su di esso.

Si passa poi alla fase di scoperta ed analisi delle vulnerabilità, facendo riferimento ai tools Nessus e OpenVas. Vengono sfruttati questi due tools per ricevere informazioni dettagliate riguardo le vulnerabilità esistenti nell'ambiente. Ci permettono, infatti, di classificare le vulnerabilità riscontrate a seconda del rischio che esse comportano e ricevere dettagli anche su come mitigarle.

Metasploit framework ha fatto si che potessimo avere un riscontro riguardo le vulnerabilità scoperte. Abbiamo utilizzato questo framework per indivuduare i giusti exploit che ci hanno aiutato ad entrare nella macchina obiettivo.

Non dobbiamo tralasciare il dettaglio che lo strumento più importante da utilizzare è la capacità e l'esperienza del tester, colui che organizza bene l'uso di questi tools.

Questi strumenti e la giusta metodologia, se ben utilizzati, ci permettono di individuare quali sono le debolezze presenti nel sistema o nella rete in esame, per poi essere sfruttati.

Il penetration testing risulta essere molto importante per la valutazione della sicurezza, ma non è un'alternativa alle misure di sicurezza. Dovrebbe essere utilizzato per completare la difesa di un ambiente che già ha adottato delle norme di sicurezza.

7.1 Future Developments

Questo lavoro di tesi può essere esteso per diversi sviluppi futuri:

- Facendo riferimento alle metodologie affrontate si può pensare di effettuare un'ulteriore metodologia. Questa metodologia potrebbe essere sviluppata prendendo in considerazione i pregi delle altre metodologie e provare a raggrupparli. Facendo così dovrebbe risultare una metodologia che sia dettagliata, ma facile da capire, che abbia delle buone linee guida in tutte le fasi del penetration testing e che aiuti anche nella stesura del report. In questo modo qualsiasi azienda potrebbe studiare questo approccio e tentare di adottarlo.
- Questo lavoro di tesi può essere sviluppato prendendo anche più in considerazione il fattore umano, e quindi di conseguenza l'ingegneria sociale. Verranno così scoperte tutte le vulnerabilità, sia di natura informatica che umana. Abbiamo infatti avuto un chiaro esempio nel nostro caso studio di come è molto importante avere all'interno della proprio ambiente personale fidato e attento agli inganni.
- Si può riprendere gli stessi passi di questa tesi ed andare però a confrontare metodologie diverse e tools differenti. Così si potrebbe avere una visione ancora più ampia di come affrontare un buon penetration testing. Se viene effettuato questo lavoro in futuro, si potrebbe poi analizzare questa tesi e quel lavoro per effettuare un ulteriore confronto del tutto.

8 Bibliography and Sitography

[1]“The Open Source Security Testing Methodology Manual”, ISECOM http://www.isecom.org/mirror/OSSTMM.3.pdf

[2]“Owasp testing guide”, OWASP Foundation 2008 v3.0 https://www.owasp.org/images/5/56/OWASP_Testing_Guide_v3.pdf

[3]“Open Information Systems Security Group”. Information systems security assessment framework, 2009. https://ht.transparencytoolkit.org/FileServer/FileServer/whitepapers/issaf/issaf0.2.1A.pdf

[4]“Rapporto CLUSIT sulla Sicurezza ICT in Italia”, CLUSIT,2016 https://clusit.it/wp-content/uploads/download/Rapporto_Clusit%202016.pdf

[5] https://www.ashleymadison.com/app/public/loginaction.p

[6]“Multi-vendor Penetration Testing in the Advanced Metering Infrastructure”, Stephen McLaughlin, Dmitry Podkuiko, Sergei Miadzvezhanka, Adam Delozier, and Patrick McDaniel https://ai2-s2-pdfs.s3.amazonaws.com/1b73/b9e81d80e91ba4e5bef4ae06805cd269dc04.pdf

[7] http://www.securityfocus.com/

[8] https://www.exploit-db.com/

[9] https://packetstormsecurity.com/

[10] “OWASP Top 10 Application Security Risks – 2017”, OWASP Foundation https://www.owasp.org/index.php/Top_10_2017-Top_10

[11]https://www.certnazionale.it/news/2017/07/12/aggiornamenti-di-sicurezza-per-prodotti-microsoft-luglio-2017/

[12] http://www.networkworld.com/article/2316231/lan-wan/is-penetration-testing-more-effective-than-vulnerability-scanning--yes.html

[13] https://www.sans.edu/cyber-research/security-laboratory/article/ron-gula-interview

[14] ”nmap network scanning”, G. F. Lyon. https://nmap.org

[15] The Wireshark Team. Wireshark (version 1.8.5) (software). January 2013. http://www.wireshark.org/download.html

[16] http://hping.org/

[17] https://www.tenable.com/products/nessus-vulnerability-scanner

[18] “About OpenVAS Software”, http://www.openvas.org/software.html

[19] https://www.metasploit.com/

[20] “Diffusione ransomware WannaCry mondiale”, http://giubin.com/diffusione-ransomware-wannacry-mondiale-cose-e-come-difendersi/

[21] https://www.bing.com/images/search?view=detailV2&ccid=wCZfBwbH&id=FE6E068F78066F1A8D3139AEFE6B24056B4C980E&thid=OIP.wCZfBwbHBn5KItwrx_FiZQEgDY&q=wannacry&simid=608025022628695005&selectedindex=3&mode=overlay&first=1

[21] https://www.microsoft.com/it-it/store/b/home?tduid=(bf6051f47eedf415e8320e4a0a06ed87)(213952)(2856052)(3-3494-__-3-3494-__-K2pqOXRscmxmdH0=)(3_c_90584802_628396260_kwd-33487154245)

[22] https://www.ibm.com/it-it/

[23] https://www.verifyit.nl/wp/wp-content/uploads/2016/02/IssafMethodology-1.png

"Vulnerability Assessment for Disaster Risk Management", ADJIE PAMUNGKAS, ST., M.Dev.Plg https://researchbank.rmit.edu.au/eserv/rmit:160494/Pamungkas.pdf

"Comparative Study of Penetration Testing Methods", Yong-Suk Kang, Hee-Hoon Cho, Yongtae Shin and Jong-Bae Kim http://onlinepresent.org/proceedings/vol87_2015/8.pdf

"Penetration testing methodologies" https://www.owasp.org/index.php/Penetration_testing_methodologies

"SQL Injection: cosa sono e come sfruttarle" http://www.html.it/pag/19638/sql-injection-cosa-sono-e-come-sfruttarle/

"Penetration Studies" https://www.cs.clemson.edu/course/cpsc420/material/Security%20Practice/Vulnerability%20Analysis/Penetration%20Studies.pdf

"Modeling security threats", B. Schneier https://www.schneier.com/academic/archives/1999/12/attack_trees.html

"NPT Network Penetration Test", ISGroup http://www.isgroup.it/downloads/products/ISGroup_Network-Penetration-Test_NPT_IT.pdf

"A Design Methodology for Computer Security Testing", Marco Ramilli http://amsdottorato.unibo.it/4438/4/Marco_Ramilli_Dissertation.pdf

C Jackson. "Network Security Auditing". C Jackson. 2010, https://www.kahkeshan.com/Source/introduceBook/20872299-26c0-4952-988e-d7cdc6a6b02e

"Technical Guide to Information Security Testing and Assessment", K. Scarfone, M. Souppaya, A. Cody, and A. Orebaugh. http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-115.pdf

"Diffusione ransomware WannaCry mondiale", http://giubin.com/diffusione-ransomware-wannacry-mondiale-cose-e-come-difendersi/

http://www.scriptalert1.com/it.html

“Informazioni sul ransomware WannaCry”, Symantec Corporation
https://support.symantec.com/it_IT/article.ALERT2341.html

“Cosa è e come funziona un Trojan”, https://hacktips.it/cosa-come-funziona-un-trojan/

“Microsoft Security Bulletin MS17-010 – Critical”, Microsoft
https://technet.microsoft.com/en-us/library/security/ms17-010.aspx

“A Study of Penetration Testing Tools and Approaches”, CHIEM TRIEU PHONG
http://aut.researchgateway.ac.nz/bitstream/handle/10292/7801/ChiemTP.pdf?sequence=3

“Exploiting Network Printers”, Müller, Jens
https://www.nds.rub.de/media/ei/arbeiten/2017/01/30/exploiting-printers.pdf

“ANEX: AUTOMATED NETWORK EXPLOITATION THROUGH PENETRATION TESTING”, Eric Francis Dazet
http://digitalcommons.calpoly.edu/cgi/viewcontent.cgi?article=2724&context=theses

“Design and implementation of a non-aggressive automated penetration testing tool”, Viggiani https://people.kth.se/~maguire/DEGREE-PROJECT-REPORTS/130529-Fabio_Viggiani-with-cover.pdf

“Security Assessment via Penetration Testing: A Network and System Administrator’s Approach”, Nishant Shrestha
https://www.duo.uio.no/bitstream/handle/10852/34904/Shrestha-masterthesis.pdf

“Metodologie e tecniche per la sicurezza delle reti aziendali”, Lorenzi Stefano
http://www.stefanolorenzi.org/site/File/Tesi2.pdf

“Constructing Plan Trees for Simulated Penetration Testing”, Dorin Shmaryahu
http://icaps16.icaps-conference.org/proceedings/dc/abstracts/shmaryahu.pdf

“The Use of Vulnerability Assessments: a Survey”, Charles D. Lybrand
http://epublications.regis.edu/cgi/viewcontent.cgi?article=1223&context=theses

"Penetration Testing – Method", tutorialspoint
https://www.tutorialspoint.com/penetration_testing/penetration_testing_method.htm

"Network penetration testing guide", TechTarget
http://searchnetworking.techtarget.com/tutorial/Network-penetration-testing-guide

"Penetration Testing ed Ethical Hacking", Gluelabs https://glue-labs.com/articoli/penetration-testing-ed-ethical-hacking

"ABC della sicurezza: Penetration testing", TechEconomy
http://www.techeconomy.it/2015/07/21/abc-sicurezza-penetration-testing/

"Penetration Test", Yarix S.r.l. https://www.yarix.com/it/penetration-test.php

"Software Penetration Testing", Gary McGraw
http://ieeexplore.ieee.org/document/1392709/

Si conclude con queste ultime righe il mio percorso di studi e di conseguenza si conclude una fase della vita per me molto importante. Sono tante le persone che ho incontrato in questi anni a Camerino e tutte, nel loro piccolo, mi hanno aiutato a raggiungere questo traguardo così importante.

Ringrazio il mio relatore, il Prof. Fausto Marcantoni, che mi ha permesso di affrontare con lui questo periodo di tesi. La sua disponibilità, i suoi preziosi consigli e la sua sempre presente supervisione hanno contraddistinto la sua personalità, rendendolo un relatore eccezionale come pochi.

Vorrei ringraziare la mia famiglia, che mi è sempre stata accanto e mi ha permesso di affrontare al meglio questa bellissima esperienza. Mi hanno sempre sostenuto ed appoggiato in tutte le mie scelte e consigliato nelle decisioni più difficili da prendere. Mi rendo conto che questo non sia stato facile e di quanto la loro fiducia è stata per me necessaria. Senza di loro non sarei mai diventata la persona che sono.

Vorrei ringraziare il mio ragazzo, che in questi ultimi anni ha sempre creduto in me, anche quando forse non era la cosa più giusta da fare. Mi è stato accanto nei momenti belli e soprattutto in quelli brutti facendomi capire che nella vita bisogna sempre lottare e contare sulle proprie forze. Lo ringrazio per la pazienza con cui mi ha sopportato e per la grande forza che mi ha dato e ancora mi da nella vita di tutti i giorni. È sempre stato un punto di riferimento, ha sempre avuto il consiglio più opportuno ed è anche grazie a lui se sono arrivata fin qui.

Infine un grandissimo grazie va a tutti i miei amici. A tutti gli amici che ho incontrato e conosciuto a Camerino, che mi hanno accompagnato in questo periodo di studi e che hanno trascorso con me lunghe giornate in questo piccolo paese. A tutti gli amici che mi hanno sostenuto a distanza e che conosco da anni con i quali ho passato attimi indimenticabili.