Simple Network Management Protocol
Analisi e Implementazione con Nagios e Cacti

Michele Manzotti

Indice

Elenco delle figure

Elenco delle tabelle

Introduzione

L'inarrestabile sviluppo della tecnologia informatica ha prodotto, specialmente nell'ultimo decennio, repentini cambiamenti soprattutto nel settore delle telecomunicazioni. Si pensi come l'utilizzo di Internet grazie alla banda larga, abbia profondamente modificato il concetto di personal computer, non piú come singolo strumento di lavoro, ma come mezzo in grado di comunicare con il tutto mondo. Ovviamente lo sviluppo delle reti, non piú circoscritte alle singole organizzazioni, ha portato radicali cambiamenti principalmente nelle realtá lavorative aumentando sensibilmente il grado di interoperabilitá tra di esse.

Questa crescita di necessitá di comunicazione sempre piú istantanea e allo stesso tempo sicura, non ha fatto altro che inizialmente far nascere, e successivamente sviluppare, all'interno della organizzazione aziendale, la figura del network system administrator, al quale l'organo direttivo oggi, riconosce sempre piú importanza. All'amministratore di rete é dunque indispensabile uno strumento che gli permetta di controllare, gestire, amministrare, in una parola monitorare la rete, riducendo in questo modo possibili problemi relativi alla sicurezza e allo stato dei servizi di rete.

Nel 1989 nasce il protocollo SNMP, che viene pensato come punto di partenza in grado di risolvere e prevedere tutti i problemi che possono verificarsi nella gestione di una rete. La versatilitá di questo protocollo, ha fin da subito trovato un riscontro positivo dal mondo degli sviluppatori e dalle aziende del settore, favorendone lo sviluppo di altre due versioni.

Oggi lo standard SNMP é supportato da una grandissima quantitá di dispositivi alcuni dei quali esulano dalla categoria dei componenti di rete come ad esempio alcune stampanti, e costituisce un protocollo sul quale la maggior parte del software di monitoraggio é sviluppata.

Obiettivo della tesi

L'obiettivo iniziale di questa tesi é analizzare tutta la teoria che regola questo protocollo in modo da poter acquisire un background di conscenze generali sull'argomento che permetta poi di capire i diversi concetti che regolano il funzionamento dei vari software di monitoraggio di rete. Conoscere tutti gli aspetti di tale protocollo, consentirá infatti l'utilizzo di tecnologie che andranno poi testate e implementate direttamente all'interno della rete dell'universitá.

Un successivo scopo di questo elaborato é quello di fornire documentazione utile per l'installazione e la configurazione di applicativi per il monitoraggio, al fine di congiungere concetti teorici ad aspetti pratici.

L'obbiettivo finale é dunque quello di realizzare un vero e proprio centro di controllo, che grazie all'ausilio di software open-source, si possa testare e implementare direttamente all'interno della rete universitaria, mettendo cosí nelle mani dell'amministratore un utile e completo strumento.

Struttura della tesi

La tesi si compone di otto capitoli.

- Il primo capitolo fornisce una conoscenza sulle nozioni base del protocollo SNMP. Nella prima parte vengono esposte alcune considerazioni sulle specifiche riguardanti il protocollo SNMP. Nella seconda invece, vengono elencati e conseguentemente trattati i singoli messaggi che costituiscono i punti di partenza della comunicazione.
- Il secondo capitolo è dedicato allo sviluppo evoluzionistico del protocollo stesso. Vi sono tre versioni del protocollo SNMP. In questo capitolo si discute in dettaglio ogni singola versione, tracciandone le differenze principali.
- Nel terzo capitolo viene presentato il Management Information Base. Viene descritto il suo specifico funzionamento all'interno del protocollo SNMP, descrivendone la struttura nella quale sono catalogate tutte le informazioni del device monitorato.
- Il quarto capitolo riguarda la scelta del software per il monitoraggio. Nella prima parte vengono presentati prodotti commerciali che lavorano sotto piattaforma Microsoft. Nella seconda invece sono elencati e discussi software open-source, che ad ogni modo svolgono la stessa funzione di quelli commerciali.
- Nel quinto capitolo viene descritto come installare un sistema LAMP. Questo segmento della tesi ha un orientamento piú pratico; si pone come una sorta di guida passo passo per l'amministratore di rete nella configurazione e implementazione di un sistema Linux.
- Il sesto capitolo é dedicato a NAGIOS. Questo software dalle enormi potenzialitá ha fatto della sua versatilitá e adattabilitá a ogni possibile evenienza le sue armi vincenti. Questo capitolo é senza dubbio uno tra i piú corposi dell'intero elaborato. Nella prima fase viene descritta in maniera esaustiva tutta l'installazione; per poi passare alla descrizione della struttura presente nei file sorgenti. Successivamente si pone l'attenzione sul corretto funzionamento di uno script per il discover dei device presenti sulla rete, al fine di ottenere un sorgente pronto per essere utilizzato da NAGIOS. Nell'ultima fase sono trattati alcuni plugin, discutendo inizialmente quelli attui al monitoraggio e successivamente i restanti, tipici di NAGIOS. Il capitolo si conclude descrivendo come implementare il servizio di notifiche e-mail.

- Nel settimo capitolo viene presentato CACTI. Questo software é un applicativo che tramite l'uso di plugin e template consente di fare reportistica, disegnare grafici e tracciare lo storico dei servizi monitorati. Nella parte iniziale viene affrontata l'installazione; poi l'attezione passa sulla configurazione dei plugin, descrivendone il loro funzionamento principale. Nell'ultima perte vengono trattati i template, utili per il monitoraggio di particolari device.
- Nell'ultimo capitolo infine si descrive una possibile soluzione per automatizzare tutto il processo di configurazione; viene infatti mostrato un piccolo script che in maniera del tutto automatica richiama i vari applicativi, e configura la macchina dell'amministratore di rete in modo da essere pronta per l'utilizzo.

Nella parte finale si trovano le conclusioni nelle quali si analizza gli studi effettuati descrivendo le impressioni e gli esiti avuti durante la trattazione dell'intera tesi.

Capitolo 1

Simple Network Management Protocol

Nell'informatica, come in qualsiasi ramo scientifico, lo studio approfondito sui fenomeni che regolano specifiche leggi di comunicazione costituisce il principale obbiettivo per l'analisi e sviluppo di nuove soluzioni. Specialmente nel campo delle reti, dove ciascun device é connesso e contribuisce all'esistenza della network stessa, é fondamentale stabilire delle regole che permettano la comunicazione dello stato degli apparati all'amministratore incaricato di monitorare la propria infrastruttura di rete. Per rispondere a questa richiesta sono stati concepiti due protocolli principali: SNMP [1] e CSMIE/CMP [2]. Nella tesi viene documentato solamente il primo protocollo poiché é risultato essere quello maggiormente implementato negli applicativi open-source utilizzati, oltre ad essere supportato praticamente da tutti i produttori di hardware ed apparecchiature di rete.

Il protocollo SNMP ufficialmente nasce nel 1989 e viene definito dalla Internet Engineering Task Force (IETF)[19]. A partire da questa data SNMP diventa un protocollo standard utilizzato per controllare lo stato degli elementi di rete tramite un unico strumento applicativo. SNMP é costituito da una serie di funzioni per il management di rete che comunicano tra loro attraverso l'Internet Protocol (IP). La prima realizzazione é stata fatta sul protocollo TCP/IP, ma in seguito é stato sviluppato anche su reti IPX e AppleTalk. Tale protocollo consente all'amministratore di tenere sotto controllo le performance della rete e di scoprire e isolare, in tempo reale, malfunzionamenti e guasti.

SNMP opera allo strato applicativo del livello OSI[] (livello 7) e utilizza un'architettura di comunicazione di tipo client-server con il protocollo UDP[] (User Datagram Protocol) sfruttando di default la porta 161.
SNMP é costituito principalmente da quattro parti:

- **Manager** é uno strato software che viene installato su un computer della rete per essere utilizzato come stazione di controllo, Management Station

[1] SNMP *é l'acronimo di Simple Network Management Protocol*
[2] CSMIE/CMP *é l'acronimo di Common Management Information Service Element / Common Management Information*

o piú propriamente detto Network Management Station (NMS), mettendo in comunicazione diretta l'amministratore di rete e il sistema da gestire. Esistono svariate applicazioni commerciali e gratuite che svolgono tale funzione compatibilmente con le principali piattaforme (UNIX, Windows e Macintosh) permettendo all'amministratore di rete di non orientarsi su una specifica architettura.

- **Agenti** sono moduli software che risiedono sui device di rete (router, switch, stampanti) e segnalano svariate informazioni come gli indirizzi fisici, il carico di lavoro e altri dettagli tecnici utili all'amministratore. Questi dati vengono poi salvate all'interno di un database, il Management Information Base (MIB).

- **Management Information Base** Il MIB costituisce l'insieme delle informazioni effettivamente recuperabili dal manager sul dispositivo da monitorare. Nello specifico il MIB é definito come il database nel quale vengono risolte tutte le richieste fatte dal manager. Tale database ha una struttura gerarchica ad albero.

- **Protocollo per la gestione** Consente al manager di recuperare i valori delle variabili MIB grazie al comando get, e all'agent di segnalare la presenza di particolari eventi alla stazione di gestione grazie al comando trap.

Tale sistema permette di interrogare i diversi segmenti di rete creando delle connessioni virtuali tra il Network Manager e l'Agent SNMP, presente sul dispositivo remoto, e comunicando informazioni o eventuali segnalazioni di errore, come riportato in figura1.1.

Figura 1.1: Installazione Fedora

Le informazioni che sono possibili gestire per una particolare apparecchiatura, mediante il protocollo SNMP, vengono indicate come oggetti. Il Management Information Base (MIB) é rappresentato dall'insieme di questi elementi.

1.1 Considerazioni sul protocollo

La principale caratteristica che ha reso possibile un cosí repentino sviluppo del protocollo SNMP é stata la sua ineguagliabile capacitá di adattamento a qualsiasi dispositivo presente in una rete di computer. Gli agenti SNMP, infatti, possono essere collocati su qualsiasi computer, switch, bridge di rete, router, modem e anche stampanti, sottolineando la sua forte interoperabilitá con le diverse tecnologie. Infatti questo protocollo possiede un alto grado di flessibilitá che ha permesso la creazione di nuove funzionalitá degli agenti SNMP, rispondendo al maggior numero di specifiche hardware presenti sul mercato. Inoltre grazie allo sviluppo di semplici applicazioni software che si interfacciano con questi agenti ha reso estremamente semplice l'utilizzo del protocollo SNMP da parte degli amministratore di rete che sempre piú hanno bisogno di conoscere lo stato dei dispositivi hardware anche non necessariamente con i compiti specifici per la rete, come ad esempio le stampanti.

Ora che abbiamo spiegato le ragioni che hanno portato allo sviluppo di questo protocollo, é dovereso trattare anche i diversi punti che caratterizzano in negativo l'impiego di SNMP. In primo luogo, a differenza del nome Simple Network Management Protocol, risulta essere un protocollo abbastanza complicato da implementare; basti pensare alla complessitá delle regole che lo codificano. Inoltre é da considerare anche l'effecienza di questo protocollo, poiché la banda utilizzata spesso risulta essere superflua alla reale necessitá di comunicazione che passa all'interno del canale di trasmissione. Spesso infatti oltre al dato che racchiude l'informazione, vengono trasmessi spiegazioni riguardanti i diversi paramentri e la descrizione dell'oggetto. Tutti questi dettagli occupano un relativo spazio all'interno di ogni pacchetto, con il risultato di utilizzare banda per la trasmissione di informazioni non prettamente utili. Ultimo lato negativo riguarda la dichiarazione degli oggetti, ossia il modo con il quale il protocollo identifica le variabili all'interno di un database MIB come stringhe di byte dove ciascun byte referenzia un particolare nodo, che potrebbe rivelarsi non sufficentemente intuitiva e soprattutto non semplice da memorizzare.

Sebbene siano presente queste imperfezioni possiamo comunque affermare che non influiscono sull'utilitá di questo protocollo. Infatti per quanto concerne la complessitá delle sue regole, tale difficoltá é esclusivamente dei programmatori poiché l'utente finale non avrá l'onere di capire a fondo la complessitá degli algoritmi con i quali i sui dati vengono utilizzati. Invece per quanto riguarda l'efficienza e l'occupazione di banda possiamo dire che grazie al continuo sviluppo delle tecnologie di comunicazione sempre piú performanti il problema che la maggior parte della trasmissione di informazioni all'interno dei pacchetti SNMP sono sostanzialmente utili é pressoché relativo e con il tempo non particolarmente significante.

Tuttavia una delle critiche piú difficili da spiegare sul protocollo SNMP ri-

guarda la reale motivazione della sua esistenza rispetto ad altri tool che con altre tecniche ricavano comunque sia le stesse informazioni. Infatti viene spontaneo domandarsi come mai é nata la necessitá di creare un nuovo protocollo piuttosto che utilizzare uno giá esistente come ad esempio l'impiego di una shell remota via telnet per collegarsi al dispositivo e scaricare tutte le informazioni necessarie. Inoltre un altro problema riguarda i venditori, che creano software basati su SNMP spacciandoli come una alternativa alla console remota piuttosto che un nuovo prodotto per la gestione e l'analisi della rete. Infatti l'attenzione viene posta maggiormente sulla interfaccia grafica rispetto al sistema automatico di raccolta dei dati. In aggiunta i principali venditori tendono a personalizzare il proprio ramo degli oggetti in base al quale solo attraverso il loro specifico software concesso con il dispositivo é in grado di interrogare, limitando in questo modo lo sviluppo di software open-source.

Per dare una risposta a queste riflessioni, possiamo innanzitutto approvare che in effetti esistono vie alternative a SNMP, ma comunque sia sono state rimpiazzate dalla universale popolaritá e interoperabilitá di questo protocollo. L'incredibile capacitá di integrazione con i piú svariati componenti hardware ha di fatto stabilito che non esiste alternativa a SNMP e la sua reale potenzialitá é maggiormente visbile quando si gestiscono network di grande dimensioni.

1.2 Messaggi SNMP

In qualsiasi protocollo la necessitá di stabilire delle regole che coordinano la comunicazione costituisce la principale esigenza per il corretto funzionamento del protocollo stesso. Non fa eccezione SNMP dove tutti messaggi che viaggiano sulla rete vengono incapsulati in una PDU[3]. Il formato di questi pacchetti non é semplice da definire, ma fortunatamente la loro complessitá viene gestita dai software presenti sui manager.

Di seguito sono elencati i principali comandi che sono stati creati per agevolare le richieste da parte dell'amministratore di rete:

- **Get Request** Questo comando viene utilizzato dall'amministratore per richiedere valori specifici all'agent servendosi del comando get. La maggior parte di questi valori ricadono sulle prestazioni e condizioni di funzionamento del dispositivo remoto. Il vantaggio del protocollo SNMP é visibile anche in questo caso poiché é possibile richiedere specifiche informazioni senza effuttuare un login remoto con un inutile carico di banda generato da una connessione TCP.
- **Get Next Request** Questo comando, molto simile al precedente, viene impiegato per individuare sulla rete la presenza di nuovi dispositivi che supportano il protocollo SNMP. In questo caso il manager si fa carico della ricerca, iterando su ogni nodo sempre lo stesso comando. Quando si verifica un errore, il manager interrompe la sua analisi poiché denota la fine della mappatura dell'intera rete.

[3] *PDU* é l'acronimo di Protocol Data Unit e indica l'unitá d'informazione scambiata tra due reti.

- **Set Request** Questo comando, risulta il piú interessante, in quanto permette di effettuare opportune operazioni di configurazione e controllo sul dispositivo remoto. Tramite l'opzione set, infatti é possibile disconnettere gli utenti, disabilitare l'interfaccia, modificare valori statici all'interno degli oggetti MIB e altre operazioni simili.
- **Get Response** Questo comando viene impietato dall'agent in risposta alle richieste fatte dal manager mediante i comandi di tipo get, get-next e set.
- **Trap Message** Questo comando risulta molto valido per la segnalazione di eventi, come ad esempio la notifica degli errori. Per il corretto funzionamento é indispensabile configurare l'agent in modo che invii un messaggio trap al manifestarsi di un particolare evento, e conseguentemente il manager che resti in ascolto in attesa della ricezione.

Questi messaggi appartengo alla prima versione del protocollo SNMP e in genarale le comunicazioni avvengono in due modi: una sequenza di tipo request/response[FIGURA 1] inizializzata dal manager e un'altra di tipo trap avviata dall'agent. [FIGURA 2]

QUI DUE FIGURE

Nella seconda versione del protocollo sono stati creati altri tre comandi:

- **Get Bulk Request** Questo comando viene utilizzato per effettuare in un'unica comunicazione diverse transazioni di tipo request/response evitando cosí il continuo invio di informazioni relative ad un dispositivo. Eccetto nel caso in cui ci sia una singola interazione, questo comando si comporta come una serie continua di GetNext request/response.
- **SNMPv2 Trap Request** Questo comando, molto simile al trap della versione precedente, é stato tuttavia creato in quanto nella seconda versione sono presenti piccole differenze che hanno reso necessario lo sviluppo di un nuovo messaggio di tipo trap.
- **Inform Request** Questo comando, molto simile all'ACK[4] nelle connessioni TCP, ha solamente la funzione di confermare la notifica di certi eventi al manager di rete.

1.3 Notazione degli oggetti SNMP

Molto spesso é facile compiere nell'errore di associare la lista degli elementi di un oggetto SNMP al MIB, Management Information Base, quando in realtá quest'ultimo é semplicemente un'astrazione di database.

Lo standard MIB che definisce i diversi oggetti SNMP si trova nella RFC 1213, dove é presente anche una serie di oggetti utili per il monitoraggio della attivitá IP, TCP e UDP, delle interfacce rete e del sistema in generale. A ogni

[4] *ACK* é l'acronimo di Acknowledge e identifica il segnale emesso in risposta alla ricezione di un'informazione completa.

singolo elemento presente negli oggetti MIB é associato sia un nome ufficiale che un valore ufficiale espresso in dot-notation. Per comprendere meglio questo aspetto é utile un esempio chiarificatore: il *sysUpTime* che indica da quanto tempo il device é acceso rappresenta il nome ufficiale di un oggetto MIB, al quale é associato anche un valore ufficiale espresso nel formato .1.3.6.1.2.1.1.3.0 come in figura 1.2.

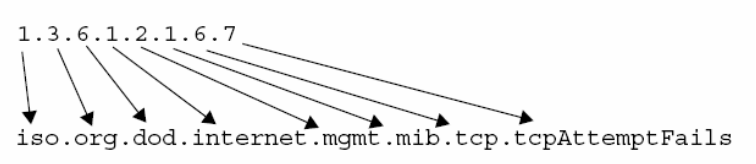

Figura 1.2: Traduzione di un valore in dot-notation

Inoltre interrogando il device attraverso il comando *snmptraslate*[1], é possibile conoscere le corrispondenze nome-valore e viceversa. Si fa presto a capire che é piú semplice utilizzare un nome piuttosto che una cifra in dot-notation, proprio come negli indirizzi di Internet dove si preferisce utilizzare un nome anziché un indirizzo IP.

Capitolo 2

Evoluzione del Protocollo

L'ingegneria del software insegna che gran parte dei progetti informatici, che siano software o protocolli come in questo caso, difficilmente raggiungono il grado di perfezione alla prima realise, ma subiscono numerose modifiche durante l'intera vita del progetto. Non é da meno il protocollo SNMP che ha visto la realizzazione di tre diverse versioni:

- SNMPv1: costituisce la prima versione e si avvale dei nomi delle community impiegate come una sorta di password. Viene documentato nelle RFC[1] 1155, 1156 e 1157;
- SNMPv2: in questa versione sono state aggiunte nuove funzionalitá come la crittografia tramite MD5. Le specifiche si trovano nelle RFC 1450, 1451 e 1452;
- SNMPv3: risulta la versione finale, anche se difficilmente viene utilizzata. Tutti i dettagli sono presenti nelle RFC 2571, 2572, 2573 e 2574.

Dopo aver spiegato nel precedente capitolo le ragioni per le quali questo protocollo si é affermato rispetto ad altri, in questa parte viene invece illustrata una panoramica sulle principali caratteristiche che ne hanno contraddistinto l'evoluzione.

2.1 SNMP versione 1

Questa prima versione del protocollo comprende un insieme di specifiche che saranno presenti anche nei successivi sviluppi. Tali caratteristiche sono:

- Indipendenza dalle piattaforme utilizzate e dalle architetture di rete;
- Facilitá d'uso, in quanto la prima versione forniva solo i comandi essenziali;
- Trasportabilitá su diversi device di rete;

[1]RFC *acronimo di Request for Comments, é un documento che riporta informazioni o specifiche riguardanti nuove ricerche, innovazioni e metodologie dell'ambito informatico o piú nello specifico, di internet.*

- Personalizzazione applicativa, ossia la possibilitá di estendere nuove funzioni di management senza dover riconfigurare gli agent;
- Poca affidabilitá dovuta alla essenzialitá del protocollo di trasporto UDP;
- Autonomia rispetto ad altri servizi di rete;
- Basso costo di implementazione sui device.

Inoltre gli agent restano in ascolto sulla porta UDP 161, mentre le risposte inviate al manager o Network Management Station utilizzano un numero di porta casuale. Le eccezioni, come le trap, e i messaggi d'errore generati dall'agent e comunicati al manager sono inviati in maniera asincrona sfruttando la porta UDP 162. Infine la massima dimensione per un messaggio SNMP é circoscritta unicamente dalla massima dimensione del payload del pacchetto UDP, ossia 65507 byte.

2.2 SNMP versione 2

Nella prima versione del protocollo SNMP é stata progettata e realizzata una struttura molto semplice e leggera in grado di soddisfare esigenze di performance, ma allo stesso tempo si é trascurato aspetti relativi alla sicurezza e limitato alcune funzionalitá.

Per questi motivi, é nata la necessitá di realizzare una nuova versione che coprisse questi aspetti: SNMPv2. Quest'ultima comprende una serie di subversioni che non hanno avuto una grande diffusione, ad eccezione per SNMPv2c impiegato in alcuni casi. Tuttavia questa versione non é mai stata standardizzata da IETF[2].

Le novitá che sono state aggiunte rispetto a a SNMPv1 nella versione SNMPv2c sono legate ad operazioni che facilitano l'accesso ai MIB. Questi sviluppi sono visibili nei seguenti comandi:

- **GetBulk** Questo comando é molto simile al GetNext della versione SNMPv1 con la possibilitá di indicare la quantitá delle richieste di lettura. Inoltre se nel campo non-repeaters é presente l'intero uno significa che GetBulk effettua anche il comando get e, nella circostanza in cui non esista la variabile, non segnali un errore, ma piuttosto esegue direttamente la GetNext. Una considerazione da fare su questo comando riguarda il valore di max-repetitions, ossia il numero di GetNext da compiere: poiché se questo valore é minimo risulterebbe piú conveniente fare delle singole GetNext, viceversa, se é alto potrebbe verificarsi di avere gran parte delle informazioni inutili.
- **Inform** Questo comando é paragonabile al trap di SNMPv1 solamente che viene impartito dall'agent verso il manager. Inoltre il manager esegue una operazione di risposta alla ricezione del messaggio dell'agent.

[2]IETF *Internet Engineering Task Force é una comunitá aperta di tecnici, specialisti e ricercatori interessati all'evoluzione tecnica e tecnologica di Internet. I gruppi di lavoro producono documenti per la standardizzazione e non denominati RFC.*

Un ulteriore miglioramento introdotto con SNMPv2c riguarda i messaggi d'errore. Infatti frequentemente in SNMPv1 questo tipo di messaggi risultavano generici e poco espliciti. Nella successiva versione invece sono state apportate delle modifiche permattendo al manager di avere una visione piú dettagliata di quelli che sono stati i problemi avuti dall'agent alla ricezione del messaggio. Tale cambiamento é utile in particolar modo per il comando set. Infine la seconda versione di SNMP, é stata sviluppata per essere implementabile su tutti i protocolli del livello di trasporto della pila OSI.

Tutti queste innovazioni di SNMPv2 hanno senza dubbio ottimizzato le performace, grazie principalmente al GetBulk, ma allo stesso tempo hanno anche appesentito il protocollo diminuendone sensibilmente l'effecienza e probabilmente questa é stata la ragione per la quale non avuto un grande successo.

2.3 SNMP versione 3

L'ultima versione del protocollo SNMP nasce nel 1999 in risposta alle mancanze derivanti dalle precedenti versioni soprattutto nell'ambito della sicurezza delle trasmissioni. I principali obbiettivi posti dagli sviluppatori convergevano nella trasportabilitá del maggior numero di componenti hardware, la compatibilitá con i MIB delle versioni precedenti e la scalabilitá dell'architettura. Sebbene quest'ultima versione ha portato significativi cambiamenti non ha avuto un riscontro positivo sul mercanto, dove ancora viene preferita la prima versione. Questo perché viene apprezzata maggiormente la semplicitá e facilitá d'uso piuttosto che l'introduzione di nuove funzionalitá.

Nell'ultima versione é stata rivoluzionata anche la classica architettura di tipo manager-agent con una piú articolata engine-applications:

- **Snmp-Engine** é un oggetto che comprende diversi elementi:
  - un dispositivo per smistare i messaggi;
  - un dispositivo per elaborare i messaggi;
  - un dispositivo per la sicurezza;
  - un dispositivo per il controllo dell'accesso.
- **Snmp-Applications** é invece un'applicazione che contiene racchiude diverse funzioni:
  - un generatore di comandi;
  - un ricettore di notifiche;
  - un risponditore ai comandi.

Infine la struttura dei messaggi SNMPv3 é fondamentalmente diversa rispetto alle precedenti versioni in quanto sono presenti alcuni segmenti relativi alla sicurezza e al controllo dell'accesso. Quest'ultimi aspetti sono implementati direttamente attraverso crittografia, funzioni di hash e altri tecniche che consentono l'autenticazione dei pacchetti. In questo modo si creano delle connessione sicure in modo da far arrivare al device SNMP solamente pacchetti autorizzati,

eliminando cosí tutti quelli non autenticati.

Nella terza versione del protocollo SNMP é quindi possibile stabilire diversi livelli di sicurezza:

- senza autenticazione;
- con semplice autenticazione;
- con autenticazione e codifica dei dati.

Capitolo 3

Management Information Base

La Management Information Base, piú semplicemente MIB, é una collezione di oggetti in un database virtuale comprendente tutte l'insieme dell'informazioni gestite dall'agent SNMP sul device remoto. Il database é di tipo gerarchico ad albero dove le foglie rappresentano gli oggetti monitorati nelle quali sono inglobate le informazioni.

Gli oggetti MIB possono essere classificabili in due categorie analoghe, ma con piccole notevoli differenze che riflettono l'organizzazione in una struttura ad albero:

- **Discrete MIB Objects** Questi oggetti discreti SNMP si collocano in una zona precisa e ben definita all'interno del MIB comprendendo dati utili al monitoraggio. Si distinguono dai Tablet MIB Objects, delineati successivamente, poiché alla fine del valore numerico espresso in dot-extension é presente il suffisso ".0". Tuttavia anche se quest'ultimo valore non viene espresso é da considerarsi comunque implicito.

- **Table MIB Objects** Questi oggetti dichiarati come tabelle, comprendono molteplici informazioni in un solo oggetto SNMP. Ció che differenzia con quelli precedenti é l'estensione "." seguita da un numero che referenzia il particolare valore all'interno della tabella alla quale fa riferimento l'oggetto SNMP.

La notazione dot-extension indica quindi l'esatto indirizzo dell'oggetto SNMP. Nell'ipotesi di oggetti discreti sará zero, al contrario nell'eventualitá di oggetti table, risulterá essere la cifra corrispondente all'indice della tabella. Spesso sono oggetti discreti, poiché contegno informazioni tipicamente utili al monitoraggio e permettono di ricavare i dati conoscendo solo il nome dell'oggetto che si vuole interrogare. Questi valori sono solitamente standard e dunque utili per confrontare le performace di diversi prodotti.

Nella maggior parte dei casi l'albero MIB presente sul dispositivo é progettatto e implementato direttamente dai produttori, e difficilmente é possibile

aggiungere o modificare questo tipo oggetti. Tale concetto potrebbe quindi entrare in conflitto con quanto detto pocanzi. In effetti ogni produttore, oltre alla struttura cosí detta standard, implementa sul dispositivo un proprio ramo nel quale inserisce tutti parametri specifici per ogni particolare device. Per questa ragione quando si esalta l'estendibilitá SNMP si considera la forte propensione all'adattabilitá del software che amministra il protocollo SNMP, dal momento che puó essere facilmente configurato per interfacciarsi con le nuove funzionalitá del produttore.

Gli oggetti presenti in una MIB sono specificati secondo delle strutture SMI.

3.1 Struttura delle informazioni

L'archietettura e la gerarchia delle informazioni degli oggetti MIB sono definite dallo standard Structure of Management Information, piú semplicemente SMI, presente nella RFC 1155[2]. Gli elementi principali di questo modello sono essenzialmente tre elementi:

- **OID** Object Identifier, ossia il nome che identifica l'oggetto MIB, e come precedentemente detto puó essere sottoforma numerica o letterale;
- **ASN.1** Abstract Syntax Notation One, che esplicita il tipo di dato riferito ad ogni oggetto MIB e consente l'autonomia dalla macchina utilizzata;
- **BER** Basic Encoding Rules, ovvero il sistema di codifica e decodifica del codice utilizzato.

I principali vantaggi nell'utilizzare una struttura come SMI sono l'estendibilitá e la semplicitá in quanto questa architettura permette l'utilizzo di valori scalari o tabelle a due dimensioni per identificare oggetti MIB.

I piú significativi insieme di dati specificati dall'SMI sono:

- **Universal** che comprende tipi come Integer, Null, Octect string, Object identifier (indispensabile per individuare l'oggetto MIB), Sequence e Sequence-of (utile per identificare le tabelle);
- **Application**, che racchiude tipi adatti per una specifica applicazione. Nella tabella 3.2 sono catalogati i tipi di dato piú comuni.

La flessibilitá di questo protocollo é riscontrabile anche in questa circostanza poiché SNMP mette a disposizione tutti gli strumenti per la definizione di oggetti MIB propri.

3.2 Strutture delle MIB

L'architettura del MIB, come precedentemente riferito, segue una struttura ad albero in cui le foglie, poste al termine del ramo, contengono le informazioni. Una tipica rappresentazione di questa struttura é mostrata in figura 3.1 dove é visibile il percorso del ramo MIB2.

Tipo di Dato	Descrizione
Counter	contatore intero che puó essere solamente incrementato. Se raggiunge la soglia massima riparte da zero
Gauge	valore che puó essere incrementato e decrementato. Se raggiunge la soglia massima deve essere azzerato
Ipaddress	identifica l'indirizzo IP del dispositivo. Generalmente visualizzato in dot-notation.
Physaddress	identifica gli indirizzi MAC del dispositivo. Generalmente visualizzato come una sequenza di valori esadecimali precedutti dal prefisso "hex"
Timeticks	contatore di secondi a partire da un istante di tempo stabilito

Tabella 3.1: Tipi di dati Application piú comuni

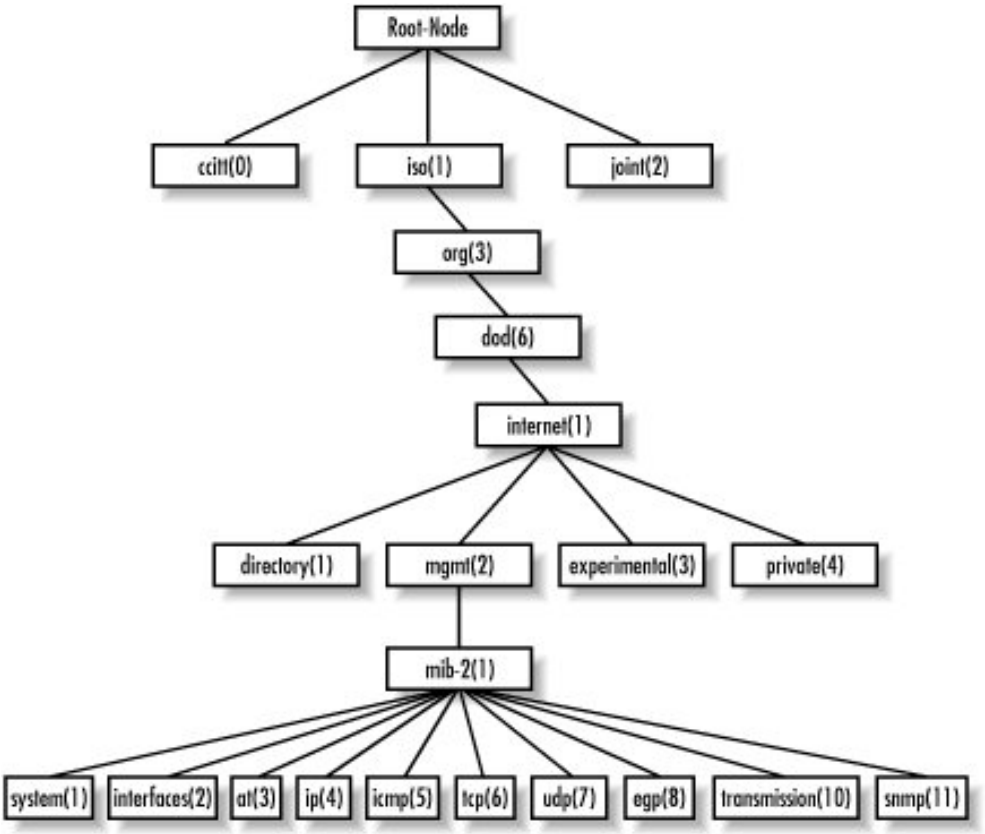

Figura 3.1: Struttura ad albero del MIB

Osservando la figura **??**, si nota che subito dopo la radice dell'albero MIB sono illustrate le principali organizzazioni di standardizzazioni riconosciute a livello internazionale: *ccitt*[3], *iso*[4] e *joint-iso-ccitt*. Proseguendo il cammino si incontra il nodo *org* che rappresenta la sigla "organization" indicante tutti gli enti mondiali riconosciuti. Procedendo verso il basso é possibile visualizzare la sigla *dod* che individua il *Department of Defense* statunitense. Giunti a questo punto si trova il nodo *internet*, il quale a sua volta si divide in altri quattro nodi:

- *Directory*: ramo lasciato da *osi* per successivi sviluppi;
- *Mgmt*: comprende tutte le MIB specificate dall'*Internet Architecture Board*, detto anche IAB[5], un'organizzazione che sorveglia l'attivitá di IETF, invoca e controlla l'Internet Standards Process e nomina l'Editore delle RFC;
- *Experimental*: ramo dedicato a sviluppi ed esperimenti futuri su Internet;
- *Private*: raccoglie tutte le MIB proprietarie. Infatti i costruttori dopo aver fatto la registrazione presso l'Internet Assigned Number Authority, IANA[6] possono aggiungere il proprio ramo sotto la voce *enterprise*.

Utilizzando la notazione ASN.1 é possibile percorre l'albero dell'oggetto MIB:

internet OBJECT IDENTIFIER ::= iso(1) org(3) dod(6) internet(1) ovvero OID = 1.3.6.1

In conclusione lo sviluppo del ramo *internet* é stata la chiave del reale successo del protocollo SNMP. Sfruttando i vantaggi posti da una gerarchia ad albero che fondamentalmente non pone limiti all'implementazioni di nuovi oggetti MIB, SNMP ha goduto di un'enorme quantitá di aggiornamenti che lo ha reso compatibile la maggior parte dell'hardware in circolazione.

3.3 RMON

Sebbene il protocollo SNMP dall'inizio della sua comparsa ha riscontrato solamente pareri positivi, si sentiva la mancanza di un oggetto dedicato al monitoraggio remoto, ovvero che isolasse le sole informazioni utili per l'analisi remota della rete da quelle prettamente di sistema.

In risposta a questa esigenza nel 1991 é nato il ramo RMON, Remote Network Monitoring MIB, pubblicato ufficialmente nella RFC 1271[7]. All'interno di questo documento sono presenti i tools utili per una Network Manament Station, NMS, cosí da poter controllare e gestire una rete. Scendendo nel ramo RMON é possibile trovare una serie di oggetti simili a quelli in figura 3.2.

Osservando l'insieme di questi oggetti MIB, Matrix é quello che risulta maggiormente rilevante per l'amministrazione delle reti switchate. Questo particolare oggetto MIB é composto da due tabelle:

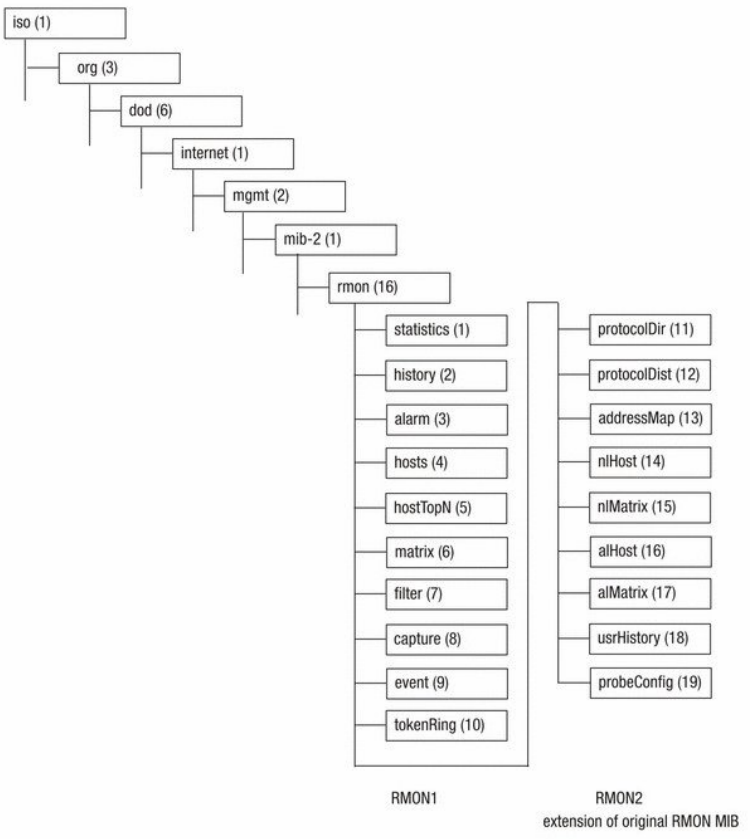

Figura 3.2: Ramo dell'oggetto RMON

- Una di controllo, la **Matrix Control Table**. In questa tabella sono presenti gran parte di tutte le coppie di indirizzo-destinazione specifiche di un'interfaccia di rete. Nel caso di riempimento dell'intera tabella vengono eliminate le associazioni piú vecchie.
- Una di dati, comprendente a sua volta altre due tabelle:
  - **Matrix Source-to-Destination Table** detta anche *matrixSDTable*. All'interno di questa tabella sono presenti un insieme di valori che permettono di tracciare delle statistiche sulla quantitá di traffico passante tra due interfaccie. Questi parametri sono per lo piú contatori di pacchetti, di errori e di byte. Nell'eventualitÃ di riempimento dell'intera tabella vengono rimosse le informazioni meno recenti. Infine gli indici presenti nella tabella hanno il seguente ordine:
    1. Interfacce monitorate
    2. Indirizzo fisico sorgente
    3. indirizzo fisico destinazione
  - **Matrix Destination-to-Source Table** detta anche *matrixDSTable* Questa tabella é pressoché uguale alla precedente, variano soltanto l'ordine degli indici:
    1. Interfacce monitorate
    2. Indirizzo fisico destinazione
    3. Indirizzo fisico sorgente

Ai giorni d'oggi lo standard maggiormente diffuso é quello relativo al ramo del MIB2 poiché le informazioni presente sul ramo MIB difficilmente sono implementate sui prodotti. Per tale ragione nel prossimo paragrafo viene affrontato in detteglio l'oggetto MIB2.

3.4 MIB2

MIB2 rappresenta indubbiamente un tra i piú importanti gruppi all'interno dell'albero, dato che costituisce la fonte principale di informazioni per le workstation che supportano SNMP. La seconda versione del Management Information Base é definata nella RFC 1213[8] e la struttura ad albero é come quella illustrata nella figura 3.1. Ognuno dei nodi sottostanti contiene informazioni specifiche per il monitoraggio e l'analisi della rete.

Nella tabella 3.2 vengono esposti e descritti i principali oggetti riguardanti MIB2.

3.5 Compilare un proprio MIB

Uno strumento essenziale che si avvale del protocollo SNMP di cui un buon amministratore di rete non puó farne a meno é un compilatore di MIB, che consente di creare ulteriore oggetti MIB da combinare con quelli giá presenti sul sistema di monitoraggio. Tuttavia questo aspetto potrebbe causare confusione agli utenti meno esperti per via della strana terminologia usata.

MIB2	OID	Descrizione
system (1)	1.3.6.1.2.1.1	Valori inerenti al sistema operativo.
interfaces (2)	1.3.6.1.2.1.2	Contiene i dati di ogni interfaccia gestita, come byte inviati, ricevuti e pacchetti errati.
at (3)	1.3.6.1.2.1.3	Nodo non impiegato.
ip (4)	1.3.6.1.2.1.4	Raccoglie tutte le informazione relative al routing e al protocollo IP.
icmp (5)	1.3.6.1.2.1.5	Contiene valori appartenenti al protocollo ICMP.
tcp (6)	1.3.6.1.2.1.6	Racchiude dati relativi allo stato della connessione e parametri sul protocollo TCP.
udp (7)	1.3.6.1.2.1.7	Dati inerenti al protocollo UDP.
egp (8)	1.3.6.1.2.1.8	Raccoglie informazioni sul protocollo EGP.
transmission (10)	1.3.6.1.2.1.10	Contiene dati dedicati alla trasmissione.
snmp (11)	1.3.6.1.2.1.11	Valori relativi al protocollo SNMP (Visibili solamente in MIB-II).

Tabella 3.2: Principali nodi appartenenti al ramo MIB2

All'atto della compilazione l'unica prerogativa da parte dell'amministratore di rete é quella di verificare che il nuovo oggetto MIB sia comunque supportato dall'agent. L'idea é analoga a quella di incorporare una nuova tabella all'intero di un database giá esistente. Di conseguenza l'agent non é condizionato dalla compilazione del nuovo oggetto MIB, in quanto ha giá implementato il fatto di dover supportare le nuove funzionalitá. Comprensibilmente l'amministratore deve conoscere quali sono le funzioni disponibili da parte dell'agent e accedere a questi oggetti come elementi tipici della struttura MIB.

Di norma quando si compila un nuovo oggetto MIB é necessario indicare la directory dove risiedono le informazione. A tal fine é utile per l'amministratore utilizzare un *Mib Browser* che permette la visualizzazione rapida dell'albero MIB. I nuovi componenti aggiunti posso essere accettati dall'agent altrimenti tramite dei warning viene segnalato il problema.

La documentazione degli oggetti MIB segue la sintassi ANS.1, vista precedentemente. In generale la maggior parte degli agent SNMP supportano oggetti MIB definiti secondo la RFC 1213 [8]. Di conseguenza l'amministratore ha la possibilitá di trasferire un proprio oggetto MIB sul proprio dispositivo e lanciando il compilatore di MIB ha la facoltá di usufruirne.

3.5.1 Esempio pratico

Per ampliare ulteriormente le nostre conoscienze in materia, vediamo ora un procedimento per scrivere e compilare un proprio oggetto MIB, nello specifico un intero. Per fare questo ci serviremo dell'applicazione *Net-SNMP*, poiché all'interno di questo pacchetto troveremo *mib2c*, un compilatore C per MIB.

Prima di iniziare é doveroso controllare che all'interno della nostra distribuzione[1] siano presenti i segue pacchetti:

- *net-snmp-perl*
- *perl-ExtUtils-MakeMaker*

Dopo aver compiuto questa opportuna verifica, proseguiamo installando l'applicazione Net-SNMP. I sorgenti sono disponibili presso il sito ufficiale[21], e tramite i seguenti comandi scarichiamo l'ultima versione e successivamente ne operiamo la decompressione:

```
# wget http://downloads.sourceforge.net/net-snmp/net-snmp-5.4.1.ta
r.gz
# tar zxvf net-snmp-5.4.1.tar.gz
```

Terminato questo passaggio, sará quindi possibile procedere con l'installazione:

```
# cd net-snmp-5.4.1
# ./configure
```

[1] vedi capitolo 5 per l'installazione di un sistema LAMP e utilizzo del gestore dei pacchetti YUM

```
# make
# make install
```

Arrivati a questo punto, é necessario procurarsi un MIB di esempio, in modo da utilizzarne la struttura, e un *Makefile*, indispensabile nell'atto di compilazione. Questi oggetti sono facilmente reperibili in un tutorial[22] presente nella documentazione di Net-SNMP. Tramite i seguenti comandi otteniamo queste risorse e le collochiamo all'interno della directory */usr/share/snmp/mibs* nella quale sono presenti anche altri MIB.

```
# cd /usr/share/snmp/mibs
# wget http://www.net-snmp.org/tutorial/tutorial-5/toolkit/mib_modu
le/NET-SNMP-TUTORIAL-MIB.txt
# wget http://www.net-snmp.org/tutorial/tutorial-5/toolkit/demoapp/
Makefile
```

Ora possiamo modificare il file *NET-SNMP-TUTORIAL-MIB.txt* sostituendo l'oggetto *nstAgentModuleObject*, con una stringa a piacimento, e volendo possiamo correggere anche la descrizione. Per il nostro esempio abbiamo utilizzato la stringa *versioneKernel*, poiché l'intero conterrá la versione del kernel del nostro sistema operativo. In questo modo abbiamo specificato l'OID, e dunque ora é possibile compilarlo:

```
env MIBS="NET-SNMP-TUTORIAL-MIB" mib2c versioneKernel
```

L'output generato dal precedente comando é il seguente:

```
# env MIBS="NET-SNMP-TUTORIAL-MIB" mib2c versioneKernel
writing to -
mib2c has multiple configuration files depending on the type of
code you need to write.  You must pick one depending on your need.

You requested mib2c to be run on the following part of the MIB tree:
  OID:                            versioneKernel
  numeric translation:            .1.3.6.1.4.1.8072.2.4.1.1.1
  number of scalars within:       1
  number of tables within:        0
  number of notifications within: 0

First, do you want to generate code that is compatible with the
ucd-snmp 4.X line of code, or code for the newer Net-SNMP 5.X code
base (which provides a much greater choice of APIs to pick from):

  1) ucd-snmp style code
  2) Net-SNMP style code

Select your choice :
```

Nella precedente schermata ci viene chiesto di scegliere quale stile adottare per la compilazione. Nel nostro caso optiamo la seconda opzione poiché risulta essere la piú recente. Il successivo output sará il seguente:

```
**********************************************************************
                GENERATING CODE FOR SCALAR OBJECTS:
**********************************************************************

  It looks like you have some scalars in the mib you requested, so I
  will now generate code for them if you wish.  You have two choices
  for scalar API styles currently.  Pick between them, or choose not
  to generate any code for the scalars:

  1) If you're writing code for some generic scalars
     (by hand use: "mib2c -c mib2c.scalar.conf versioneKernel")

  2) If you want to magically "tie" integer variables to integer
     scalars
     (by hand use: "mib2c -c mib2c.int_watch.conf versioneKernel")

  3) Don't generate any code for the scalars

Select your choice:
```

In questa parte ci viene chiesto quale stile delle API adottare. Come prima, scegliamo la seconda opzione, e l'esecuzione riprende generando quest'ultimo output:

```
using the mib2c.int_watch.conf configuration file to generate your code.
*** Warning: only generating code for nodes of MIB type INTEGER
writing to versioneKernel.h
writing to versioneKernel.c
```

```
**********************************************************************
* NOTE WELL: The code generated by mib2c is only a template.  *YOU*  *
* must fill in the code before it'll work most of the time.  In many *
* cases, spots that MUST be edited within the files are marked with  *
* /* XXX */ or /* TODO */ comments.                                  *
**********************************************************************
running indent on versioneKernel.c
running indent on versioneKernel.h
```

A questo punto, il compilatore *mib2c* ha terminano l'esecuzione creando due file: *versioneKernel.c* e *versioneKernel.h*. Per completare la creazione di un proprio oggetto MIB, ora dobbiamo compilare il *Mikefile*, ma ancor prima é necessario modificarlo nel seguente modo:

```
versioneKernel.so: versioneKernel.c Makefile
        $(CC) $(CFLAGS) $(DLFLAGS) -c -o versioneKernel.o versioneKernel.c
        $(CC) $(CFLAGS) $(DLFLAGS) -o versioneKernel.so versioneKernel.o
```

Ora, é possibile compilare attraverso il seguente comando:

```
make versioneKernel.so
```

Conclusa la fase di compilazione senza errori, possiamo proseguire modificando il file */etc/snmp/snmpd.conf*, caricando l'oggetto MIB appena creato:

```
dlmod   versioneKernel   /usr/share/snmp/mibs/versioneKernel.so
```

Eseguito quest'ultimo passaggio abbiamo terminato il lavoro; non ci rimane quindi che riavviare il servizio e controllare che il nuovo OID funzioni correttamente.

```
# /etc/init.d/snmpd restart
# snmptranslate -IR -On versioneKernel
.1.3.6.1.4.1.8072.2.4.1.1.1
```

Ora, mediante il comando *snmpset*, é possibile inserire il valore corretto, per poi poterlo interrogare:

```
# snmpset -v 1 -c public localhost versioneKernel.0 i 2
NET-SNMP-TUTORIAL-MIB::versioneKernel.0 = INTEGER: 2

# snmpget -v 1 -c public localhost versioneKernel.0
NET-SNMP-TUTORIAL-MIB::versioneKernel.0 = INTEGER: 2
```

3.6 Alla ricerca dei MIB proprietari

Con lo scopo di approfondire il discorso riguardante i MIB ho speso tempo nella ricerca di OID proprietari utili al monitoraggio. Infatti nella maggioranza dei casi le aziende di componenti di reti non rilasciano questo genere di informazioni sui loro prodotti. Nei siti ufficiali si trovano solamente notizie marginali e persino dopo una richiesta via email hanno ribadito di non poter concedere questo genere di dati. Ovviamente le ragioni sono tutte di carattere commerciale. Le aziende tendono a vendere i loro prodotti hardware con applicativi software dedicati. Perció nel caso in cui pubblicassero tali informazione renderebbero nulli gli sforzi effettuati per lo sviluppo del software del device.

In nostro aiuto viene comunque sia uno strumento potentissimo: Google. Dopo varie ricerche sul web ho raccolto un leggero numero di OID. Inizialmente avevo trovato questo .1.3.6.1.2.1.25.3.3.1.2.7 che restituiva la media, in percentuale, dell'ultimo minuto della CPU di una macchina windows,e successivamente .1.3.6.1.2.1.17.4.3.1.1 e .1.3.6.1.2.1.17.4.3.1.2 che permettono di conoscere rispettivamente gli indirizzi MAC e il numero di porta presenti su uno switch 3Com. Non soddisfatto ho cercato e ottenuto altri OID proprietari. Grazie a quest'ultimo insieme abbastanza cospicuo siamo in grado di effettuare persino delle enumerazioni sui server che hanno abilitato il protocollo SNMP. A tal fine esiste giá uno script pronto all'uso ed é scaricabile direttamente da web [9].

Ho provato direttamente questo eseguibile su qualche macchina all'interno della rete universitaria e i risultati sono stati a dir poco sorprendenti. Di seguito vediamo parte dell'output generato:

```
----------------------------------------
INSTALLED SOFTWARE
```

```
---------------------------------------

AFPL Ghostscript 8.11
AFPL Ghostscript Fonts
---------------------------------------
UPTIME
---------------------------------------
47 days, 17:10:58.57
---------------------------------------
HOSTNAME
---------------------------------------
WINSERV2
---------------------------------------
USERS
---------------------------------------
paolacupri
paolo.cera
paolotenti
---------------------------------------
DISKS
---------------------------------------
A:\
C:\ Label:  Serial Number e4962d0b
---------------------------------------
RUNNING PROCESSES
---------------------------------------
smss.exe
csrss.exe
winlogon.exe
---------------------------------------
LISTENING UDP PORTS
---------------------------------------
17
19
161
---------------------------------------
SYSTEM INFO
---------------------------------------
Hardware: x86 Family 15 Model 2 Stepping 7 AT/AT COMPATIBLE
Software: Windows Version 5.2 (Build 3790 Multiprocessor Free)
---------------------------------------
SHARES
---------------------------------------
SYSVOL
hpspec
HPLDOTT
---------------------------------------
LISTENING TCP PORTS
---------------------------------------
17
```

```
19
21
----------------------------------------
SERVICES
----------------------------------------
HTTP SSL
Intel PDS
Net Logon
----------------------------------------
DOMAIN
----------------------------------------
INFORMATICA
```

Naturalmente la maggior parte dell'output non é stato copiato, ma si fa presto a capire la reale potenza di questo protocollo.

Capitolo 4

Applicativi

Il mondo informatico presenta svariate soluzioni nel campo del monitoraggio, ogniuna con diverse peculiaritá. La prima principale distinzione, é senza dubbio la scelta della piattaforma: Windows, Linux, BSD oppure Macintosh. In secondo luogo é necessario prestare attenzione alla licenza: freeware oppure commerciale.

Questo capitolo presenta un elenco dei software provati, con riferimento ai due soli sistemi operativi a disposizione: Windows e Linux. Per gli applicativi commerciali é stato necessario usufruire delle versioni demo a 30 giorni o a servizio limitato, tuttavia sufficienti per i nostri scopi; viceversa per quelle freeware non si é presentato tale problema.

4.1 Piattaforma Windows

In casa Microsoft esistono diversi applicativi che svolgono la funzione di monitoraggio. I piú diffusi sono:

- OpManager [10] pure essendo un applicativo per windows utilizza strumenti adatti al mondo linux come apache e mysql. La configurazione facilissima. Si lancia l'exe e tutto funzia
- SNMPc [11] molto scarno e difficile da configurare
- Axence NetTools [12] molto scarno semplice da configurare
- Netmon [13] risulta molto semplice intuitivo. provata la demo. per il trial vuole che lo contatto
- InterMapper [14] disponibili anche per moltissime distribuzioni linux. necessita di scaricare tre file InterMapper, RemoteAccess DataCenter. molta documentazione. InterMapper molto intuitiva da usare. molto soddisfacente.
- Mon.itor.us [15] utilizza una piattarma web. si installa gli agent su ogni macchina windows. ancora in versione beta. presto esce anche per linux

4.2 Piattaforma Linux

Nel mondo Linux la scelta dell'applicativo per il monitoraggio non presenta particolari compromessi rispetto al concorrente Microsoft. I software in circolazione sono abbastanza numerosi e validi. Inoltre questi prodotti open-source hanno un alto grado di flessibilitá in modo da poter personalizzare la configurazione dell'applicatico in ragione alle proprie esigenze.

Esistono svariati applicativi, piú o meno diffusi, tuttavia i strumenti sui quali ho deciso di implementare il protocollo SNMP sono:

- NAGIOS
- CACTI

La scelta di questi due software, a differenza di altri come OpenNMS [16], converge sulla semplicitá di integrazione di questi due strumenti. Inoltre hanno entrambi una comunitá molto attiva che prontamente risponde a ogni sorta di problema segnalato. Oltre tutto ambedue subiscono rempentini cambiementi di versione e possiedono numerosi plugin a dimostrazione di come questi progetti stanno effettivamente sviluppandosi. Applicativi come MRTG [17] e Munin [18] non hanno ancora raggiunto questo livello poiché non risultano avere lo stesso dinamismo e supporto dei precedenti.

Ecco dunque le motivazioni che hanno fatto ricadere la scelta su questi due software rispetto a quelli presenti nel mondo open-source.

4.3 Preferenza

Dopo aver esaminato analizzato software che girano su i due diversi sistemi operativi, la scelta é ricaduta su Linux. Le motivazioni di questa decisione sono da ricondursi agli stessi numerosi vantaggi che si ricava da un prodotto non commerciale e freeware:

- Costo economico praticamente nullo;
- Ottima qualitá del codice;
- Ampio supporto dalla comunitá di sviluppo;
- Alto grado di flessibilitá;

Si fa presto a capire come questi fattori siano risultati determinanti sulla scelta. Al fine di utilizzare questi software é necessario tuttavia l'installazione e la configurazione di una struttura che permetta il corretto funzionamento di NAGIOS e CACTI. L'archietettura che andremo ad utilizzare prende il nome di LAMP, Linux Apache Mysql Php. Ovviamente la scelta della distribuzione é poco rilevante, poiché qualsiasi distribuzione Linux si presta facilmente all'installazione di questo infrastruttura.

Nel nostro caso la scelta é ricaduta su Fedora 8[**?**], un sistema operativo open-source basato su Red Hat[**?**] indubbiamente una tra le distribuzioni lato server piú utilizzate. Nel prossimo capitolo viene quindi illustrata l'installazione di un sistema LAMP su Fedora 8.

Capitolo 5

LAMP

A partire da questo capitolo e i successivi, la tesi assume un carattere molto piú pratico. Vediamo infatti come installare un sistema di tipo LAMP su una distribuzione Linux, nel nostro caso Fedora 8. Per prima cosa é fondamentale procurarsi una ISO[1] direttamente dal sito ufficiale di Fedora[27]. Dopo averne fatto una copia su un supporto DVD, si riavvia la macchina e si imposta il BIOS alla lettura del supporto magnetico. Normalmente é possibile accedere al BIOS digitando F12 nell'instante in cui si avvia la macchina. Dopo aver fatto questi opportuni settaggi si procede con l'installazione di Fedora 8.

L'installazione avviene tramite interfaccia grafica come in figura 5.1, quindi non presenta particolari difficoltá. Un accorgimento banale, ma non meno importante é che l'installazione di Fedora 8 sulla partizione di un sistema operativo ovviamente riscriverá tale partizione con la perdita totale dei dati presenti in quel settore del disco fisso. Dunque é buona norma installare Fedora 8 su una macchina dove non siano presenti dati sensi per non incorrere in spiacevoli conseguenze.

Terminata l'installazione al riavvio della macchina il nuovo sistema operativo sará pronto per essere utilizzato e si presenta come in figura 5.2.

5.1 Installazione e configurazione su un sistema Linux

Dopo aver installato Fedora 8, iniziamo ad installare il primo pacchetto del sistema LAMP: Apache. Apache [28] é il server web di linux che ci permette di caricare sul proprio computer pagine html che potranno poi essere visualizzate tramite un browser da tutti gli host all'interno della rete. Per installare Apache utilizzeremo il comando *yum*, il gestore dei pacchetti per tutti sistemi basati su Red Hat. Quindi di seguito sono riportati i comandi per l'autenticazione in modalitá root [2]

```
\$ su -
# yum install httpd
```

[1] ISO *estensione di tipo filesystem secondo lo standard ISO 9660*

[2] root *superutente nei sistemi linux con privilegi di amministratore*

Figura 5.1: Installazione Fedora

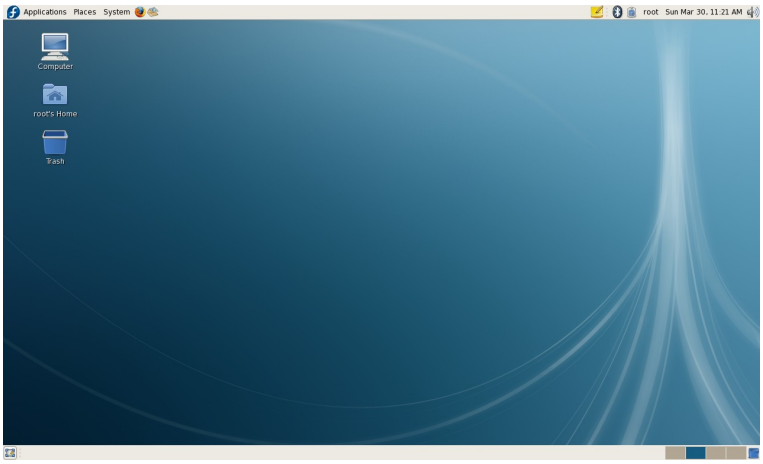

Figura 5.2: Primo avvio di Fedora 8

Fatto questo passiamo all'installazione di MySQL[20]. MySQL Ã¨ un Database management system (DBMS) relazionale, composto da un client con interfaccia a caratteri e un server, entrambi disponibili sia per sistemi Unix come GNU/Linux che per Windows, anche se prevale un suo utilizzo in ambito Unix. Questo pacchetto consente la creazione di un database per il salvataggio di tutti i dati. Anche in questo caso l'installazione risulta essere molto semplice:

```
# yum install mysql && mysql-server
```

Dopo aver installato il pacchetto mysql é buona norma impostare una password per l'amministratore di sistema. Questa esigenza é facilmente risolvibile tramite i seguenti comandi dove al posto di *rootpw* inseriamo la nostra password:

```
#/usr/bin/mysql
mysql> set password for root@localhost=password('rootpw');
mysql> exit
```

Terminata questa procedura si passa all'installazione di PHP[3], in particolare dell'ultima versione, la cinque. PHP é un linguaggio di programmazione web al fine di creare pagine web dinamiche. Questo pacchetto consente anche l'esecuzione di script, ossia piccoli programmi a linguaggio interpretato con l'utilizzo spesso di programmi esterni. Anche in questo caso l'esecuzione del comando é molto semplice, grazie all'utilizzo di *yum*:

```
# yum install php
# yum install php-mysql
# yum install php-snmp
```

Per i nostri fini, poiché esula dalla'archietettura LAMP, dobbiamo installare il pacchetto NMAP, il software piú conosciuto di port scanning che ci consentirá di effettuare il discovery della rete.

```
# yum install nmap
```

Per completare la configurazione, anche se non strettamente necessari per il funzionamento dell'archietettura LAMP, ma sicuramenti utili possiamo installare altri due pacchetti. Il primo é l'interfaccia grafica di *yum* e prende il nome di *yumex*. Possiamo avere una sua rappresentazione grafica come in fugura 5.3.

Il secondo é invece l'interfaccia grafica per MySQL e risponde al nome di phpMyAdmin[24]. Anche qui possiamo avere una rappresentazione grafica come in figura *fig:phpmyadmin*

Per installare queste interfaccie grafiche che sicuramente ci aiutano nella configurazione del sistema, basta digitare i seguenti comandi:

```
# yum install yumex
# yum install phpMyAdmin
```

[3] PHP *Hypertext Preprocessor - processore di ipertesti*

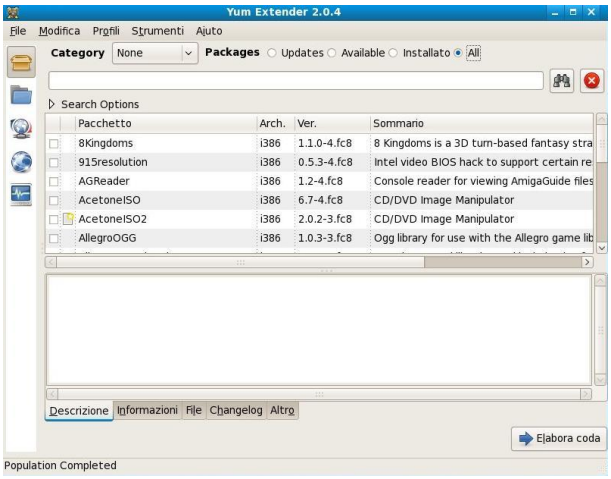

Figura 5.3: Yumex: interfaccia grafica a yum

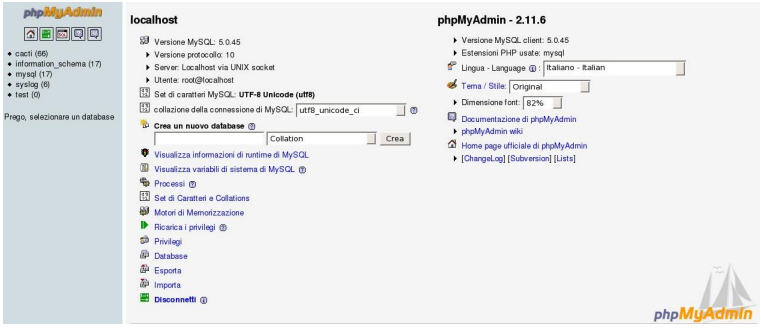

Figura 5.4: phpMyAdmin: interfaccia grafica a MySQL

5.1.1 Esecuzione automatica all'avvio

Dopo aver installato tutti i pacchetti necesseri per la nostro sistema LAMP, é necessario eseguire i servizi. In nostro aiuto viene la directory */etc/init.d* nella quale sono presenti tutti gli script d'avvio per questi servizi; basterá semplicemente richiamare lo script relativo al servizio all'interno di questa directory per avviare il demone. Nei nostri casi i comandi da lanciare sono:

```
#/etc/init.d/httpd start
#/etc/init.d/mysqld start
```

Ovviamente questi servizi non partono di default all'avvio del sistema operativo, é quindi necessario inserire uno script all'interno di */etc/rc* in modo che possa essere eseguito ogni volta all'avvio. Per fare questa piccola modica basta semplicemente modificare il file */etc/rc* con editor di testo. Nel mio caso utilizzo *nano*.

```
if [ -x /etc/init.d/mysqld ]; then
   echo -n " Start MySQL "
   /etc/init.d/mysqld start
fi

if [ -x /etc/init.d/httpd ]; then
   echo -n " Start Apache "
   /etc/init.d/httpd start
fi
```

Capitolo 6

Nagios

NAGIOS é un'applicazione open source per il monitoraggio di computer e risorse di rete, presente dal 1999. Il suo compito é quello di controllare lo stato dei servizi specificati, come processi attivi, spazio libero sul disco, server web, ftp, dhcp, avvertendo l'amministratore quando questi risultano non essere perfettamente funzionanti o in una condizione critica.

In origine, NAGIOS é stato sviluppato per sistemi operativi Linux like, ma ora é in grado di funzionare correttamente anche su altri sistemi operativi. Questo applicativo nasce come successore di NetSaint e il suo acronimo significa *NAGIOS Ain't Gonna Insist On Sainthood* ossia, *non insisterá piú sulla sanitá*, marcando appunto la diversitá dal suo predecessore.

Inoltre NAGIOS, é lasciato sotto licenza GPL[36], *General Public License* permettendo una serie di vantaggi che stanno alla base del reale successo di questo applicativo. Tali punti di forza sono espressi dalle seguenti caratteristiche:

- *modularitá* possibilitá di configurazione basata su modoli;
- *scalabilitá* possibilitá di piú server NAGIOS su un'unica interfaccia;
- *versatilitá* possibilitá di personalizzare la propria configurazione tramite plugin;
- *interfaccia web* possibilitá di visualizzazione dell'andamento delle prestazioni, dello storico dei log e report consultabili via browser.

Il demone di NAGIOS effettua controlli periodici su host e servizi specificati, attraverso l'uso plugin esterni che rispondono alle richieste di informazioni di stato. Quando si riscontrano dei problemi, il demone é in grado di inviare notifiche ai contatti amministrativi in diverse modalitá: e-mail, messaggi istantanei e SMS.

Infine i file sorgenti per l'installazione sono presenti nel sito ufficiale di NAGIOS[26] dove é possibile trovare tutta la documentazione necessaria al corretto funzionamento. Nel prossimo capitolo si affronterá proprio questa parte.

6.1 Installazione

La procedura per l'installazione é ampiamente documentata presso il sito ufficiale di NAGIOS[25]. Si prende come punto questa documentazione proprio per avere una configurazione standard compatibile con la comunitá per delle future modifiche. Questa parte del capito si discute quelli che sono gli aspetti mancanti per completare l'installazione di questo applicativo sul nostro server. Per completare l'installazione sono necessari i seguenti pacchetti *gcc glibc glibc-common gd gd-devel*. Procediamo quindi con l'installazione:

```
# yum install gcc
# yum install glibc glibc-common
# yum install gd gd-devel
```

Completato questo comando il prossimo passo é quello di creare un utente e un gruppo specifico per NAGIOS

```
# /usr/sbin/useradd nagios
# passwd nagios
```

Si crea un nuovo gruppo chiamato *nagcmd*, per permettere l'esecuzione di comandi esterni che sono visualizzabili tramite l'interfaccia web. Si aggiunge entrambi l'utente di nagios e l'utente apache al gruppo appena creato.

```
# /usr/sbin/groupadd nagcmd
# /usr/sbin/usermod -G nagcmd nagios
# /usr/sbin/usermod -G nagcmd apache
```

Si crea una directory nella quale tramite il comando *wget* si effetta il download dell'eseguibile di NAGIOS e dei plug-in.

```
# mkdir ~/downloads
# cd ~/downloads
# wget http://osdn.dl.sourceforge.net/sourceforge/nagios/nagios-3.0.2.tar.gz
# wget http://osdn.dl.sourceforge.net/sourceforge/nagiosplug/nagios-plugins
-1.4.11.tar.gz
```

Poi si passa all'estrazione del file compresso, per poi recarsi dentro la cartella di NAGIOS.

```
# cd ~/downloads
# tar xzf nagios-3.0.2.tar.gz
# cd nagios-3.0.2
```

Quindi si passa alla compilazione e l'installazione.

```
# ./configure --with-command-group=nagcmd
# make all
# make install
# make install-init
# make install-config
# make install-commandmode
# make install-webconf
```

entra nella directory

```
# cd ~/downloads
# tar zxvf nagios-plugins-1.4.11.tar.gz
# cd nagios-plugins-1.4.11
```

Installazione PLUGIN

```
# ./configure --with-nagios-user=nagios --with-nagios-group=nagios
# make
# make install
```

Nel momento in cui si avvia l'interfaccia web di NAGIOS per la prima volta e non si hanno le corrette credenziali di accesso, é possibile tramite questo comando impostare di nuovo una nuova password per l'utente nagiosadmin

```
# htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin
```

E' buona norma verificare la configurazione di NAGIOS con il comando

```
# /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg
```

Dopo di che é possibile avviare il servizio

```
service httpd restart
```

6.2 Struttura

La struttura di NAGIOS segue una chiara e ben definata logica. I file di configurazione e gli script si trovano all'interno della directory */usr/local/nagios/etc/*. In questa directory possiamo trovare il file *nagios.cfg* nel quale sono presenti tutte le opzioni per la configurazione di NAGIOS. Nella directory */usr/local/nagios/etc/objects/* troviamo i principali file di configurazione:

- *commands.cfg* definisce i comandi da utilizzare nei servizi;
- *templates.cfg* definisce il modello per la configurazione di un oggetto facendo riferimento agli host, servizi e contatti definiti in altri file;
- *contacts.cfg* definisce i contatti necessari per le notifiche;
- *timeperiods.cfg* definisce i periodi di tempo;
- *localhost.cfg* definisce il computer locale da monitorare;

Questi file rispecchiano a grandi linee la struttura attraverso la quale NAGIOS lavora. Infatti la sudddivisione in diversi file é solamente per facilitarne la comprensione: si puó benissimo utilizzare un unico file. NAGIOS, nell'instante in cui si avvia, genera un file chiamato *object.cache* nella directory *var* dove raggruppa tutti i file di configurazione in un unico blocco. Vediamo infatti che NAGIOS rispetta questa struttura principale:

- Contatti

- Host
- Gruppi
- Servizi
- Time Period

Nei prossimi capitoli affronteremo questi elementi singolarmente, descrivendo in maniera piú dettagliata i loro specifici compiti.

6.2.1 Contatti

La definizione dei contatti si basa su due fasi: per primo si descrive un contatto, inteso come persona fisica identificata da un nome e da, ad esempio, l'indirizzo mail sul quale inviare le notifiche. Si puó anche definire la modalitÃ di notifica. In seguito va descritto un contact-groups, in altre parole un gruppo cui i contatti definiti in precedenza faranno parte. In questo modo le notifiche, se specificato, saranno inviate al gruppo (inteso come insieme di contatti).

6.2.2 Host

Occorre definire un template comune a tutti gli host, il che significa che tutti quelli che utilizzeranno questa definizione avranno le stesse caratteristiche basilari. Per ogni computer in rete da monitorare dovremo inserire una definizione con i seguenti parametri:

- use il template da usare;
- host name un nome con il quale sará identificato all'interno di tutti i file di configurazione di NAGIOS;
- alias un alias che descrive brevemente l'host;
- check_command un comando che NAGIOS utilizzerá per determinare lo stato dell'host;
- max_check_attempts il numero massimo di tentativi consecutivi da compiere prima della notifica d'irraggiungibilitá di un host;
- notification_interval l'intervallo di tempo da aspettare, in minuti, prima di inviare una notifica;
- notification_options quando é possibile inviare notifiche, secondo il file di configurazione timeperiods.cfg.

I tipi di notifiche sono definiti come:

- d Host in stato down (non risponde);
- u Host in stato di unreacheable (non puó essere raggiunto);
- r Host in stato di recovery (la situazione ritorna alla normalitá dopo che l'host é stato o down o unreachable).

NAGIOS mette a disposizione dell'utente anche la possibilitÃ di raggruppare piú host insieme e di vederne una visuale completa tramite tre viste, Hostgroup Overview, Hostgroup Summary e Hostgroup Grid. La definizione di un gruppo e dei propri membri ha bisogno di un nome(hostgroup_name), di una breve descrizione (alias), e dell'elenco degli host membri separati da una virgola (members).

6.2.3 Comandi

La definizione di un comando esige un nome (che sarÃ richiamato dalla definizione successiva di un servizio, command_name) e dalla linea di comando del plug-in utilizzato (ricordiamo che la variabile $USER1$ é descritta in resources.cfg) con le opzioni necessarie. I parametri passati avranno una numerazione incrementale ($ARG1$,$ARG2$,...) mentre la variabile che indica l'indirizzo Ip sul quale eseguire il controllo Ã¨ fissa (chiamata $HOSTADDRESS$).

6.2.4 Gruppi

La definizione di un gruppo permette di creare un insieme di contatti con le stesse caratteristiche e modalitá di notifica. La sintassi é la seguente:

```
define contactgroup{
        contactgroup_name       admins
        alias                   Nagios Administrators
        members                 nagiosadmin
        }
```

6.2.5 Servizi

Occorre definire un template comune a tutti i servizi, in cui specifichiamo, ad esempio, di abilitare la notifica in caso di problemi (notifications_enabled). Inoltre bisogna inserire una definizione per ogni servizio da utilizzare. Questa dovrá contenere una descrizione, l'host verso il quale eseguire il servizio, il contactgroup per le notifiche e, non per ultimo, il comando definito in check_commands da usare. I parametri sono passati con la sintassi: *nome_plugin!argomento1!..argomento2!...!argomentoN*

6.2.6 Time Period

Un time period é una descrizione che indica quando é possibile inviare notifiche. Ad esempio, posso specificare che le notifiche debbano essere inviate solo durante la notte, o solo in determinati giorni. Ogni periodo ha un nome (ad esempio 24x7) , una breve descrizione (alias) e un elenco dei giorni e degli orari ammessi.

6.3 Nmap2nagios

Dopo aver trattato ampiamente ogni elemento che compone l'infrastrutta di NAGIOS, é giunto il momento di scrivere tutti i file di configurazione per monitorare la nostra rete. Dunque é necessario per ciascun device inserire un template, un gruppo, gestire i servizi e i contatti. Questo lavoro sarebbe spaventosamente

lungo e noiso, soprattutto come nel nostro caso se la rete Ã¨ composta da centinaia di host. In nostro soccorso viene nmap2nagios[29], un plug-in per NAGIOS che permette di risolvere il nostro problema. Questo plug-in lavora effetuando una scansione della intera rete, riconoscere le porte aperte attraverso le quali ne determina i servizi remoti e infine genera un sorgente pronto per essere utilizzato da NAGIOS. In questo modo é possibile automatizzare il processo di configurazione di NAGIOS.

Questo plugin sfrutta il lavoro di nmap per effettuare il discovery della rete, il quale restituisce come output un file in xml. Il comando ´Ã¨

```
# sudo nmap -sS -O -oX rete.xml 193.205.92.1-100
```

L'opzione sS indica la scansione di tipo SYN scan. Questa é la piú usata per buone ragioni. Puó essere compiuta velocemente: effettua la scansione su migliaia di porte al secondo su una rete veloce non limitata da firewall intrisivi. La SYN scan é relativamente poco invasiva e nascosta. Inoltre permette una differenziazione chiara e affidabile tra le porte appartenenti agli stati open, closed, e filtered. L'opzione O abilita l'OS detection, ossia individua il sistema operativo presente sulla macchina remota. In alternativa, é possibile utilizzare l'opzione A per attivare sia l'OS detection sia il version detection. Infine l'opzione oX indica il percorso dove salvare il file XML.

Nell'esempio é ispezionato un intervallo di cinque host. Per esaminare tutta la rete é sufficiente sostituire l'intervallo degli indirizzi con l'indirizzo di rete e la maschera opportuna: 193.205.92.0/24. Tuttavia per conoscere tutte le opzioni si puó consultare il manuale[30].

Una volta ottenuto il file XML (nell'esempio rete.xml), é possibile aprirlo con il browser firefox presente su Ubuntu. Il browser, infatti, interpreterá i vari tag del file XML visualizzando in modo schematico gli indirizzi IP delle macchine remote con i loro relativi servizi. A questo punto, si esegue nmap2nagios. Come input gli viene dato il file XML appena creato e nmap2nagios si assumerá dunque lonere di generare un sorgente ad hoc per Nagios. Per far questo, occorre collocarsi nella directory in cui é presente lo script nmap2nagios.pl e si esegue con il seguente comando:

```
#./nmap2nagios.pl -v -r rete.xml -o /usr/local/nagios/etc/objects/rete.cfg
```

L'opzione v indica il verbose. L'opzione r indica il percorso del file di input. Infine l'opzione o indica la destinazione del file sorgente di NAGIOS. Naturalmente (come indicato nell'esempio) il file sorgente di NAGIOS andrá salvato sotto la directory *usr/local/nagios/etc/objects/*. Ora, non rimane che riavviare NAGIOS in modo da aggiornare il servizio con i nuovi file sorgenti appena creati. Il comando é il seguente:

```
# sudo /etc/init.d/nagios restart
```

Se la shell non restituisce nessun messaggio di tipo error, allora indica che la nuova configurazione é andata a buon fine. Sará dunque possibile monitorare la presenza dei nuovi host attraverso l'interfaccia web, collegandosi con il browser all'indirizzo *http://localhost/nagios* oppure tramite il comando da shell:

```
# firefox http://localhost/nagios
```

Una volta inserite le credenziali d'accesso, si avrÃ la possibilitá di visualizzare gli host monitorati.

6.3.1 Modifiche e Settaggi

Per far funzionare lo script nmap2nagios occorre modificare il file */urs/local/nagios/etc/objects/commands.cfg* inserendo i comandi *check_mysql, check_dns* e *check_telnet* non definiti per default:

```
# 'check_mysql' command definition
define command{
        command_name    check_mysql
        command_line    $USER1$/check_tcp -H $HOSTADDRESS$ -P 3306
        }

# 'check_dns' command definition
define command{
        command_name    check_dns
        command_line    $USER1$/check_dns -H $HOSTADDRESS$
        }
# 'check_telnet' command definition
define command{
        command_name    check_telnet
        command_line    $USER1$/check_tcp -H $HOSTADDRESS$ -p 23
        }
```

6.4 Monitoraggio Remoto

La vera funzionalitá di applicativo di monitoraggio come NAGIOS, oltre a quella di controllare lo stato dei servizi delle macchine presenti nella LAN, é anche quello di controllare lo stato della macchina stessa. Con questa semplice configurazione non é infatti possibile determinare il carico della CPU, lo spazio sul disco e quant'altro. Per risolvere questo genere di problemi esistono due diversi plugin:

- check_netsnmp un plugin che si interfaccia con l'applicativo net-snmp
- check_snmp un plugin che si interfaccia con il protocollo SNMP
- check_nrpe un plugin che si interfaccia l'agente NRPE

Questi plugin sono presenti all'interno della directory */usr/local/nagios/libexec/* Si discuterá nei prossimi capitoli questi plugin nel dettaglio.

6.4.1 check_snmp

Questo plugin presente nella suite dei plugin di default consente di fare delle interrogazioni SNMP agli host remoti, passandogli specifici OID. Se volessimo interrogare la media della temperatura della CPU di un sistema Microsoft e sappiamo che questo valore corrisponde all'OID .1.3.6.1.2.1.25.3.3.1.2.7. Controlliamo che nel nostro commands.cfg compaia:

```
# 'check_snmp' command definition
define command{
```

```
        command_name    check_snmp
        command_line    $USER1$/check_snmp -H $HOSTADDRESS$ -C public $ARG1$
        }
```

Dopo di che sará sufficiente inserire un servizio sull'host che vogliamo monitorare indicando quello specifico OID.

```
define service {
  host_name             193-205-92-4.mhsnetworks.net
  service_description   Load CPU
  check_command   check_snmp!.1.3.6.1.2.1.25.3.3.1.2.7
  max_check_attempts    3
  normal_check_interval 5
  retry_check_interval  1
  check_period          24x7
  notification_interval 5
  notification_period   24x7
  notification_options  c,r
  contact_groups        admins
}
```

6.4.2 check_netsnmp

Questo plugin presente nella suite dei plugin di default consente delle interrogazioni SNMP agli host remoti.

```
# cd /usr/local/nagios/libexec/
# wget http://mitya.pp.ru/check_netsnmp
# chmod ugo+x check_netsnmp
```

Modificando opportunamente il fila snmpd.conf presente nella directory */etc/snmp/* si puó utilizzare questo plugin per interfacciarsi con NAGIOS e restituire i valori tramite il protocollo SNMP

6.4.3 check_nrpe

Questo non Ã¨ un vero proprio plugin, ma é un agente che permette il monitoraggio di host remoti, tramite semplici parametri di configurazione. Questo plugin non sfrutta il protocollo SNMP. Su ogni macchina remota viene installato un agente che viene poi interrogato da NAGIOS per avere le informazione.

```
command[check_users]=/usr/local/nagios/libexec/check_users -w 5 -c 10
command[check_load]=/usr/local/nagios/libexec/check_load -w 15,10,5 -c 30,25,20
command[check_hda1]=/usr/local/nagios/libexec/check_disk -w 20% -c 10% -p /dev/hda1
command[check_zombie_procs]=/usr/local/nagios/libexec/check_procs -w 5 -c 10 -s Z
command[check_total_procs]=/usr/local/nagios/libexec/check_procs -w 150 -c 200
```

6.5 Panoramica plugins

In questo paragrafo si analizzano i principali plugin di NAGIOS. Per ognuno di essi si evidenziano le caratterestiche principali, i parametri e un esempio d'uso.

- **check_disk** -w <wlim>% -c <clim>% [-p <path>] [-l] [-M] [-e] [-u <units>]

  controlla la capacitá del disco usato su un qualsiasi file system montato e segnala un errore se il valore é al di sotto di una percentuale definita come soglia. Anche la memoria ram */dev/shm* rientra tra i file system controllati.

  - w <wlim>% percentuale di spazio libero sotto la quale viene comunicato un WARNING
  - w <wlim> numero di unitá di spazio libero al di sotto della quale viene comunicato WARNING
  - c <clim>% percentuale di spazio libero al di sotto della quale viene generato un errore di tipo CRITICAL
  - c <clim> numero di unitá di spazio libero al di sotto della quale viene generato un errore di tipo CRITICAL
  - p <path> percorso della cartella o partizione da controllare
  - l controlla solo i file system locali e non quelli remoti
  - M visualizza anche i mount point
  - e visualizza solo i device/mountpoint con errori
  - u <units> specifica le unitÃ da utilizzare; di default Ã¨ MB, ma si puó scegliere anche kB, GB, TB.

  Esempio d'uso:

  ```
  check_disk -w 10% -c 5% -p /tmp -p /var -w 100000 -c 50000 -p /
  ```

  Controlla le cartelle /tmp e /var al 10 e 5 % e la directory radice / a 100 MB, 50 MB.

- **check_swap** -w <warn> -c <crit> [-a]

  controlla lo spazio libero sulla partizione swap presente sul computer locale; genera un errore se il valore é minore di una percentuale (o di un intero) definita come parametro.

  - w <warn>% percentuale di spazio libero al di sotto della quale il plugin genera un errore di tipo WARNING
  - w <warn> numero intero di bytes di spazio libero al di sotto del quale il plugin genera un errore di tipo WARNING
  - c <crit>% percentuale di spazio libero al di sotto della quale il plugin genera un errore di tipo WARNING
  - c <crit> numero intero di bytes di spazio libero al di sotto del quale il plugin genera un errore di tipo WARNING
  - a fa una comparazione per tutte le partizioni swap, una per una

  Esempio d'uso:

  ```
  check_swap -w 10% -c 5%
  ```

- **check_users** -w <warn> -c <crit>

  controlla il numero di utenti correntemente collegati al sistema locale e genera un errore se il valore eccede le soglie specificate.

  - w <warn> numero intero di utenti al di sopra del quale il plugin genera un errore di tipo WARNING
  - c <crit> numero intero di utenti al di sopra del quale il plugin genera un errore di tipo CRITICAL

  Esempio d'uso:

  ```
  check_users -w 5 -c 10
  ```

- **check_icmp** -[wcni] -H <hostaddress>

  controlla l'host specificato inviando pacchetti icmp; genera errore se il ritardo (RTA) della risposta o la percentuale di pacchetti persi (packet loss) sono maggiori di valori dati. Il protocollo ICMP (Internet Control Message Protocol) é usato dai router per segnalare eventi inattesi. Viene anche utilizzato per testare la rete: per esempio viene usato da comando PING (Racket Internet Groper) per verificare se é possibile comunicare con un host.

  - H indirizzo IP dell'host da controllare
  - w soglia critica al di lá della quale il plugin genera un errore di tipo WARNING; puó essere espressa in millisecondi (di default 200.0ms) o in percentuale di pacchetti persi (dei default 40%)
  - c soglia critica al di lá della quale il plugin genera un errore di tipo CRITICAL; puó essere espressa in millisecondi (di default 500.0ms) o in percentuale di pacchetti persi (dei default 80%)
  - n numero di pacchetti da inviare (di default 5)
  - i intervallo di tempo massimo tra i pacchetti (di default 80.0ms). Le unitá di misura del ritardo possono essere di microsecondi (us), millisecondi (ms) e secondi (s).

  Esempio d'uso

  ```
  check_icmp -w 10% -c 30% -H 192.168.1.5
  ```

- **check_dhcp** [-r <requestedip>] [-i <interface>] [-t <timeout>] -s <serverip>

  Questo plugin controlla la disponibilitá di un server DHCP su una rete. Genera errori se il server non é raggiungibile, o se il messaggio *DHCPOFFER* arriva dopo un ritardo superiore ad un certo timeout specificato. Il protocollo DHCP (Dynamic Host Configuration Protocol) é un sistema di tipo client/server per la configurazione automatica e dinamica degli host. Il server viene configurato con degli intervalli di indirizzi IP (scope) che puó assegnare. Oltre all'indirizzo IP puó fornire al client anche altre informazioni di configurazione come la machera di sottorete, il gateway o il server DNS.

- s <serverip> indirizzo IP del server DHCP da controllare
- r <requestedip> indirizzo IP che dovrebbe essere offerto dal server
- i <interface> interfaccia di rete da usare per il controllo (ad esempio eth0)
- t <timeout> intervallo di tempo massimo (in secondi) da aspettare per l'arrivo di un messaggio DHCPOFFER

Esempio d'uso:

```
check_dhcp -i eth0 -t 20 -s 192.168.1.10
```

- **check_dns** -H <host> [-s <server>] [-a <expected-address>] [-A] [-t <timeout>]

  Questo plugin usa il programma nslookup per ottenere l'indirizzo IP del dominio/host specificato. Puó essere usato un server DNS opzionale, ma se non é specificato, viene usato il server (o i server) DNS di default presenti nel file /etc/resolv.conf. Nslookup é un programma per le richieste DNS presente in qualsiasi distribuzione Linux.

  - H <host> l'indirizzo stringa che si vuole controllare (ad esempio www.unicam.it)
  - s <server> Server DNS opzionale con cui si vuole controllare (al posto di quello di default)
  - a <expected-address> Indirizzo IP opzionale che ci si aspetta dal server DNS come risposta (dopo aver risolto la corrispondenza nome-indirizzo)
  - t *timeout* Secondi prima che la connessione venga chiusa (timeout), di default é 10

  Viene generato errore se non é possibile risolvere il nome o se il timeout scade.
  Esempio d'uso:

```
check_dns -H www.google.it -s 192.168.12.2 -t 20
```

  Controlla il server DNS 192.168.12.2 con l'indirizzo www.google.it e impone un timeout di 20 secondi.

- **check_ssh** [-46] [-t <timeout>] [-r <remote version>] [-p <port>] [-H] <host>

  Prova a connettersi ad un server SSH attraverso un indirizzo e una porta specificati.

  - H <host> Nome dell'host o indirizzo IP
  - p <port> Numero della porta da utilizzare (di default: 22)
  - 4 Usa una connessione IPv4
  - 6 Usa una connessione IPv6

  - t <timeout> Secondi prima che la connessione venga chiusa (timeout), di default é 10
  - r <remote version> Genera un errore di tipo WARNING se la versione del server SSH non coincide con la stringa (ad esempio: OpenSSH_3.9p1)

  Esempio d'uso:

  ```
  check_ssh -H 192.168.12.5
  ```

- **check_smtp** -H host [-p <port>] [-e <expect>] [-C <command>] [-R <response>] [-f <from addr>] [-w <warn>] [-c <crit>] [-t <timeout>] [-n] [-4—-6]

  Questo plugin cerca di aprire una connessione SMTP con un host specificato. Una connessione con successo ritorna un stato OK, una connessione rifiutata o che ha fatto scadere il timeout dá uno stato CRITICAL, gli altri errori uno stato UNKNOWN. Una connessione con successo ma una risposta non corretta dall'host controllato ritorna uno stato WARNING. Il protocollo SMTP (Simple Mail Transfer Protocol) é utilizzato per inviare la posta attraverso un server SMTP. Il server copia i messaggi nelle appropriate caselle postali; se non é possibile viene spedita una notifica di errore al mittente.

  - H <host> Nome dell'host o indirizzo IP
  - p <port> Numero della porta da utilizzare (di default: 25)
  - 4 Usa una connessione IPv4
  - 6 Usa una connessione IPv6
  - e <expect> Stringa da aspettarsi come prima linea della risposta del server (di default: 220)
  - n Sopprime i comandi SMTP command
  - C <command> Specifica un commando SMTP (puó essere usato ripetutamente)
  - R <response> Risposta che ci si aspetta al comando eseguito (puó essere usato ripetutamente)
  - f <from addr> FROM-address da includere nel comando MAIL, richiesto da Exchange 2000
  - w <warn> Tempo di risposta massimo (double, in secondi) al di lá del quale emette un errore di tipo WARNING
  - c <crit> Tempo di risposta massimo (double, in secondi) al di lá del quale emette un errore di tipo CRITICAL
  - t <timeout> Secondi prima che la connessione venga chiusa (timeout, di default é 10).

  Esempio d'uso:

  ```
  check_smtp -H 192.168.12.11
  ```

- **check_ftp** -H <host> -p <port> [-w <warning time>] [-c <critical time>] [-s <send string>] [-e <expect string>] [-q <quit string>] [-m <maximum bytes>] [-d <delay>] [-t <timeout seconds>] [-v] [-4—-6] [-S <use SSL>]

  Questo plugin testa una connessione FTP con l'host specificato. Il protocollo FTP (File Transfer Protocol) é utilizzato per la trasmissione o la ricezione di file e richiede che l'utente si colleghi con nome e password (anche in modo anonimo, con nome anonymous) per avere accesso al server. Un server FTP utilizza due porte: la 20 (dati) e la 21 (controllo).

  - H <host> Nome dell'host o indirizzo IP
  - p <port> Numero della porta da utilizzare (di default: nessuna)
  - 4 Usa una connessione IPv4
  - 6 Usa una connessione IPv6
  - s <send string> Stringa da inviare al server
  - e <expect string> Stringa da aspettarsi nella risposta del server
  - q <quit string> Stringa da inviare al server per iniziare una procedura di chiusura sicura della connessione
  - m <maximum bytes> Chiude la connessione se viene ricevuto un numero di bytes superiore a questo valore di soglia
  - d <delay> Seconti da aspettare tra l'invio di una stringa e la ricezione della risposta
  - S Usa SSL per la connessione
  - w <warning time> Tempo di risposta massimo (double, in secondi) al di ládel quale emette un errore di tipo WARNING
  - c <critical time> Tempo di risposta massimo (double, in secondi) al di lá del quale emette un errore di tipo CRITICAL
  - t <timeout> Secondi prima che la connessione venga chiusa (timeout, di default é 15).

  Esempio d'uso:

  ```
  check_ftp -H 192.168.12.240 -p 21
  ```

- **check_pop** -H <host> -p <port> [-w <warning time>] [-c <critical time>] [-s <send string>] [-e <expect string>] [-q <quit string>] [-m <maximum bytes>] [-d <delay>] [-t <timeout seconds>] [-4—-6] [-S <use SSL>]

  Questo plugin prova a generare una connessione POP verso l'host specificato. Se la connessione viene rifiutata genera un errore di tipo CRITICAL. Il protocollo POP (Post Office Protocol) Ã¨ utilizzato per comunicare con i server di posta; possiede comandi per l'autenticazione del client, per copiare i messaggi e per cancellarli. La sua peculiaritá é che i messaggi vengono copiati dalla casella di posta al computer dell'utente.

  - H <host> Nome dell'host o indirizzo IP

  - p <port> Numero della porta da utilizzare (di default nessuna, usualmente la 110)
  - 4 Usa una connessione IPv4
  - 6 Usa una connessione IPv6
  - s <send string> Stringa da inviare al server
  - e <expect string> Stringa da aspettarsi nella risposta del server
  - q <quit string> Stringa da inviare al server per iniziare una procedura di chiusura sicura della connessione
  - m <maximum bytes> Chiude la connessione se viene ricevuto un numero di bytes superiore a questo valore di soglia
  - d <delay> Secondi da aspettare tra l'invio di una stringa e la ricezione della risposta
  - S Usa SSL per la connessione
  - w <warning time> Tempo di risposta massimo (double, in secondi) al di lá del quale emette un errore di tipo WARNING
  - c <critical time> Tempo di risposta massimo (double, in secondi) al di lá del quale emette un errore di tipo CRITICAL
  - t <timeout> Secondi prima che la connessione venga chiusa (timeout, di default é 10).

  Esempio d'uso:

  ```
  check_pop -H 192.168.12.5 -p 110
  ```

- **check_imap** -H <host> -p <port> [-w <warning time>] [-c <critical time>] [-s <send string>] [-e <expect string>] [-q <quit string>] [-m <maximum bytes>] [-d <delay>] -t <timeout seconds>] [-4—-6] [-S <use SSL>]

  Questo plugin prova a generare una connessione IMAP verso l'host specificato. Se la connessione viene rifiutata genera un errore di tipo CRITICAL. Il protocollo IMAP (Interactive Mail Access Protocol) é utilizzato per la lettura dei messaggi di posta come POP, ma permette di leggere la posta dal server senza copiarla sul computer dell'utente; in questo modo si puó accedere alla propria posta da computer diversi.

  - H <host> Nome dell'host o indirizzo IP
  - p <port> Numero della porta da utilizzare (di default: nessuna)
  - 4 Usa una connessione IPv4
  - 6 Usa una connessione IPv6
  - s <send string> Stringa da inviare al server
  - e <expect string> Stringa da aspettarsi nella risposta del server
  - q <quit string> Stringa da inviare al server per iniziare una procedura di chiusura sicura della connessione
  - m <maximum bytes> Chiude la connessione se viene ricevuto un numero di bytes superiore a questo valore di soglia

  - d <delay> Secondi da aspettare tra l'invio di una stringa e la ricezione della risposta
  - S Usa SSL per la connessione
  - w <warning time> Tempo di risposta massimo (double, in secondi) al di lá del quale emette un errore di tipo WARNING
  - c <critical time> Tempo di risposta massimo (double, in secondi) al di lá del quale emette un errore di tipo CRITICAL
  - t <timeout> Secondi prima che la connessione venga chiusa (timeout, di default é 10).

  Esempio d'uso:

  ```
  check_imap -H 192.168.12.12 -p 150
  ```

- **check_nagios** -F <status log file> -e <expire_minutes> -C <process_string>

  Questo plugin controlla lo stato del processo Nagios sul computer locale, tramite un check sullo status log; se é piú vecchio del numero di minuti specificati nell'opzione empire, genera un errore. Controlla anche che il processo sia attivo nella lista dei processi di sistema.

  - F <status log file> Nome del file log da controllare (percorso assoluto)
  - e <expire_minutes> Soglia in minuti dopo la quale il file log é considerato obsoleto (e quindi non aggiornato da Nagios)
  - C <process_string> Sottostringa da cercare nella lista dei processi

  Esempio d'uso:

  ```
  check_nagios -F /usr/local/nagios/var/status.log -e 5 -C /usr/local/nagios/bin/nagios
  ```

- **check_tcp** -H host -p port [-w <warning time>]

  Questo plugin genera e testa una connessione TCP con l'host specificato.

  - H <host> Nome dell'host o indirizzo IP
  - p <port> Numero della porta da utilizzare (di default: nessuna)
  - 4 Usa una connessione IPv4
  - 6 Usa una connessione IPv6
  - s <send string> Stringa da inviare al server
  - e <expect string> Stringa da aspettarsi nella risposta del server
  - q <quit string> Stringa da inviare al server per iniziare una procedura di chiusura sicura della connessione
  - m <maximum bytes> Chiude la connessione se viene ricevuto un numero di bytes superiore a questo valore di soglia
  - d <delay> Secondi da aspettare tra l'invio di una stringa e la ricezione della risposta
  - S Usa SSL per la connessione

  - w <warning time> Tempo di risposta massimo (double, in secondi) al di lá del quale emette un errore di tipo WARNING
  - c <critical time> Tempo di risposta massimo (double, in secondi) al di lá del quale emette un errore di tipo CRITICAL
  - t <timeout> Secondi prima che la connessione venga chiusa (timeout, di default é 10).

  Esempio d'uso:

```
check_tcp -H 192.168.12.1 -p 80
```

- **check_udp** -H host -p port [-w <warning time>] [-c <critical time>] [-s <send string>] [-e <expect string>] [-t <timeout seconds>]

  Questo plugin genera e testa una connessione UDP con l'host specificato.

  - H <host> Nome dell'host o indirizzo IP
  - p <port> Numero della porta da utilizzare (di default: nessuna)
  - s <send string> Stringa da inviare al server
  - e <expect string> Stringa da aspettarsi nella risposta del server
  - w <warning time> Tempo di risposta massimo (double, in secondi) al di lÃ del quale emette un errore di tipo WARNING
  - c <critical time> Tempo di risposta massimo (double, in secondi) al di lÃ del quale emette un errore di tipo CRITICAL
  - t <timeout> Secondi prima che la connessione venga chiusa (timeout, di default é 10).

  Esempio d'uso:

```
check_udp -H 192.168.12.1 -p 5000
```

- **check_time** -H <host_address> [-p <port>] [-u] [-w <variance>] [-c <variance>] [-W <connect_time>] [-C <connect_time>] [-t <timeout>]

  Questo plugin controlla l'orario di un host specificato.

  - H <host> Nome dell'host o indirizzo IP
  - p <port> Numero della porta da utilizzare (di default: 37)
  - u Usa il protocollo UDP per la connessione al posto del TCP
  - w <variance> Differenza di orario (in secondi) necessaria per generare un errore di tipo WARNING
  - c variance Differenza di orario (in secondi) necessaria per generare un errore di tipo CRITICAL
  - W <connect_time> Tempo di risposta necessario per generare un errore di tipo WARNING
  - C <connect_time> Tempo di risposta necessario per generare un errore di tipo CRITICAL

– t <timeout> Secondi prima che la connessione venga chiusa (timeout, di default é 10).

Esempio d'uso:

```
check_time -H 192.168.12.3 -w 30 -c 60
```

- **check_http** -H <vhost> — -I <IP-address> [-u <url>] [-p <port>] [-w <warn time>] [-c <critical time>] [-t <timeout>] [-a auth] [-A <useragent>] [-N] [-f <ok — warn — critical — follow>] [-e <expect>] [-4—-6] [-M <age>] [-T <con_type>] [-C <days>]

  Questo plugin testa il servizio HTTP su un host specificato. Puó controllare un web server normale (http) e sicuro (https), seguire ridirezioni, controllare i messaggi di risposta del server, controllare il tempo di connessione, testare certificati. Tenta di generare una connessione HTTP con l'host. Una connessione con successo genera uno stato OK, connessioni rifiutate o con timeout scaduto generano uno stato CRITICAL, gli altri errori uno stato UNKNOWN. Una connessione corretta, ma messaggi di risposta sbagliati dall'host generano uno stato WARNING. Il plugin puó anche testare se sia possibile una connessione tramite SSL, o controllare se il certificato X509 sia ancora valido per un numero specificato di giorni.

  – H <vhost> Parametro host name per i server che utilizzano virtual host. Se viene usata una porta diversa dalla 80, specificarlo subito dopo l'indirizzo (ad esempio example.com:5000).
  – I <IP-address> Indirizzo Ip o nome dell'host (utilizzare l'indirizzo numerico per evitare la richiesta al DNS)
  – p <port> Numero della porta (di default: 80)
  – 4 Utilizza il protocollo ipv4 per la connessione
  – 6 Utilizza il protocollo ipv6 per la connessione
  – S Connessione tramite SSL
  – C <days> Minimo numero di giorni affinché un certificato sia valido (quando l'opzione Ã¨ in uso non viene controllato l'url, ma solo il certificato).
  – e <expect> Stringa che ci si aspetta nella prima linea della risposta del server (default: HTTP/1.)
  – u <url> URL per GET o POST (di default: /)
  – N Non aspetta il corpo (body) del documento: smette di leggere dopo le intestazioni.
  – M <age> Genera un errore di tipo WARNING se il documento é piú vecchio di <age> secondi. Il numero puó anche essere nella forma 10m per i minuti, 10h per le ore e 10d per i giorni.
  – T <con_type> Specifica l'intestazione (header) Content-Type
  – a <username:password> Nome utente e password per i siti con basic authentication
  – A <useragent> Stringa da inviare nell'header http come User Agent

- f <ok—warning—critical—follow> Come comportarsi ad una ridirezione
- w <warning time> Tempo di risposta massimo (double, in secondi) al di lá del quale emette un errore di tipo WARNING
- c <critical time> Tempo di risposta massimo (double, in secondi) al di lá del quale emette un errore di tipo CRITICAL
- t <timeout> Secondi prima che la connessione venga chiusa (timeout, di default é 10).

Esempio d'uso:

```
check_http -w 5 -c 10 --ssl www.verisign.com
```

Quando il server www.verisign.com invia il proprio content in meno di 5 secondi, viene generato uno stato OK. Quando lo invia ma a tempo scaduto (in piú di 5 secondi) viene emesso un errore di tipo WARNING. Ad ogni altro errore, viene generato un errore di tipo CRITICAL. Nell'esempio inoltre viene utilizzata una connessione sicura tramite SSL.

```
check_http www.verisign.com -C 14
```

Quando il certificato di www.verisign.com é valido da piú di 14 giorni, viene generato uno stato OK. Quando il certificato é ancora valido, ma da meno di 14 giorni, viene emesso un errore WARNING. Se il certificato é scaduto, un errore di tipo CRITICAL.

- **check_ping** -H <host_address> -w <wrta>,<wpl>% -c <crta>,<cpl>% [-p <packets¿] [-t <timeout>] [-4—-6]

  Usa il comando ping per verificare le statistiche di connessione con un host remoto. Si puó utilizzare per controllare la percentuale di pacchetti persi o la media del round trip time (in millisecondi).

  - 4 Utilizza il protocollo ipv4 per la connessione
  - 6 Utilizza il protocollo ipv6 per la connessione
  - H <host-address> Indirizzo Ip o nome dell'host
  - w <wrta>,<wpl>% Coppia di soglie WARNING (vedi commento finale)
  - c <crta>,<cpl>% Coppia di soglie CRITICAL (vedi commento finale)
  - p <packets> Numero di pacchetti ICMP ECHO da inviare (Default: 5)
  - t <timeout> Secondi prima che la connessione venga chiusa (timeout, di default é 10).

  Le soglie sono nel formato <rta>,<pl>% dove <rta> é la media del round trip travel time (ms) e <pl> la percentuale di pacchetti persi (entrambi se superati generano errori CRITICAL o WARNING). Esempio d'uso:

  ```
  check_ping -H 192.168.12.1 -w 50,60% -c 60,80%
  ```

6.6 Notifiche e-mail

Il servizio principale di NAGIOS é quello di mostrare, attraverso l'interfaccia WEB, la consistenza della rete monitorata segnalando la presenza di un malfunzionamento mediante l'utilizzo di opportuni colorazioni e suoni . In ogni caso é indispensabile che l'amministratore di rete abbia l'interfaccia WEB di NAGIOS aperta e sia sempre in ascolto per rendersi immediatamente conto dello stato della rete.

Un modo per risolvere questo vincolo é la possibilitá di notifica email all'amministratore di rete. A questo proposito si sfrutta il lavoro fatto da *ssmtp*, un piccolo applicativo che consente l'invio di email mediate shell[1].

La notifica email puó essere effettuata con qualsiasi provider di posta, nel nostro caso abbiamo utilizzato quello di google, creando un account specifico per questo scopo. Di seguito riportiamo il comandi per l'installazione di *ssmtp*:

```
# yum install ssmtp
```

Ora é necessario configurare il file */etc/ssmtp/ssmtp.conf*, sostituendo tutto il suo contentuo con le seguenti righe:

```
mailhub=smtp.gmail.com:587
UseSTARTTLS=YES
AuthUser=utente (senza@gmail)
AuthPass=passdigmail (in chiaro)
FromLineOverride=YES
```

Terminata questa modifica, possiamo provare se l'*ssmtp* funziona correttamente. A tal fine cerchiamo di inviare una email mediante shell utilizzando questa sintassi:

```
#ssmtp email@destinatario
Subject: (L'oggetto del messaggio)
(invio a vuoto)
Testo del messaggio
(invio a vuoto)
ctrl-d (per spedire l'email)
```

Se controllando la casella di posta elettronica, l'email risulta essere arrivata possiamo sfruttare questo servizio per NAGIOS. Modifichiamo quindi il file */usr/local/nagios/etc/objects/contacts.cfg* decommentando la riga email e inserendo l'accont al quale vogliamo che vengano pervenute le notifiche:

```
define contact{
        contact_name                    nagiosadmin
        use                             generic-contact
        alias                           Nagios Admin
        email                           amministratoredirete@gmail.com
        }
```

[1]shell - *ambiente di lavoro attraverso il quale Ã¨ possibile impartire al computer comandi, richiedendo l'esecuzione di programmi*

In conclusione avremo un account di gmail che spedisce email di notifica alle case di posta degli amministratori di rete.

Capitolo 7

Cacti

Nel vasto modo delle reti degli elaboratori e dei prodotti open source, CACTI si presenta come una completa soluzione per il monitoraggio di rete. Questo software permette la visualizzazione dei dati tramite l'ausilio di grafici che lo rendono particolamente attraente agli occhi dell'amministratore di sistema, impaziente di conoscere cosa sta accadendo nella propria rete nel minor tempo possibile.

Questo software sfrutta il lavoro di RRDTool[33], uno strumento impiegato per gestire una serie temporali di dati come la larghezza di banda della rete, le temperature, il carico della CPU, e cosí via. I dati sono memorizzati secondo una schedulazione round-robin in un database, in modo tale che il periodo di attesa resti costante nel tempo. Questo tool é composto da una serie di librerie e funzioni in perl, python, ruby, tcl e php, realizzando in questo modo il motore grafico per CACTI. Tutto ció é racchiuso in una facile e intuitiva interfaccia WEB permettendo cosí l'accesso da ogni punto della rete.

La vera potenza di CACTI si trova nella disponibilitá di plugin e template specifici a ogni particolare evenienza. Tale offerta é il frutto del costante lavoro da parte della comunitá che rende il processo di sviluppo del software, in uno stato di aggiornamento continuo.

7.1 Installazione

Questo capitolo é dedicato interamente all'installazione di CACTI sul nostro sistema. Questo procedimento non presenta particolari difficoltá, tuttavia é necessario seguire tutte le fasi attentamente in modo da evitare probabili errori, che puntualmente si verificano.

I passi successi sono una raccolta dei principali comandi tratti dalla documentazione del sito ufficiale[31], eliminando passaggi che per i nostri scopi risultano essere superflui.

Per iniziare, é necessario controllare se sul nostro sistema é presente il pac-

cheto Net-SNMP; qualora non fosse presente, con i privelegi di amministratore, si procede all'installazione tramite il gestore dei pacchetti YUM:

```
# yum install net-snmp
# yum install rrdtool
```

Dopo di che, scarichiamo il pacchetto di CACTI dal sito ufficiale[32] e procediamo con la decompressione nella directory di Apache.

```
# cd /var/www/html
# wget http://surfnet.dl.sourceforge.net/sourceforge/cacti/cacti-0
.8.7b.tar.gz
# tar -zxvf cacti-0.8.7b.tar.gz
# mv cacti-0.8.7b cacti
```

Arrivati a questo punto si prosegue con la creazione di un database, importando le tabelle direttamente dal pacchetto di installazione di CACTI.

```
# mysqladmin -u root -h localhost -p create cacti
# mysql cacti -u root -h localhost -p < cacti.sql
```

Successivamente, proprio come abbiamo fatto con NAGIOS, si crea un utente specifico per CACTI.

```
# /usr/sbin/useradd cactiuser
# /usr/sbin/groupadd cacticmd
# /usr/sbin/usermod -G cacticmd cactiuser
```

Dopo di che si procede con la creazione di una password per MySQL e i configurando i relativi permessi.

```
# mysql -u root -h localhost -p
mysql> GRANT ALL ON cacti.* TO cactiuser@localhost IDENTIFIED BY 'password_MySQL';
mysql> flush privileges;
```

A questo punto ci si reca in */var/www/html/cacti/include/* e si modifica il file *config.php* inserendo le nuove credenziali.

```
\$database_type = "mysql";
\$database_default = "cacti";
\$database_hostname = "localhost";
\$database_username = "cactiuser";
\$database_password = "cacti";
\$database_port = "3306";
```

Successivamente, si configurano i permessi per la directory dei grafici e per quella dei log.

```
# chown -R cactiuser rra/ log/
```

Dopo di che é necessario un sistema che permetta a CACTI di aggiornarsi constantemente; per raggiungere questo obbiettivo, si sfrutta il lavoro effettuato da *cron*. Attraverso l'uso di un editor si aggiunge in */etc/crontab* il seguente comando che permette di eseguire il poller[1] di CACTI ogni cinque minuti.

[1] poller - *script che consente l'esecuzione di CACTI*

```
# */5 * * * * cactiuser php /var/www/html/cacti/poller.php > /dev/
null 2>&1
```

Nel precedente comando l'opzione */dev/null 2>&1* serve a non generare l'e-mail di notifica ogni volta che viene eseguito il poller.

Terminata la procedura di installazione si puó finalmente incominciare ad utilizzare CACTI, collegandoci direttamente dal nostro browser all'indirizzo *http://localhost/cacti/*, oppure tramite il comando:

```
#firefox http://localhost/cacti/
```

Al primo avvio noteremo che CACTI ci chiederá di inserire la password anche se noi precedentemente non ne avevamo impostata una. Per il primo avvio utilizzare le credenziali *admin admin*,e vedremo che alla schermata successiva ci verrá richiesto di impostare una nuova password: a questo punto inseriamo la nostra password da amministratore.

7.1.1 Installazione della Plugin Architecture

Dopo l'installazione di CACTI é necessario installare la *Plugin Architecture*, indispensabile per l'utilizzo e l'installazione dei plugin. Come abbiamo fatto precedentemente, ci procuriamo il pacchetto, lo scompattiamo e aggiorniamo il database:

```
# cd /var/www/html/cacti
# wget http://cactiusers.org/downloads/cacti-plugin-arch.tar.gz
# tar zxvf cacti-plugin-arch.tar.gz
# patch -p1 -N < cacti-plugin-arch/cacti-plugin-0.8.7b-PA-v2.1.diff
# mysql -u cactiuser -h localhost -p cacti < pa.sql
```

Enecessario modificare anche il file */var/www/html/cacti/include/global.php* perché avendo aggiornato il databese alcune informazioni sono state sovrascritte.

```
$config['url_path'] = "/cacti/";
```

Terminati questi comandi, é possibile visualizzare l'opzione *Plugin Management* modificando il relativo permesso in *Sistem Utilities*, *admin* e *Realm Permissions*.

A questo punto il nostro sistema é pronto per l'installazione di software aggiuntivo.

7.1.2 Installazione dei Plugin

L'installazione dei componenti aggiuntivi invece risulta comune a quasi tutti plugin. Per comprendere la metodica d'installazione, prendiamo ad esempio *Thold* e descriviamone in dettaglio i vari passaggi.

Per iniziare é necessario procurarsi il pacchetto e decomprimerlo nella cartella */var/www/html/cacti/plugins* di CACTI:

```
# cd /var/www/html/cacti/plugins
# wget http://cactiusers.org/downloads/thold.tar.gz
# tar zxvf thold.tar.gz
```

Successivamente é indispensabile aggiungere una riga al file */var/www/html/cacti/include/global.php* sotto *$plugins = array();* come segue

```
$plugins = array();
$plugins[] = 'thold';
```

Per concludere, é necessario controllare se le opzioni *Configure Thresholds* e *View Thresholds* sono state aggiunge nel *Realm Permissions*, proprio come per la *Plugin Architecture*.

7.2 Plugin e Template

Il costante lavoro compiuto dalla comunitá di CACTI contribuisce il regolare sviluppo di nuovi template e plugin, che poi vengono ampliamente utilizzati dagli utenti finali. Questi componenti aggiuntivi forniscono un alto grado di flessibilitá, poiché permettono di utilizzare CACTI in relazione alle specifiche macchine presenti nella rete. Se non ci fossero queste estensioni, l'intero applicativo sarebbe poco configurabile e sicuramente avrebbe un ciclo di vita destinato a concludersi.

Nello specifico, i plugin sono delle funzionalitá aggiuntive che forniscono all'amministratore un ulteriore supporto alla gestione della propria rete. Questi frammenti di software supplementari, possiedono specifiche operazioni che variano dalla reportistica, alla sicurezza, sino ad arrivare alla amministrazione dei log del sistema.

I template invece sono identificabili come dei modelli, al fine di gestire determinate macchine presenti nella rete. Il vantaggio di questi componenti si trova nel poter usufruire di uno specifico template per ogni device da monitorare, incrementando notevolmente il raggio di compatibilitá delle tecnologie con le quali l'amministratore deve operare.

Per comprendere meglio il concetto e in relazione con nostre esigenze, abbiamo deciso di utilizzare un template specifico per una stampante *HP Laserjet* presente nella rete all'indirizzo *193.205.92.25*. A tal fine abbiamo cercato sul forum ufficiale e scaricato[34] il template adatto ai nostri scopi. Il codice si presenta in formato XML[2] da importare nella sezione *Import Templates* del menú di CACTI, copiando il tutto nell'apposita form. A questo punto é sufficiente aggiungere il device da monitorare facendo attenzione di utilizzare il template correto, e creare un grafico che ci consenta di visualizzare i dati raccolti. Alla fine avremo ottenuto un grafico indicante il livello del toner della stampante.

Una buona norma prima di utilizzare questi template é quella di verificare, all'interno del codice XML, l'OID che viene utilizzato durante l'interrogazione

[2]XML *acronimo di eXtensible Markup Language, ovvero Linguaggio di marcatura estensibile é un metalinguaggio creato e gestito dal World Wide Web Consortium (W3C)*

via SNMP, sia quello corretto. Attraverso un semplice *snmpwalk* si capisce se immediatamente il template funzionerá correttamente.

Nei prossimi capitoli si analizzano i plugin che durante un fase di testing sono risultati maggiormente utili.

7.2.1 Graphs

La vera potenza di CACTI si identifica nella possibilitá di creare grafici specifici per ogni device presente nella rete. Questo compito é svolto ampliamente da *Graphs*, un plugin presente nell'applicazione di default, che consente anche di generare grafici secondo periodi temporali differenti (un'ora, un giorno, due giorni e cosí via). Una volta aggiunto il device da monitorare tramite l'opzione *Create Graphs for this Host* é possibile creare il grafico per l'host in questione. Nel caso in cui non dovesse comparare il grafico, pur avendo eseguito i passi precedenti, é utile eseguire l'opzione *Rebuild Poller Cache* presente in *System Utilities*, effettuando una sorta di aggiornamento. Questo plugin, inoltre, permette di organizzare i device secondo delle gerarchie, tramite l'opzione *Graph Trees*.

7.2.2 Discover

Questo plugin é sicuramente uno tra i piú utili, perché come si puó facilmente intuire dal nome, permette il discovery della rete, evitando cosí la noiosa procedura di inserimento manuale di ogni device.

Un'accortezza, degna di essere nota, riguarda la configurazione dell'orologio di sistema con quello hardware, poiché in alcuni casi non risultano essere sincronizzati. Infatti l'inizio del funzionamento del discovery dipende dall'orologio e se questo non risulta corrispondente alla realtá, é facilmente intuibile che il plugin non avvia l'esecuzione al tempo stabilito. L'orologio di sistema é possibile visualizzarlo tramite il comando *date*, mentre quello hardware attraverso *hwclock*. Se i due orologi non risultano avere gli stessi valori é necessario sincronizzarli tramite il seguente comando:

```
# hwclock --systohc
```

A questo punto é possibile inserire l'orario di inzio discovery, comprendente la sigla *AM*, nella form *Start Time for Polling* situata nell'interfaccia grafica sotto la voce *Settings* e *Misc*. Questa operazione é necessaria per l'utilizzo corretto del plugin.

7.2.3 Reports

Un'altra funzionalitá che non dovrebbe mai mancare in un software di monitoraggio é la possibilitá di generare report periodicamente dei device controllati. Il plugin che realizza questo compito risponde al nome di *Reportit*.

Il suo funzionamento é altro che banale, poiché durante la fase di configurazione é necessario prestare attenzione alle formule per il calcolo dei valori sui

report. Questo plugin infatti consente di personalizzare secondo le proprie esegenze l'output dei report.

A grandi linee, inizialmente é necessario creare un *Report Templates*, comprendente di variabili e misure, attraverso le opzioni *Create a new variable* e *Create a new measurand*. Infine occorre sbloccare il template appena creato, e aggiungere gli host, *Add data items*, in *Report Configuration* in modo da poter finalmente eseguire il report.

7.2.4 Mactrack

Un aspetto importante come la sicurezza é senza dubbio una delle prerogative dell'amministratore di rete. Un piccolo strumento di difesa, implementabile su CACTI, é *Mactrack*.

Questo plugin permette di interrogare lo switch di rete associando l'indirizzo MAC con la porta fisica alla quale é collegato. In questo modo, dato un indirizzo IP é possibile conoscere quale porta dello switch utilizza, evitando cosí possibili attacchi di tipo ARP Spoofing[3]. Per sommi capi, é necessario inserire un *Sites*, in *Device Tracking Management* in modo da poter poi configurare lo switch in *Devices*. Alla fine é possibile eseguire lo script e visuallizzare i dati.

7.2.5 Thold

Dopo aver configurato cosa monitorare e come, puó nascere il bisogno di controllare la quantitá di traffico presente nella nostra rete, in modo da riconoscere immediatamente i cosí detti *colli di bottiglia*.

Told diminutivo di *Threshold*, che sta per soglia, soddisfa le nostre richieste monitorando il livello minimo e quello massimo di qualsiasi servizio, e segnalando all'amministratore di rete via email oppure tramite dei colori rossi sull'interfaccia WEB, il superamento delle soglie. Tale plugin per funzionare correttamente necessita di un altro componente aggiuntivo, che risponde al nome di *Settings*. *Thold* é utile soprattutto quando nella rete sono presenti numerosi host, e la configurazione di ogni singolo template risulterebbe esageratamente lunga. La messa a punto di questo plugin é semplice, a differenza di quelli precedenti, e permette di effettuare tutto con pochi passaggi. In un primo momento é essenziale la creazione di un template con *Threshold Templates* impostando la soglia minima e massima, poi si aggiunge un device dal discovery e si crea un grafico per quel host, ed infine basta scegliere l'opzione *Auto-create thresholds* per autogenerare il template adatto.

I risultati saranno poi visibili in *Thresholds*, con le loro relative soglie minime e massime. Nell'instante in cui si verifica un superamento dei limiti, il plugin provvederá alla notifica.

[3]ARP Spoofing *é una tecnica di hacking che consente ad un attacker, in una switched LAN, di concretizzare un attacco di tipo MITM, Man In The Middle, verso tutte le macchine che si trovano nello stesso segmento di rete*

7.2.6 Syslog

Nella maggior parte dei casi, i log di sistema visto la quantitá e la rapiditá con la quale vengono generati, sono spesso eliminati e quindi non vengono considerati con la dovuta attenzione. Quando si verificano errori, o peggio ancora sono in atto attacchi informatici, i log di sistema immediatamente acquisiscono molta importanza poiché costituiscono le uniche informazioni attraverso le quali l'amministratore di sistema puó sapere quello che effettivamente é successo sulla macchina. Si fa presto a capire, che la gestione dei log diventa una operazione pesante, ma necessaria. Anche in questo caso la comunitá di CACTI ha sviluppato un plugin che permette la visualizzazione e lo storage dei log delle macchine della rete. L'installazione e la configurazione non presentano particolari difficolta. L'unico requisito é la presenza del server *syslogd* all'interno delle macchine della nostra rete.

7.2.7 Hostinfo

Lo sviluppo di CACTI ha permesso la creazione di numerose versioni sia della applicazione stessa che dei relativi plugin, in alcuni casi non pienamente compatibili tra di loro. Quando si utilizza questo software é di primaria importanza conoscere la versione adottata con quella dei plugin, molto utile quando si chiede supporto alla comunitá. Per semplificare questo processo, é nato *Hostinfo*, un plugin che fornisce tutte le informazione del nostro sistema, ideale per far conoscere l'intera struttura di cui disponiamo. L'installazione segue l'iter classico per l'installazione dei plugin, e la sua configurazione non necessita di particolari modifiche.

7.2.8 Monitor

La maggior parte dei plugin per CACTI, oltre alla creazioni dei grafici, estrapolano ulteriori dati utili per il monitoraggio. Uno di questi é *Monitor* che fornisce informazioni importanti, come l'*availability* che indica la disponibilitá del servizio, la data dell'ultima caduta, lo stato corrente e cosí via. L'installazione risulta comune alle altre estensioni, e la sua configurazione non presenta perticolari attenzioni. Per il suo utilizzo occorre solamente abilitare il *Monitor Host* nella configurazione device. Dopo di che saremo in grado di visuallizzare il device mediante l'interfaccia grafica di *Monitor*, che permette con una rapida occhiata di capire immediatamente lo stato della rete.

7.2.9 Npc, integrare Cacti con Nagios

Il forte sviluppo da parte della comunitá ha consentito la creazione di numerosi plugin tra i quali non poteva mancare uno che rispondesse alla nostra particolare esigenza di integrare i due applicativi NAGIOS e CACTI in un unica interfaccia grafica. Tale plugin si presenta con il nome di *Npc*. L'installazione risulta abbastanza complessa, e quindi ne illustreremo i singoli passaggi tratti dal sito ufficiale del plugin[35].

Per iniziare, procuriamoci il sorgente e lo decomprimiamo nella directory */var/www/html/cacti/plugins* attraverso i seguenti comandi:

```
# cd /var/www/html/cacti/plugins/
# wget http://www.assembla.com/spaces/npc/documents/aUjAwmd
W8r3BuPab7jnrAJ/download?filename=npc-2.0.0b.166.tar.gz
# tar zxvf download?filename=npc-2.0.0b.166.tar.gz
```

A questo punto modifichiamo il file *global.php* presente in */var/www/html/cacti* e aggiungiamo la seguente riga, come nella comune installazione di un plugin:

```
# \$plugins[] = 'npc';
```

Per eseguire la nuova estensione andiamo sull'interfaccia grafica in *Plugin Management* e scegliamo le sequenti opzioni, rispettivamente *Uninstall*, *Install*, *Installed* ed infine *Enable*.

Terminata l'installazione, proseguiamo con la configurazione. A tal fine, collochiamoci in *Settings*, *NPC* e abilitiamo il *Remote Commands*, inseriamo il *Nagios Command File Path*, che nel nostro caso é */usr/local/nagios/var/rw/nagios.cmd*, ed infine configuriamo la *Nagios URL* con *http://localhost/nagios*. Dopo di che siamo in grado di visualizzare l'interfaccia web di NAGIOS caricata sul nostro CACTI, in *NPC* e poi in *Nagios* simile a quella in FIGURA X.

Tuttavia per far funzonare correttamente *Npc* é indispensabile l'utilizzo di un altro pacchetto, *NDO2DB* che ha il compito di sincronizzare il database di CACTI con quello di NAGIOS. Mettiamo quindi in download il software e procediamo con la decompressione e installazione:

```
# wget http://downloads.sourceforge.net/nagios/ndoutils-1.4b7.tar.gz
# tar zxvf ndoutils-1.4b7.tar.gz
# cd ndoutils-1.4b7
# ./configure
# make
```

Se non sono comparsi errori, possiamo proseguire con i seguenti comandi che servono per abilitare l'utilizzo di NAGIOS con *NDO2DB*:

```
# cp src/ndomod-3x.o /usr/local/nagios/bin/ndomod.o
# cp config/ndomod.cfg /usr/local/nagios/etc
# cp src/ndo2db-3x /usr/local/nagios/bin/ndo2db
# cp config/ndo2db.cfg /usr/local/nagios/etc
```

Dopo di che bisogna modificare il file */usr/local/nagios/etc/nagios.cfg* controllando i seguenti valori:

```
check_external_commands=1
command_check_interval=-1
event_broker_options=-1
process_performance_data=1
```

e aggiungere questa stringa nella parte relativa al *broker_module*:

```
broker_module=/usr/local/nagios/bin/ndomod.o config_file=/usr/local/nagios/etc/ndomod
```

Ora é necessario modificare due file *ndo2db.cfg* e *ndomod.cfg* presenti in */usr/local/nagios/etc/* sia con i giusti permessi per il database che con i seguenti parametri:

```
# ndo2db.cfg #
ndo2db_user=nagios
ndo2db_group=nagios
socket_type=tcp
socket_name=/usr/local/nagios/var/ndo.sock
tcp_port=5668
db_servertype=mysql
db_host=localhost
db_port=3306
db_name=DATABSE_NAME
db_user=DATABASE_USER
db_pass=DATABASE_PASSWORD
db_prefix=npc_
max_timedevents_age=1440
max_systemcommands_age=10080
max_servicechecks_age=10080
max_hostchecks_age=10080
max_eventhandlers_age=44640
debug_level=1
debug_verbosity=1
debug_file=/usr/local/nagios/var/ndo2db.debug
max_debug_file_size=1000000

# ndomod.cfg #
instance_name=default
output_type=tcpsocket
output=127.0.0.1
tcp_port=5668
output_buffer_items=5000
buffer_file=/usr/local/nagios/var/ndomod.tmp
file_rotation_interval=14400
file_rotation_timeout=60
reconnect_interval=15
reconnect_warning_interval=15
data_processing_options=-1
config_output_options=2
```

A questo punto riavviamo sia NAGIOS che l'applicativo che ci permette l'integrazione con CACTI:

```
# /etc/init.d/nagios restart
# /usr/local/nagios/bin/ndo2db -c /usr/local/nagios/etc/ndo2db.cfg
```

Finalmente abbiamo terminato la procedura, e se non compaiono errori, significa che tutta la configurazione é andata a buon fine, ed é quindi possibile visualizzare il risultato ottenuto direttamente sull'interfaccia grafica di CACTI, come in FIGURA X

Capitolo 8

Automatizzazione

Il vero vantaggio che ha portato il cosí grande sviluppo dei computer trova la sua origine nella possibilitá automatizzare processi che se compiuti dall'umile lavoro dell'uomo risolterebbero assai lunghi. Anche nei software precedentemente esaminati, particolari plugin rispecchiano la figura di prodotto che automatizza delle operazioni.

Si puó quindi affermare che le varie funzioni di installazione e configurazione tendono ad occupare numerose ore di lavoro con conseguente inutile perdita di tempo. Per di piú l'esperienza insegna che difficilmente alle prime messe a punto tutto funziona correttamente. Spesso ci si imbatte in una serie infinta di errori e warning che portano l'amministratore direttamente in strato di frustazione. A volte questi genere di problemi si verifica anche con software di grande dimensioni, che dunque possiedono uno sviluppo guidato da grosse software-house.

Nel nostro caso specifico, quando si utilizzano i plugin, ci si scontra con pezzi di codice non di ottima qualitá. Tale affermazione non significa, non funzionanti, ma che non viene eseguito nessuna fase di testing e convalida che l'ingegneria del software insegna. Lo sviluppo si basa essenzialmente su un approccio di tipo evoluzionistico, ossia basato sulla prototipizzazione. Questa é la reale ragione per la quale vengono rilasciate numerose patch [1]. Spetta quindi all'utente finalre riconoscere falle, e segnalarle alla comunitá.

Per risolvere tutti questi problemi di compatibiltá e stabilitá del software, la comunitá di CACTI ha sviluppato una intera distribuzione dedicata, basata su CentOS 4.3, nella quale si trovano i piú diffusi plugin giá preinstallati e preconfigurati. Nel prossimo paragrafo viene appunto presentata questa distribuzione, indicandone tutte le caratteristiche tecniche.

[1] patch*letteralmente pezza é un termine inglese che indica un file eseguibile creato per risolvere uno specifico errore di programmazione, che impedisce il corretto funzionamento di un programma*

8.1 CactiEZ

CactiEZ costituisce la soluzione piú rapida e indolore per avere installato e funzionante un sistema con CACTI. Questa distribuzione é una immagine autoinstallante basata su CentoOS 4.3, e fornisce solamente i servizi strettamenti necessari. Ad esempio non presenta interfaccia grafica GUI[2]. L'immagine é di circa 355MB, mentre il sistema installato é poco piú di 855MB, dunque facilmente trasportabile su qualsiasi server. Possiede circa 305 pacchetti, e un firewall preconfigurato che consente l'accesso solamente ai servizi *HTTP*, *HTTPS*, *SSH*, *Netflow*, *Webmin*, e *SysLOG*.

Attualmente CactiEZ presenta CACTI versione 0.8.7 con la *Plugin Architecture* versine 2.1. All'interno possiamo trovare installati, oltre ad un buon numero di template, i seguenti plugin:

- Discovery
- Mactrack
- Monitor
- NTop
- RRD Cleaner
- Syslog (Haloe)
- Thold
- Tools
- Update
- Weathermap

Tutto é completamente preconfigurato e il lavoro dopo l'installazione iniziale é praticamente nullo. In questo modo si da la possibilitá anche a chi é meno avvezzo con sistemi LINUX di testare CACTI.

8.2 Script di autoconfigurazione

```
#!/bin/sh
nmap -v -sS -O -oX rete2.xml 193.205.92.0\24
./nmap2nagios.pl -v -r rete.xml -o rete.cfg
cp rete.cfg /usr/local/nagios/etc/objects/rete.cfg
service nagios restart
echo "Nagios Configurato Correttamente!"
```

[2]GUI *Graphical User Interface é lo strato software che si occupa del dialogo con l'utente del sistema utilizzando un ambiente grafico*

0pt0.4pt

Conclusioni

Seconda Bibliografia [**?**] dovrebbe essere numero 1 **e invece non lo é**
L'installazione di NAGIOS [26] PROVO LA BIBLIOGRAFIA

Provo tabella

Modello	Formato modello	Modello del rumore
ARX	$A(q)y(t) = \sum_{i=1}^{nu} B_i(q)u_i(t - nk_i) + e(t)$	Il modello del rumore è dato da $\frac{1}{A}$ ed è accoppiato al modello dinamico e non possono essere trovati separatamente.
ARMAX	$A(q)y(t) = \sum_{i=1}^{nu} B_i(q)u_i(t - nk_i) + C(q)e(t)$	È un estensione del modello ARX. Utilizza il parametro C per modellare in modo più flessibile il modello del rumore.
Box-Jenkins (BJ)	$y(t) = \sum_{i=1}^{nu} \frac{B_i(q)}{F_i(q)} u_i\,(t - nk_i) + \frac{C(q)}{D(q)} e(t)$	Permette di modellare in modo assolutamente indipendente il modello del sistema da quello del rumore.
Output-Error (OE)	$y(t) = \sum_{i=1}^{nu} \frac{B_i(q)}{F_i(q)} u_i\,(t - nk_i) + e(t)$	Usato quando non si ha la necessità di modellare il rumore.

Tabella 8.1: Modelli polinomiali discreti

0pt0.4pt

Bibliografia

[1] TUT: snmptranslate
http://net-snmp.sourceforge.net/wiki/index.php/TUT:snmptranslate

[2] RFC: Structure and Identification of Management Information for TCP/IP-based Internets
http://tools.ietf.org/html/rfc1155

[3] CCITT - Comité consultatif international téléphonique et télégraphique
http://www.itu.int/net/home/index.aspx

[4] ISO - International Organization for Standardization
http://www.iso.org/iso/home.htm

[5] IAB - Internet Architecture Board
http://www.iab.org/

[6] IANA - Internet Assigned Number Authority
http://www.iana.org/

[7] RFC - Remote Network Monitoring Management Information Base
http://www.ietf.org/rfc/rfc1271.txt

[8] MIB2 - Management Information Base for Network Management of TCP/IP-based internets
http://www.ietf.org/rfc/rfc1213.txt

[9] Snmpenum - SNMP Enumeration
http://www.filip.waeytens.easynet.be/snmpenum.zip

[10] OpManager
http://manageengine.adventnet.com/index.html

[11] SNMPc
http://www.snmpc.co.uk/

[12] Axence NetTools
http://www.axencesoftware.com

[13] Netmon
http://www.netmon.ca/

[14] InterMapper
http://dartware.com/index.html/

[15] Mon.itor.us
http://mon.itor.us//

[16] openNMS
http://www.opennms.org/index.php/Main_Page

[17] MRTG
http://oss.oetiker.ch/mrtg/

[18] Munin
http://munin.projects.linpro.no/

[19] Internet Engineering Task Force
http://net-snmp.sourceforge.net/

[20] MySQL
http://www-it.mysql.com/

[21] Net-SNMP
http://www.net-snmp.org/

[22] Tutorial: Writing a Dynamically Loadable Object
http://www.net-snmp.org/wiki/index.php/TUT:Writing_a_Dynamically_Loadable_Object

[23] Yum Extender
www.yum-extender.org

[24] The phpMyAdmin Project
http://www.phpmyadmin.net/home_page/index.php

[25] Nagios - Quickstart Installation Guides
http://nagios.sourceforge.net/docs/3_0/quickstart.html

[26] Nagios - The official Nagios website
http://www.nagios.org/

[27] Fedora 8 - Download Distribution
ftp://download.fedora.redhat.com/pub/fedora/linux/releases/8/Fedora/i386/iso/Fedora-8-i386-DVD.iso

[28] The Apache Software Foundation
http://www.apache.org/

[29] Nagios Exchange
http://www.nagiosexchange.org/

[30] Manuale di nmap in italiano
http://nmap.org/man/it/

[31] Documentazione ufficiale per l'installazione di Cacti
http://nmap.org/man/it/

[32] Cacti - The official Nagios website
http://www.cacti.net/

[33] RRDtool - Round Robin Database tool
http://oss.oetiker.ch/rrdtool/

[34] HP Laserjet Percentage Templates
http://forums.cacti.net/download.php?id=5245

[35] NPC - Nagios Plugin for Cacti
http://trac2.assembla.com/npc/wiki/QuickStartGuide

[36] GPL - General Public License
http://www.gnu.org/licenses/gpl.html