Amministrazione Unix

25 marzo 2005

Indice

Parte I

Environment

Capitolo 1

Introduzione

1.1 L'Amministratore di Sistema

1.1.1 Ruolo

1.1.2 Compiti

1.1.3 Responsabilità

1.2 Metodologia

1.2.1 Approccio Razionale (Modello)

1.2.2 Approccio Intuitivo (Esperienza)

1.3 Calcolatori Elettronici

1.3.1 Programma (Linguaggi, Interpreti , Traduttori)

1.3.2 Livelli (Layers)

1.3.2.1 Interfaccia

1.3.2.2 Servizi

1.3.2.3 Protocollo

1.3.3 Modello delle Macchine Multilivello

1.3.3.1 Hardware

1.3.3.2 Logica Digitale

1.3.3.3 Logica Combinatoria (Booleana)

- Logica Sequenziale (Macchine a Stati)
- Microprogrammazione

1.3.3.4 Macchina Convenzionale

1.3.3.5 Sistema Operativo

1.3.3.6 Linguaggio Assembler

1.3.3.7 Applicazioni

1.3.4 Principali Componenti Hardware

1.3.4.1 Central Processor Unit

1.3.4.2 Memoria

1.3.4.3 Device di Input/Output

1.3.4.4 Bus

1.4 Sistemi Operativi

1.4.1 Compiti Fondamentali

1.4.1.1 Gestione delle Risorse Hardware /drivers)

1.4.1.2 Controllo delle Applicazioni Attive

1.4.1.3 Interfaccia (Macchina Virtuale)

- processo (modello della memoria)
- interfaccia per il programmatore
- utente

1.4.2 Classificazione dei Sistemi Operativi

1.4.2.1 Mainframe

1.4.2.2 Server

1.4.2.3 Multiprocessor

1.4.2.4 Personal Computer

1.4.2.5 Real Time

1.4.2.6 Embedded

1.4.2.7 Smart Card

1.4.3 Concetti Fondamentali

1.4.3.1 Processi

- Multiprogramming
- Multitasking

1.4.3.2 Deadlocks

1.4.3.3 Memory Management

1.4.3.4 Input/Output

- Buffers

- Simoultaneus Peripherals Operation On-Line

1.4.3.5 Files

1.4.3.6 Security

1.4.3.7 Shell

1.4.3.8 System Calls

1.4.4 Modello a Strati

1.4.4.1 Kernel

1.4.4.2 System Calls

1.4.4.3 Librerie di Sistema

1.4.4.4 Programmi a corredo del S.O.

1.4.4.5 Applicazioni Utente

1.4.5 Struttura dei Sistemi Operativi

1.4.5.1 Kernel Monolitico

1.4.5.2 Kernel Modulare

1.4.5.3 Micro Kernel

1.4.5.4 Virtual Machines Systems

Capitolo 2

Fondamenti di Unix

2.1 Mentalità "How It Work" vs "How To"

2.2 EveryThing is a File

2.2.1 Tipi di File

2.2.1.1 Normal

2.2.1.2 Directory

2.2.1.3 Symbolic Link

2.2.1.4 Character Device

2.2.1.5 Block Device

2.2.1.6 (Named) Pipe

2.2.1.7 Socket

2.2.2 File Permissions

2.2.3 i-node

2.2.3.1 File Permissions

- Owner, Group, Other
- Read, Write, eXecute
- Set User IDentifier
- Set Group IDentifier
- Sticky Bit

2.2.3.2 Partizioni e Slice

2.2.3.3 Tabella degli I-Node

2.2.3.4 Struttura di un I-Node

- Type
- Owner

- Group
- Permissions
- Attributes (optional)
- Number of Link
- Pointers

2.2.3.5 Localizzazione dei File (Directory + I-Node)

2.2.3.6 Differenza tra Link Hard e Symbolic

2.2.3.7 Cache degli I-Node

2.3 Astrazione di "Albero"

- Partizionamento (Ordine)
- Gerarchia (Astrazione di Livello)

2.3.1 Hyerarchical File System

2.3.1.1 S.O. "in nuce" (runlevel 1)

etc: file di configurazione

bin: binari fondamentali

sbin: binari di amministrazione fondamentali

lib: librerie fondamentali

tmp: file temporanei

dev: descrizione dei device di sistema

var: log, spool, etc

2.3.1.2 S.O. con Demoni

2.3.1.3 S.O. con Servizi "Particolari"

2.3.2 Struttura ad Albero dei Processi

2.3.2.1 fork()

2.3.2.2 Process IDentifier

2.3.2.3 Process 0 (kernel)

2.3.2.4 Process 1 (init)

2.3.2.5 Bootstrap del Sistema

2.3.3 Comandi

- ogni comando esegue una singola azione, ma al massimo delle possibilità
- operazioni complesse tramite interazione tra comandi primitivi tramite operatori (scripting)
- gli standard

2.3.4 Breve Storia di Unix

2.3.4.1 Gli Albori

2.3.4.2 System V

2.3.4.3 Berkley Software Distribution

2.3.4.4 POSIX

Parte II

Network

Capitolo 3

Introduzione al Networking

3.1 Modello ISO-OSI

3.1.1 Physic Layer

3.1.2 Data Link Layer

3.1.3 Network Layer

3.1.4 Transport Layer

3.1.5 Session Layer

3.1.6 Presentation Layer

3.1.7 Application Layer

3.2 TCP/IP

3.2.1 Internet Protocol

3.2.2 Internet Control Message Protocol

3.2.3 User Datagram Protocol

3.2.4 Transmission Control Protocol

3.2.5 Kernel Parameters

3.3 Network Daemon

3.3.1 Unix Socket vs TCP/IP Socket

3.3.2 Connection, Session e fork()

3.3.3 Daemon e Superdaemon

3.3.4 TCP Wrapper

Capitolo 4

Secure SHell

4.1 Gli r-commands

4.2 OpenSSH

4.3 SSH Secure Shell

Capitolo 5

X Windows System

Nella storia dell'informatica non vi è mai stato un eccessivo interesse per le interfacce utente. Anche le odierne interfacce grafiche rappresentano solo un piccolo sforzo per il miglioramento della comunicazione uomo-macchina.

Solo contemporaneamente alla nascita di Unix, durante gli anni '70, si sviluppava una delle prime interfacce grafiche per computer: il sistema X negli Xerox's PARC laboratories.

L'X Windows System, che mirava a fornire a Unix, sistema ormai abbastanza maturo in cui erano assodati concetti portanti come Multiprogrammazione, multitasking, timesharing e supporto rete, una GUI, deriva dal progetto Athena, sviluppato inizialmente al Massachussets Institue of Technology.

5.1 Graphical User Interface

5.1.1 Vantaggi

5.1.1.1 Facilità del Controllo in Ambienti Multitasking

5.1.1.2 Migliore Presentazione dei Dati

5.1.2 Svantaggi

5.1.2.1 Maggiori Risorse Necessarie

5.1.2.2 Minore Versatilità

Capitolo 6

Web Server

6.1 Hyper Text Transfer Protocol

6.2 Hyper Text Transfer Protocol Secure

6.3 Apache Web Server

Capitolo 7

Electronic Mail

7.1 Introduzione

All'inizio non vi era un vero e proprio protocollo per il trasferimento delle e-mail. Si usava una sorta di FTP, in cui la risorsa trasferita conteneva nella prima riga esclusivamente l'indirizzo del destinatario. Questo portava però diversi svantaggi, così furono pensate soluzioni "ad hoc":

1982: ARPANET propone lo standard SMTP:

RFC 821: trasmission protocol

RFC 822: message format

1984: CCITT abbozza lo standard X.400

1986: OSI MOTIS

1988: CCITT adegua lo standard X.400 per essere pienamente compatibile con il MOTIS

Negli anni '90 divenne chiaro che lo standard sarebbe stato quello proposto dall'ARPANET, in quanto:

- più semplice
- meno ambiguo
- più facilmente implementabile

7.1.1 Architettura Generica

Un servizio di Posta Elettronica consta solitamente di due entità:

Mail Transfer Agent: tipicamente un demone di sistema, che lavora dunque in background, il cui compito è muovere il messaggio dall'originatore alla destinazione

User Agent: denominato anche client; programma locale che permette di leggere e spedire le e-mail, interagendo con il MTA

7.1.1.1 Servizi Fondamentali

Composition: processo di creazione del messaggio

Transfer: spostamento del messaggio dall'originatore al recipiente del destinatario

Reporting: comunicazione all'originatore dell'esito del trasferimento del messaggio

Displaying: insieme delle operazioni necessarie alla visualizzazione del messaggio (e.g. conversione e formattazione)

Disposition: manipolazione dei messaggi in arrivo (e.g. filtri, conservazione, etc)

7.1.1.2 Servizi Avanzati

Forwarding: inoltro automatico dell'e-mail su un altro recipiente

Mailboxes Manipulation: insieme delle operazioni di gestione dei raccoglitori di messaggi

Mailing List: lista di indirizzi di recipienti, in modo da facilitare l'invio di messaggi a gruppi di recipienti

Notification: messaggi di verifica dell'avvenuta consegna, lettura, etc

Carbon Copy:

Blind Carbon Copy:

Priority:

Encryption:

Alternate Recipient:

7.2 Simple Mail Transfer Protocol

7.2.1 Concetti Fondamentali

User Agent: programma (mail reader) che accetta i comandi dall'utente per fornire le funzionalità di mailing

Recipient: indirizzo di posta elettronica configurato per poter ricevere la posta. Differenti sistemi adottano differenti metodi di identificazione:

SMTP: l'identificazione avviene mediante una stringa legata al nome DNS del server e del recipient, nel modo: `<recipient-name>@<host-name>`

X.400: l'identificazione avviene mediante una stringa del tipo:

```
recipient::= /<attribute>=<value>/[<attribute>=<value>/]
```

Message Structure: il messaggio, nel suo complesso, si compone dei seguenti elementi:

Envelope: contenitore del messaggio. In esso è presente una introduzione che contiene tutte le informazioni utili all'instradamento, necessarie al corretto funzionamento dell'MTA

Header: informazioni di controllo per lo User Agent. Generalmente duplicano alcune informazioni contenute nell'Envelope

Body: contenuto del messaggio

Message Storing: i messaggi vengono conservati in recipienti di vario genere:

Mailboxes: il formato più diffuso; si tratta di un file di testo, essendo la e-mail fondamentalmente di testo, in cui i vari messaggi sono giustapposti uno dopo l'altro, conservando però informazioni di envelopping minimali

Maildir: formato meno diffuso ma di manipolazione più facile e sicura; ogni messaggio viene conservato in file a sè stante, sempre mantenendo informazioni di envelopping minimali

DB proprietario: generalmente i software di tipo commerciale conservano i messaggi in formati strutturati ma soprattutto proprietari e binari

7.2.2 Message Format (RFC 822)

Ogni messaggio consiste di una sequenza di byte interpretati come ASCII text (cui solo i primi 7 bit hanno significato) costituito dai seguenti elementi:

Message Header: intestazione nella quale sono contenuti, uno per ogni riga, i campi necessari alla manipolazione del messaggio da parte dello User Agent, nel seguente modo:

```
<field>::=<field-name>:<field-value>[;<attribute>=<value>]
```

Line Feed: carattere che crea una riga vuota, che simboleggia la fine dei campi

Message Body: corpo del messaggio in ASCII form (solo 7 bit per ogni byte sono interpretati)

7.2.2.1 Header Fields Principali

Questi sono i campi della intestazione che devono essere obbligatoriamente presenti:

To: recipient del destinatario

From: recipient dell'originatore

CC: recipient dei destinatari addizionali

BCC: recipient dei destinatari addizionali, che non vengono però visualizzati dagli altri

Sender: recipient della persona che fisicamente invia il messaggio (potrebbe essere diversa da From)

Received: campo riempito di volta in volta dai MTA per segnare ogni passaggio

Return-Path:

7.2.2.2 Header Fields Secondari

Questi sono Header Fields che non devono essere obbligatoriamente presenti:

Date: data di arrivo

Reply-To: indirizzo cui rispondere

Message-ID: identificativo univoco del messaggio

In-Reply-To: identificativo del messaggio di cui il corrente è la risposta

References:

Keywords:

Subjects: oggetto del messaggio

7.2.2.3 eXtended Header Fields

Secondo la RFC 822 è possibile inventare nuovi fields, utilizzando nomi che cominciano per "X-". I seguenti sono i fields estesi che si trovano più comunemente:

X-Mailer: indica lo User Agent utilizzato per scrivere la e-mail

X-MimeOLE:

X-Priority: indica il livello di pririotà

X-UIDL:

X-Authentication-Warning:

X-Mime-Autoconverted:

X-Accept-Language:

X-Scanned-By, X-Virus-Scanned: indica che il messaggio è stato esaminato da un antivirus

X-Disclaimer:

X-Subliminal:

X-Admonition:

X-MSMail-Priority:

X-Original-ArrivalTime:

X-Originating-IP:

X-Originating-Server:

X-Sender:

X-Auth-User:

X-MIMETrack:

X-MS-Has-Attach:

X-MS-TNEF-Correlator:

7.2.3 Multipurpose Internet Mail Extension (RFC 1341, RFC 1521)

Estensione dell'RFC 822 per inviare altri tipi di dati, ovvero inserire nei messaggi informazioni che fossero solo ASCII a 7 bit. Prevede l'introduzione di altri 5 fields nell'Header:

MIME-Version: versione del protocollo. Campo obbligatorio, altrimenti il messaggio viene interpretato come ASCII normale; in pratica indica che si sta usando il MIME.

Content-Description: descrizione sommaria del contenuto

Content-ID: identificativo del contenuto (analogo a Message-ID)

Content-Type-Encoding: metodo di codifica del contenuto. Si possono definire tip di codifica personalizzati anche se solitamente si utilizza uno dei 5 encoding standard seguenti:

ASCII Text: metodo standard, analogo all'RFC 822 semplice; di ogni byte sono utilizzati solo i primi 7 bit; ogni riga può essere lunga non più di 1000 caratteri

8-bit: di ogni byte sono utilizzati tutti e 8 i bit (in questo modo si aggiungono i caratteri accentati); ogni riga non può essere lunga più di 1000 caratteri

binary: gli 8 bit di ogni byte sono usati in binario, onde senza limite dei 1000 byte a riga; può dare luogo a fraintendimenti nella analisi, in quanto i messaggi, come illustrato precedentemente, sono intrinsecamente di testo

base64 (ASCII Armor): particolare codifica dei contenuti binari attraverso ASCII, in cui:

1. ogni gruppo originale di 24 bit (3 byte) viene splittato in 4 unità da 6 bit (da cui il nome, infatti $2^6 = 64$)
2. ogni unità viene tradotta in ASCII secondo la corrispondenza indicata nella tabella di seguito presentata
3. qualora il gruppo ultimo contenga solo un byte, viene inserito alla fine un "="
4. qualora il gruppo ultimo contenga solo due byte, vengono inseriti alla fine "=="

quoted-printable encoding: usato per trasmettere messaggi in cui la parte binaria è minima ($\Rightarrow$ *base*64è inefficiente); funziona nel modo seguente:

- tutti i caratteri $<$ 128 sono codificati ASCII
- tutti i caratteri $>$ 127 sono codificati nel modo "`=<hex-value>`"

Unit Binary Value	base64 Character Encoding
0	A
1	B
...	...
25	Z
26	a
27	b
...	...
51	z
52	0
...	...
61	9
62	+
63	/

Poichè i messaggi sono fondamentalmente sequenze di bit trattate come ASCII, conviene trasmettere i messaggi in una delle 2 codifiche: base64 o quoted-printable

Content-Type: specifica la natura del contenuto del messaggio, fornendo utili indicazioni su come effettuarne la decodifica, attraverso direttive della forma: `<type>/<subtype>`. Nella RFC 1521 sono definiti i type e subtype di partenza

7.2.3.1 MIME Types

text: testo semplice; la codifica ASCII potrebbe bastare

image: imagine (e.g. GIF, JPEG)

audio: flusso audio

video: flusso video

application: contenuto per la fruizione del quale è necessario l'ausilio di una applicazione esterna (e.g. postscript)

message: messaggio incapsulato

multipart: permette ad un messaggio di essere composto da più parti. Necessaria la specifica `boundary` che permette di discernere il confine tra due parti consecutive

mixed: i contenuti sono indipendenti ma devono essere visualizzati nell'ordine in cui sono stati inseriti

alternative: i contenuti sono alternativi, in quanto contengono tutti lo stesso tipo di informazione, ma in formati diversi, normalmente inseriti dal più complesso al più semplice, in modo che lo User Agent sia in grado di scegliere il contenuto adeguato

parallel: i contenuti devono essere visualizzati in contemporanea (e.g. audio e video)

digest: ogni contenuto è un messaggio separato

7.2.4 Message Transfer (RFC 821)

Viene effettuato secondo Simple Mail Transfer Protocol.

7.2.4.1 Comandi SMTP

Tutti i comandi, per motivi non ben noti, sono composti da 4 lettere e sono case insensitive:

HELO <indirizzo-sorgente>: indica al server SMTP che ci si sta connettendo dal dominio <indirizzo-sorgente>. Non è obbligatorio

MAIL FROM: <recipient>: indica al server SMTP che l'originatore è <recipient>

RCPT TO: <recipient>: indica che si vuole mandare un messaggio a recipient. Viene attivato un meccanismo di controllo per verificare la effettiva eseguibilità del comando.

DATA: indica che si sta cominciando ad inserire il messaggio. Di seguito andrano inseriti l'Header e il Body. Si indica che il messaggio è terminato inserendo una riga che contiene solo il carattere "."

QUIT: permette di sconnettersi dal server

7.2.4.2 Extended SMTP

Con SMTP possono sorgere i seguenti problemi:

64KB: vecchie implementazioni non possono superare il limite dei 64 KB a messaggio

TimeOut: differenti configurazioni dei tempi di timeout tra client e server possono portare a scollegamenti

MailStorm: qualora due server ospitino ciascuno una lista in cui siano presenti dei riferienti vicendevoli, si può generare un invio infinito di e-mail

Per questi ed altri motivi è stato ideato l'ESMTP (RFC1425)

7.2.4.3 E-Mail Gateway

Sono dei server che hanno il compito di tradurre da un sistema di posta ad un'altro (e.g. SMTP vs X.400).

7.3 Final Delivery

Oggi non è più necessario che il destinatario della posta abbia un MTA sulla propria macchina, infatti la struttura di Rete e Software è fatta in modo tale che la macchina del destinatario possa collegarsi ad un server, un MTA, e reperire la posta.

Questa operazione è possibile compierla attraverso molteplici meccanismi e protocolli.

7.3.1 Post Office Protocol version 3

Il protocollo largamente più diffuso per la lettura della posta è il POP3 (RFC 1225). Esso permette di:

- Autenticarsi sul Server (contrariamente all'SMTP)
- Scaricare la Posta
- Cancellare la Posta

7.3.1.1 Comandi POP3

Analogamente a SMTP, POP3 è un protocollo che accetta comandi in formato testo. I comandi implementati sono:

user <username>: invia <username> come identificativo utente per il login

pass <password>: invia <password> come parola chiave associata all'identificativo utente digitato precedentemente

list: lista le e-mail presenti

retr <number>: scarica il messaggio numero <number> nella lista

dele <number>: cancella il messaggio numero <number> nella lista

quit: effettua il login dal server

7.3.2 Altri Protocolli

7.3.2.1 Interactive Mail Access Protocol (RFC 1064)

Per utenti che usano più computer e quindi non gradiscono scaricare la posta su uno solo ed usano quindi il server di posta come repository

7.3.2.2 Distribuited Mail System Protocol (RFC 1056)

Parte del PCMAIL. I messaggi possono essere su server differenti.

7.4 E-Mail Security

7.4.1 Pretty Good Privacy

Utilità dovuta a Phil Zimmermann (1995), è un programma estreno allo User Agent (ma ultimamente integrabile) che consente:

- confidenzialità
- autenticazione
- digital signatures
- compression
- key management facilities

Si basa su protocolli standard (RSA, MD5) ma non definiti secondo dettami di enti governativi e dunque abbastanza affidabili.

7.4.1.1 Funzionamento

Supponiamo che Alice voglia scrivere a Bob e che ciascuno abbia la chiave privata (D_X) e la chiave pubblica (E_X), quest'ultima conosciuta anche dall'altro. Sia M il messaggio originale:
$N = RSA_{D_A}(MD5(M)), M$
$N.z = zip(N)$
$P = IDEA_{K_M}(N), RSA_{E_B}(K_M)$
$M' = base64(P)$
Di seguito il messaggio viene inviato come ASCII standard, poichè contiene solo caratteri ASCII.

7.5 Mail User Agents

7.5.1 /bin/mail

Berkeley Mail Program

7.5.2 /bin/mailx

System V Mail Program

7.5.3 mush, elm, mh, pine

7.6 Qmail

7.6.1 Installazione

Reperire i seguenti pacchetti, da installare nell'ordine:

- ucspi-tcp-0.88 (utilità di tcp wrapping)
- daemontools-0.76 (utilità di environment configuration)
- netqmail-1.04 (programma vero e proprio)

7.6.2 Compilazione

7.6.3 Architettura

7.6.3.1 /var/qmail/bin

Daemons

qmail-start: fa partire i demoni qmail-lspawn, qmail-rspawn, qmail-send, qmail-clean

qmail-lspawn:

qmail-rspawn:

qmail-send:

qmail-clean:

qmail-smtpd:

qmail-pop3d:

qmail-popup:

Queue Monitoring

qmail-qread:

qmail-qstat:

qmail-tcpto:

Message Sending Utilità per l'invio di messaggi

qmail-inject: legge un messaggio dall standard input e lo invia, riscrivendo l'header. Invoca `qmail-queue`

mailsubj: inserisce l'oggetto (`Subject: ...`) e la lista dei recipient (`To: ...`) nell'header di un messaggio

User Programs Di seguito vengono elencati una serie di programmi di utilità per gli utenti, alcuni resisi necessari come rimpiazzamento di altri di interfaccia a MTA più tradizionali, data la diversità si approccio di Qmail (e.g. Maildir anzichè Maibox).

Utilità "Una Tantum" E' necessario configurare un ambiente base di lavoro per la consegna delle e-mail ad ogni singolo utente. Tali rpogrammi aiutano a questo scopo

maildirmake: costruisce la directory Maildir (composta dalle sottodirectory `cur`, `new`, `tmp`) per le proprie e-mail. Per i nuovi utenti è possibile definire automaticamente la creazione di tali directory inserendole in `/etc/skel`

maildir2mbox: effettua la conversione da formato Maildir a Mailbox

maildirwatch: controlla l'arrivo di nuovi messaggi in una Maildir

Utilità Richiamabili anche da .qmail E' possibile definire il comportamento di Qmail personalizzato, sulla base del settaggio che viene effettuato sul file `.qmail` nella propria `HOMEDIR`. In particolare è possibile convenire una serie di operazioni sulle e-mail in arrivo, utilizzando le utilità seguenti:

qbiff: analogo di biff, controlla se vi sono nuove e-mail

qreceipt: notifica ad un indirizzo remoto l'arrivo di una e-mail in Maildir

forward: effettua il forward di una e-mail

bouncesaying: invia ogni nuovo messaggio ad un programma, recependo l'eventuale codice di errore di quest'ultimo

condredirect: redirezione condizionale dei messaggi, in funzione del codice di uscita del programma cui sottopone ogni messaggio

7.6.3.2 /var/qmail/boot

Programmi di ausilio

7.6.3.3 /var/qmail/queue

7.6.3.4 /var/qmail/rc

Run Control Script per la partenza di netqmail (Qmail 1.04).

7.6.4 Configurazione

7.6.4.1 /var/qmail/control

File di configurazione (`qmail-control (5)`).

Hostname Related Control Files File di configurazione legati a hostname e dunque di controllo per i recipient. L'unico veramente necessario è me, eventualmente usato come default per gli altri hostname-related.

me: contiene il Fully Qualified Domain Name del server di posta

defaultdomain: contiene il nome di dominio che viene aggiunto alla fine di ogni nome di macchina che sia stato specificato non FQDN

defaulthost: host di default

plusdomain: viene aggiunto alla fine di ogni nome di macchina indicato con un segno "+" finale

locals: indica, i domini che vengono considerati come locali, uno per ogni riga; in pratica permette di fare il virtual server. Per indirizzi afferenti a tali domini viene chiamato `qmail-lspawn` anzichè `qmail-rspawn`.

rcpthosts: indica i domini che vengono accettati nel campo RCPT, uno per ogni riga, senza senza implementare completamente il virtual host ed accettando dunque messaggi provenienti da tali domini. Per indirizzi afferenti a tali domini viene chiamato `qmail-rspawn`. In tale modo è possibile il Domain Name Selective Relaying.

morercpthosts:

badmailfrom: domini dai quali non accettare le e-mail (Black List)

badrcptto: domini verso i quali non inviare le e-mail

bouncehost: indirizzo che deve comparire nei messaggi di mancato invio dei messaggi. Default `me`

virtualdomains:

doublebouncehost:

helohost

idhost

idhost

localiphost:

locals:

Limits File di configurazioni che indicano valori di limite

timeoutsmtpd: tempo, in secondi, oltre il quale il demone `qmail-smtpd`, in assenza di comandi ulteriori, si disconnette

timeoutremote: tempo, in secondi, oltre il quale, in assenza di risposte, il demone `qmail-remote` si disconnette

timeoutconnect: tempo, in secondi, oltre il quale, non ricevuta risposta, il demone `qmail-remote` cessa il tentativo di connessione

concurrencylocal:

concurrencyremote:

percenthack:

queuelifetime:

Miscellaneus
doublebounceto: Default postmaster

nodnscheck:

7.6.4.2 /var/qmail/users

7.6.4.3 /var/qmail/alias

7.6.4.4 /var/qmail/service

Contiene le 3 directory:

smtp: per la partenza del server SMTP

pop3: per la partenza del server POP3

send:

run: script per la partenza

env: directory contenente la configurazione delle variabili

log: directory contenente i log

7.7 Sendmail

UUCP usa sendmail come programma uux.

7.7.1 Components

7.7.1.1 Configuration File

Contiene tutte le informazioni necessarie a Sendmail per essere eseguito:

- file location
- permissions
- mode of operation
- rewriting rules (la parte più criptica)

7.7.1.2 Queues

Rappresentano le cache dei messaggi che non si sono pututi trasmettere immediatamente, salvati localemente in modo da garantire la ritrasmissione. Una e.mail potrebbe essere messa in coda per i seguenti motivi:

- destination machine unreachable or down
- some recipient not successful reached
- message expensive (on a slow line)
- safety (queue all message)

7.7.1.3 Aliases e Mailing List

Permettono di redirezionare le e-mail dirette verso un indirizzo verso un altro

7.7.2 Command Line

Sendmail può essere usato anche a riga di comando, sebbene sia una pratica poco diffusa.
`/usr/sbin/sendmail <rcpt-to> <message>.`

7.7.3 sendmail.cf

Il pricipale file di configurazione, quello che vine letto prima di tutti gli altri. Il formato è il seguente:
`<file> ::= <line> {<line>LF}`

V Define configuration file version (beginning with V8.6)

M Define a mail delivery agent

D Define a macro

R Define a rewriting rule

S Declare a rule-set start

C Define a class macro

F Define a class macro from a file or a pipe

O Define an option

H Define a header

P Define delivery priorities

T Declare trusted users (ignored in V8.1, back in V8.7)

K Declare a keyed database (beginning with V8.1)

E Define an environment variable (beginning with V8.7)

L Include extended load average support (contributed software, not covered)

```
<line> ::= <Option-line> | <Mailer-line> | <Comment-line>|<File-line> | <K-line>
     <Option-line> ::= O <Option-name> = <value>
     <Mailer-line> ::= M<descr> <Mailer-options>
     <Comment-line> ::= #.*
     <K-line> ::= K<> <K-options>
     <File-line> ::= F<file> <File-options>
     <descr> ::= local | prog
     <Mailer-options> ::= P=<program>, F=<?>, S=<value>, R=<>, D=<>
```

7.7.4 aliases

File che descrive le conversioni fra i recipient e i nomi utente reali. Viene specificato dalla direttiva:

```
O AliasFile=/etc/mail/aliases
```

Il formato è del tipo:

```
<line> ::= <user-line> | <list-line> | <include-line> | <to-file-line> | <command-line>
<user-line> ::= <alias>:  <realname>
<list-line> ::= <alias>:  <realname> {,<realname>}
<include-line> ::= <alias>:  :include:  <filename>
<to-file-line> ::= <alias>:  <file-name>
<command-line> ::= <alias>:  |<command-full-path>
```

Un esempio tipico:

```
# Mandatory aliases.
postmaster:  root
MAILER-DAEMON: postmaster
# The five forms of aliases
John_Adams:  adamj
xpres:  ford,carter,bush
oldlist:  :include:  /usr/local/oldguys
nobody:  /dev/null
ftphelp:  |/usr/local/bin/sendhelp
```

I nomi degli alias sono case-insensitive. Le sostituzioni vengono ricercate ricorsivamente nel file finchè non se ne trovano più.

7.7.5 Queue Directory

Directory nella quale vengono messe le e-mail in attesa di essere inviate. Viene indicata dalla direttiva:

```
O QueueDirectory=/var/spool/mqueue
```

Ogni e-mail viene conservata in due file:

- header file: il cui nome comincia con qf
- body file: il cui nome comincia con df

Capitolo 8

Name Server

8.1 Domain Name System

Generalmente non ci riferisce ai servizi di INternet tramite l'inidirizzo IP del server. Al posto viene utilizzato un sistema di nomi di "mnemonico", basato su sequenze di caratteri ASCII. In conseguenza di ciò, poichè lo strato di rete discerne unicamente indirizzi binari, è necessario fornire un meccanismo che aiuti ad associare agli indirizzi numerici quelli mnemonici e viceversa.

8.1.1 ARPANET naming

Agli albori di Internet, quando il numero di macchine effettivamente in uso era minimo, si utilizzava un meccanismo molto semplice: vi era un server centrale, presso il *Network Information Center* dello *Standford Research Institute*, sul quale veniva manutenuto il contenuto di un file denominato host, in cui in ogni riga veniva inserito un indirizzo IP ed i nomi ad esso associati. Tale file veniva prelevato giornalmente, generalmente durante la notte o la mattina presto, da tutti gli hosts collegati su Internet.

Questo approccio, sicuramente valido nelle prime fasi, non era utilizzabile a lungo, fondamentalmente per i seguenti motivi:

Name Collisions: difficile controllo sui conflitti di nomi

Size: eccessiva lunghezza del file

Traffic and Load: eccessivo carico di richieste verso il server centrale

Consistency: dato il ritmo elevato di crescita di Internet, la manutenzione diveniva difficile

Fu subito chiaro che la soluzione alternativa doveva fornire:

- Amministrazione Decentralizzata
- Disponibilità dei Dati Mondiale

8.1.2 Domain Name System (RFC 1034, 1035)

Paul Mockapetris fu il responsabile di progetto della architettura del nuovo sistema, la cui prima versione, del 1984, è descritta nelle RFC 882 e 883. Tali specifiche sono state sostituite nel 1987 dalle RFC 1034 e 1035. Ulteriori RFC sono state aggiunte in seguito con argomenti vari, tracui: sicurezza, problemi implementativi, meccanismi di aggiornamento dinamico.

Il DNS è un sistema di DB distribuiti, contenenti informazioni relative agli host e metainformazioni necessarie al funzionamento del DNS stesso.

Le informazioni, non risiedendo più su ogni singolo host, devono essere all'occorrenza reperite in rete. Allo scopo viene adoperato il classico modello client-server, per mezzo dei programmi *name server* e *resolver*.

Il principio essenziale di tale sistema è uno schema di organizzazione gerarchico, sulla base di una struttura ad albero n-ario, i cui elementi intermedi (nè radice, nè foglia) sono detti domini, relativamente a:

Domain-based names: costruzione dei nomi mnemonici

Distribuited DataBase: distribuzione delle informazioni

Tale sistema permette di effettuare varie operazioni di associazione, anche se in origine fu ideato principalmente per il mappaggio di indirizzi di posta elettronica (standard SMTP) e hostname sui corrispondenti indirizzi IP.

8.1.2.1 DNS Name Space

In questa sezione viene esaminato il modo in cui vengono costruiti i nomi dei domini e degli host.

I nomi vengono costruiti tramite una struttura ad albero. Ogni livello identifica l'albero sottostante, ovvero il *dominio*, e viene indicato mediante una etichetta (*label*).

Concetti Fondamentali Definizioni importanti per la comprensione del funzionamento del Name Space e dell'intero DNS

domain sotto-alberi propri dell'albero dei nomi DNS

delegation atto, da parte di una struttura amministratrice di un determinato dominio, di assegnare la responsabilità della gestione di un proprio sotto-dominio ad un'altra organizzazione

zone sotto-insieme di un dominio, contiene le informazioni di delegation dei sotto-domini e talvolta le informazioni di sotto-domini non delegati

name-server programma che conserva le informazioni relative agli host di una o più zone

authoritative name-server di riferimento per ua determinata zona

data file file contenente le informazioni su una o più zone

Costruzione dei Nomi La costruzione del nome viene fatta includendo il path completo dalla radice al dominio specifico, in modo analogo a quanto viene fatto per i nomi dei file su uno hierarchical file system Unix, ma elencando il persorso in ordine inverso ed usando il punto (dot) come separatore dei domini di vario livello (al posto dello slash) , nel seguente modo:

`<name>::={<sub-domain-name>.}[<second-level-domain>].[<top-level-domain>].`

dove

`<top-level-domain>::=<generic-domain> | <country-domain>`

Possono essere utilizzati in modo interscambiabile i termini top-level domain e first-level domain.

Si osserva esplicitamente che:

- il dominio corrispondente alla root (unnamed domain) corrisponde ad un semplice punto
- le label sono case insensitive
- le label componenti il path non possono superare i 63 caratteri ciascuno
- il path completo non può superare i 255 caratteri
- il numero delle label componenti il path non può superare 127

- ogni macchina dovrebbe appartenere ad un dominio, onde è possibile utilizzare path di dominio relativi (teoricamente tutti quelli senza punto finale)
- un dominio non può contenere un sotto-dominio o host aventi la stessa label o nome

Il nome di dominio (path) può essere indicato:

- in modo assoluto (includendo il "." della root)
- in modo relativo rispetto alla root (omettendo il "." della root) detto Fully Qualified Domain Name
- in modo relativo rispetto al dominio in cui ci si trova

Ogni dominio controlla l'allocazione dei domini sotto di esso. E' necessario ottenere un'autorizzazione da chi amministra il dominio per poterne definire uno subalterno (e.g. è necessario ottenere l'autorizzazione di chi amministra `uniroma1.it` per poter aprire `ing.uniroma1.it`)

generic-Top-Level-Domains (gTLDs): domini ideati per contenere una classificazione iniziale dei domini. Utilizzata quasi esclusivamente dalle organizzazioni statunitensi

com: commercial

edu: educational institutions (ora "liberalizzata")

net: network provider (ora "liberalizzata")

org: organization no-profit (ora "liberalizzata")

gov: U.S. federal government institution

int: international organization

mil: military

Nuovi gTLDs sono stati creati recentemente come biz, info, tv (cfr. http://www.gtld-mou.org).

Country-Domain: definiscono una entry per ogni stato, come da direttiva ISO 3166

8.1.2.2 DNS Server

In questa sezione viene descritto il modo in cui agisce un server DNS ed in particolare le modalità con cui viene realizzata la distribuzione del DataBase dei nomi su tutto Internet.

Per fare in modo tale da non forzare un DNS server a contenere tutto il DataBase DNS mondiale, lo spazio dei nomi è stato diviso ricorsivamente in *zones* non sovrapponibili, per ognuna delle quali è competente un DNS distinto. Si osserva che che le zones non coicidono necessariamente con i domini o sotto-domini.

Ogni server DNS può gestire più di una zona.

Resolution La ricerca di un "association record" può avvenire in uno dei seguenti modi:

local query: la richiesta viene fatta al DNS locale, che riesce ad evaderla (e.g. nel caso di domini Intranet)

recursive query: la richiesta viene fatta ad un DNS, magari quello locale, che però non riesce ad evaderla, in tal caso esso stesso contatta in modo ricorsivo i DNS server, partendo da uno dei root name server, fino ad arrivare al server autoritativo per la zona cui afferisce il record cercato

iterative query la richiesta viene fatta ad un DNS, magari quello locale, che però non riesce ad evaderla, in tal caso esso stesso contatta una serie di name server, dei quali uno risponderà con il record cercato

next server: la richiesta viene fatta ad un DNS che non riesce ad evaderla e comunica al richiedente l'indirizzo di un altro DNS server da contattare

Nel caso di ricerca di tipo ricorsivo, ci si deve riferire ad uno dei root name server per conoscere l'indirizzo del DNS autoritativo per il top-level domain ricercato. Attualmente ce ne sono una trentina sparsi per il mondo (MILNET, SPAN, NASA Internet, Europa, Giappone)

Domain Server Types Ogni DNS Server può svolgere una o più delle seguenti funzionalità:

Primary Master: conservazione permanente del DB di riferimento delle associazioni per il proprio dominio. Risponde alle richieste per il proprio dominio con "*Authoritative Records*", in quanto è l'autorità che li gestisce

BackUp Master: conservazione permanente di una copia del DB di riferimento delle associazioni per il proprio dominio, prelevata periodicamente dal Primary Domain Server, attraverso l'operazione di "*zone transfer*"

Cache: conservazione temporanea delle associazioni, in funzione delle richieste transitate

in-addr.arpa Importante è anche il meccanismo mediante il quale viene effettuato il mappaggio inverso, dagli indirizzi IP ai nomi di dominio (utile ad esempio nella generazione di log o per controlli di accesso host based).

Occorre inserire una ulteriore struttura dati per non dover effettuare una ricerca esaustiva nel DB organizzato per nomi e non per indirizzi.

Per questo motivo è stato creato una porzione del Domain Name Space, inizialmente dedicata alla transizione da HOST.TXT a DNS, che utilizza gli indirizzi IP come label.

In tale sotto-spazio, i nodi sono rappresentati da label in rappresentazione "*dotted-octet*", dove l'ottetto più significativo rappresenta la label di terzo livello, mentre l'ottetto meno significativo rappresenta la label di settimo livello. Ad esempio `www1.uniroma1.it` che corrisponde a `151.100.4.4`, avrà una rappresentazione inverse address `4.4.100.151.in-addr.arpa`. Come si nota gli indirizzi appaiono in ordine invertito, in questo modo permettendo di effettuare la delegation anche su in-addr.arpa

8.1.2.3 Resource Records

La funzione primaria di un DNS server è quell a di mappare i nomi in "resource record" ovvero insiemi di dati a struttura prefissata inerenti una determinata risorsa. Tali informazioni possono essere di svariato tipo. Vediamo la struttura di tali informazioni:

Domain_Name: nome di dominio della risorsa cui si riferisce il record. Solitamente per una risorsa vi sono più di un record

Time_to_live: indicazione sulla durata di validità delle informazioni contenute in tale record. Espreso in secondi, generalmente assume valori da 1 a 86400 (numero di secondi in un giorno)

Class: classificazione delle informazioni.

IN: Internet (il più diffuso)

CH: CHAOSnet

HS: HESIOD

Type: indica il tipo di record. I più importanti sono:

Start Of Authority: fornisce le seguenti informazioni:

Primary Master NS: il nome della risorsa primaria di informazioni riguardo la zona

Admin E-mail: indirizzo di posta cui inviare report di problemi qualora necessario (eventulamente in modo relativo)

Serial: numero seriale univoco, generato come YYYYMMDD<day-serial>

Refresh Interval: intervallo allo scadere del quale lo slave server contatta il primary per reperire eventuali modifiche di configurazione delle zone

Retry numero di secondi scaduti i quali lo slave riprova a contattare il primary in caso di fallito collegamento

Expire scaduto tale intervallo di tempo senza essere riuscito a contattare il Primary, lo Slave elimina la zona in questione permanentemente

Timeout

Address: rappresenta un'associazione semplice nome-indirizzo IP

Mail eXchange: definisce il mail server cui inviare la e-mail il cui indirizzo di destinazione ...

Name Server: definisce un name server (e.g. il DNS di livello superiore, del top-level domain)

Canonical NAME: permette di creare degli alias, per mezzo di macro

PoinTeR: puntatore ad un altro Domain name. Generalmente usato per far puntare ad un domain name associato ad un indirizzo IP

HostINFOrmation: descrive il tipo di macchina e sistema operativo

TEXT: record utilizzato per convenienza dell'utente e non necessario. Contiene informazioni varie

Value: indica il valore assegnato. Può essere un:

numero:

stringa ASCII:

domain name:

8.1.3 Security

- Zone Transfer Access Control (IP + TSIG)
- No TXT, HINFO e MINFO type
- no name e-mail address in field
- Anti-Spoofing
- Anti Cache Poisoning
- Running As Non-Root

8.2 DNS Client

8.2.1 nslookup

8.2.2 host

8.2.3 dig

8.3 Berkley Internet Name Domain

8.3.1 Cenni Storici

La prima implementazione si deve a Mockapetris stesso e si chiamava JEEVES. In seguito fu sviluppata una sua evoluzione BIND, per BSD Unix 4.3. Attualmente il programma è manutenuto dall'Internet Software Consortium (cfr. http://www.isc.org/bind.html).

La versione attuale è la 9, risalente al 2000, sebbene non sia raro trovare in uso ancora le versioni 8 e 4.

8.3.2 Files

/usr/sbin/named Name Server Daemon

/etc/named.conf file di configurazione

/usr/sbin/named-checkconf utility di verifica della correttezza sintattica del file di configurazione

/usr/sbin/named-checkzone utility per il check della correttezza sintattica di un file contenente le informazioni relative ad una zona

/usr/sbin/named-bootconf.sh script di settaggio della configurazione di partenza

/var/named/ directory contenente i working files

/usr/sbin/rdnc Remote Name Daemon Control; programma di controllo e configurazione da remoto del DNS server

/etc/rndc.conf file di configurazione del RNDC

8.3.3 Sintassi di named.conf

La sintassi del file principale di configurazione /etc/named.conf è mutuata direttamente dal C:

- righe di commento (stile C++)
- struct

8.3.3.1 options

La struttura options contiene le opzioni di partenza del server:

directory locazione dei "working file"

query-source porta sorgente per l'inoltro di richieste

forward first indica di inoltrare le richieste

forwarders {<ns-server-ip>; [<ns-server-ip>;} lista dei server cui vengono inoltrate le richieste

8.3.3.2 controls

opzioni di controllo d'accesso da parte di rndc

inet indirizzi

8.3.3.3 key "<name>"

struttura relativa alle opzioni delle chiavi per la comunicazioni con i backup e rndc

algorithm algoritmo di hashing. Attualmente l'unico supportato è l'MD5

secret shared secret (stringa)

8.3.3.4 zone "<name>"

strutture relative alla definizione delle zone. Generalmente devono essere presenti almeno le definizioni per le zone:

- .
- <localzone>
- <localnet>.in-addr.arpa

Le principali opzioni disponibili sono:

type tipo di gestione della zona

master il ns è authoritative e primary master per questa zona

slave il ns è authoritative, ma è slave

hint il ns non è authoritative, onde chiederà ad altri

masters {<ns-ip-addr>; [<ns-ip-addr>;]} lista dei ns da cui prelevare le informazioni

notify [yes|no] configura se notificare o meno i cambiamenti allo slave server

file file contenente i resource record della zona (path relativo rispetto a *directory*)

allow-transfer {<ns-server>; [<ns-server>;]} permette di effettuare il trasferimento della zona (AXFR e IXFR), ma solo verso le macchine indicate

allow-query {<ip-addrs>|<network>;[<ip-addrs>|<network>]} risponde unicamente alle richieste che provengono dagli indirizzi indicati

allow-recursion {<ip-addrs>|<network>;[<ip-addrs>|<network>]} permettere la ricorsione unicamente da indirizzi interni o locali alla rete, in modo da eliminare il rischio di cache poisoning

8.3.4 Zone File Format

<line>::= <null-line> | <comment line> | <TTL-line> | <record-line>

<comment-line>::= <CRLF> <comment>

<comment>::= ";" {[alphanum]}

<TTL-line>::= "$" "TTL" <seconds-number>

<seconds-number>::= {[0-9]}+ | {[0-9]}+M | {[0-9]}+H | {[0-9]}+D

<record-line>::= <SOA-line> | <resource-line>

<SOA-line>::= “@” <class> “SOA” <authority-name-server> <administrator-e-mail> <serial>
<refresh> <retry> <expire> <minimum> <comment>

<serial>::= <year> <mounth> <day> <day-serial>

<refersh>::= <seconds-number>

<retry>::= <seconds-number>

<expire>::= <seconds-number>

<minimum>::= <seconds-number>

<resource-line>::= <domain-name> <type> <value> <comment>

<inverse-resource-line>::= <number> “PTR” <domain-value>

Nei resource record di tipo MX il domain name può essere omesso qualora si intenda il medesimo del record A precedente.

Capitolo 9

Network Management

9.1 Simple Network Management Protocol

Ai tempi di ARPAnet, il controllo dello stato di salute delle macchine attestate in rete veniva effettuato mediante l'esecuzione del comando ping. Tale approccio era efficace dal momento che il numero di sistemi presenti era esiguo.

Con la crescita delle dimensioni di Internet, occorreva un sistema maggiormente efficiente. Per tale motivo sono stati ideati vari protocolli che permettono di eseguire dei controlli automatici sui nodi remoti. Uno dei protocolli maggiormente utilizzati è l'SNMP, del quale sono state progettate le seguenti release:

SNMPv0: RFC1028 e 1067

SNMPv1: RFC 1155 e 1157 del maggio 1990

SNMPv2: RFC 1441-1452

SNMPv2c:

SNMPv2*:

SNMPv3: RFC 2274 (User Based Security Module) gennaio 1998

Ulteriori RFC riguardanti componenti ed aggiunte:

MIB-II: RFC1213

Identification MIB: RFC1414

Host Resource MIB: RFC 1514

9.1.1 Modello SNMP

9.1.1.1 Componenti

Managed Nodes: componenti da monitorare

Management Station: computer che effettua il monitoraggio

Management Information: informazioni tramite le quali viene effettuato il monitoraggio

Management Protocol: protocollo attraverso il quale vengono scambiate le informazioni

9.1.2 Managed Nodes

Ogni nodo da monitorare contiene un agente software; deve essere il più leggero possibile, capace di leggere e scrivere (secondo richiesta) una serie di variabili, denominate *objects*.

Il modo in cui vengono organizzati gli oggetti è stabilito nella definizione delle Management Informations.

La comunicazione tra Managed Nodes e Management Station avviene attraverso il Management Protocol.

9.1.3 Management Station

Contiene il software speciale di amministrazione, molto più corposo rispetto all'agente presente sui singoli nodi. Comunica con i nodi, inviando comandi ed aspettando risposte. Contiene una copia di tutti gli objects definiti su ogni nodo, arrangiati in una struttura delle MIB (Management Information Base).

- query-response: la management invia delle richieste ai nodi aspettando risposta
- snmp-trap: qualora succeda qualcosa di strano a bordo di un nodo, esso lo comunica alla management di competenza autonomamente
- trap-directed-polling: la management interroga i nodi con cui non comunica da qualche tempo per sapere se esistono delle situazioni di errore di cui non è a conoscenza
- authentication: effettuata tramite MD5 ad ogni comunicazione dei nodi
- segretezza:

9.1.4 Management Information Base

E' la struttura dati contenente tutti i possibili objects necessari al management. La definizione degli objects, e dunque anche la costruzione dei contenuti di comunicazione mode-management, viene effettuata con l'ausilio dell'ASN.1. Mediante tale linguaggio è stato definito l'albero degli oggetti MIB.

9.1.4.1 ASN Data Description Language (ISO 8824)

SNMP utilizza solo un sottoinsieme dell'intero linguaggio, in modo da eliminare tutto quello che non è distretta utilità, offrendo la possibilità di semplificare l'agent e la management. Un file contenente definizioni ASN.1 per SNMP presenta, dunque, il seguente formato:

```
<identifier> <type> ::= <value>

<identifier> ::= [a-z]+

<type> ::= <standard-type> | <defined-type>

<standard-type> ::= INTEGER | BIT STRING | OCTET STRING | NULL | OBJECT IDENTIFIER

<defined-type> ::= [A-Z][alphanum]+
```

dove:

INTEGER: numero intero

BIT STRING: stringa di bit, di una data lunghezza, rappresentante ciascuno un flag

OCTET STRING: stringa di ottetti (caratteri)

OBJECT IDENTIFIER: stringa di identificazione del particolare object, definita in base alla struttura dell'albero MIB

inserire disegno dell'albero MIB

9.1.4.2 ASN.1 Transfer Syntax (ISO 8825)

Si tratta della sintassi utilizzata nel trasferimento, ovvero il modo in cui vengono codificate le informazioni, il risultato di una sorta di "compilazione" delle informazioni contenute nei file ASN.1.

Lo standard SNMP stabilisce di usare la codifica Basic Encoding Rules.

9.1.4.3 Structure of Management Information

Per adattare l'ASN.1 alle effettive necessità dell'SNMP, sono state definite nella RFC 1442:

- 8 nuovi tipi di dati
- 4 key macros

9.1.4.4 MIB-II

Albero generale degli oggetti definito nella RFC 1213.

Il MIB-II è stato suddiviso in dieci categorie (i 10 nodi del MIB-II nell'albero di ASN.1):

System: nome, locazione, descrizione, responsabile, etc.

Interfaces: interfacce di rete e statistiche di comunicazione associate

AT: corrispondente ARP (obsoleto=

IP: dati sui pachetti

ICMP: IP error messages

TCP: connessioni aperte, segmenti inviati e ricevuti

UDP: datagrams inviati e ricevuti

EGP: Extension Gateway Protocol

Transmission: gruppo riservato per dati particolari

SNMP: statistiche dell'SNMP stesso

Altri gruppi posono essere aggiunti.

9.1.5 SNMPv3

Ideato per includere le seguenti features:

- Security
  - Authentication
  - Privacy
  - Access Control
- Engine-Module Software Architecture
- Processing
  - Protocol Data Unit (tipo di comando management object list)
  - Message (Message Header)

9.1.5.1 Formato del Messaggio

- Message Processing Fields (5)
  - msgVersion:
  - msgID: 32 bit
  - msgMaxSize: dimensione in byte del messaggio
  - msgFlags (3bit):
    - reportFlag: Report PDU restituito al mittente
- USM Fields (6)
  - securityName: nome utente
  - authProtocol: MD5 | SHA
  - prvProtocol: null | DES
  - authKey
  - privKey
- scoped PDU:
  - content engine ID
  - content name
  - PDU

9.1.5.2 View Based Access Control Model

Differenti livelli di accesso per differenti principal. Definizione per gruppi di utenti.

9.2 Net-SNMP

9.3 Sun Management Center

9.4 Sistemi Integrati

9.4.1 Tivoli

9.4.2 HP Open View

9.4.3 CA Unicenter TNG

Capitolo 10

Network Information System

Il NIS è un servizio per la fornitura di informazioni, che devono essere fruite tramite rete, a tutte le macchine della rete. tipicamente:

- login, password, home directories (/etc/passwd)
- group information (/etc/group)

Il supporto per il "traditional NIS" è inserito, generalmente, nelle librerie standard libc di sistema.

E' detto anche Yellow Pages (tm), dal nome, depositato da British Telecom, dell'elenco telefonico riservato alle informazioni su esercizi commerciali.

Il funzionamento del NIS è realizzato tramite apposite RPC (Remote Procedure Call).

10.1 Definizioni

NIS+: Network Information System Plus, ovvero una evoluzione di NIS, progettata da Sun Microsystems Inc., in modo da aggiungere supporto per una più adeguata gestione della sicurezza e di installazione di grosse proporzioni

NSS: Name service Switch, supporto introdotto da Sunn nei suoi sistemi SOlaris 2, per la scelta del meccanismo ideale di risoluzione dei nomi (tipicamente DNS, hosts, NIS)

NYS: acronimo di NIS+, YP, Switch. Simboleggia un progetto, diretto da Peter Eriksson, che contiene la reimplementazione di NIS (alias YP) in modo da utilizzare le funzionalità NSS

10.2 Funzionamento

10.2.1 Network Information System HostTypes

Master: server principale, contenente le informazioni. Ne notifica il cambiamento agli slave server (yppush)

Slave: server secondario, fornisce le informazioni ai client, riceve le notifiche di avvenuto cambiamento delle informazioni dal master e provvede prontamente a reperirle

Client: macchina che reperisce le informazioni, indicate in /etc/nsswitch.conf, dal server NIS più vicino

10.2.2 NIS+

Nuova versione ideata da Sun. La più grossa differenza consiste nel supporto per:

- data encryption
- authentication

attraverso l'uso di secure RPC

10.2.2.1 Naming Model

Basato sulla struttura di albero. Ogni nodo nell'albero corrisponde ad un oggetto NIS+, del quale esistono 6 tipi differenti:

- directory
- entry
- group
- link
- table
- private

La directory che forma la root del namespace NIS+ è chiamata la root directory. Esistono due directory NIS+ speciali:

org_dir: contiene tutte le tabelle di amministrazione: passwd, group, host, mail_aliases

group_dir: contiene tutti gli oggetti che vengono usati per l'Access Control

L'insieme di org_dir, group_dir e la directory genitrice (la root) viene detto "NIS+ domain".

10.2.3 RPC Portmapper

RPC portmapper è un server, in ascolto sulla porta 111, che converte il numero di un programma RPC in un numero di porta TCP o UDP. Deve essere attivo, in modo da consentire di "pubblicare" i programmi disponibili ad un eventuale client che ne facesse richiesta. Il client al momento dell'eventuale utilizzo chiede preventivamente al portmap se esistono i programmi di cui necessita e su quale porta risiedono.

I vari server RPC comunicano alla loro partenza a portmap la porta su cui sono in ascolto. Poichè alcuni di essi sono avviati da inetd è necessario che portmap sia avviato prima di inetd.

10.2.4 Client Programs

In questa sezione vengono presentati i programmi che svolgono le funzioni del client

ypbind: demone, deve essere avviato durante il boot

ypwhich: restituisce il nome del NIS server utilizzato da una determinata macchina o per una determinata mappa

ypcat: visualizza i valori di tutte le chiavi di un database NIS

yppoll: restituisce la versione ed il master server di una mappa NIS

ypmatch: stampa il valore di una o più chiavi di una mappa NIS

ypset: setta il NIS server su una determinata macchina

yppasswd: analogo di passwd (cambia la password)

ypchsh: analogo di chsh (cambia la shell)

ypchfn: analogo di chfn (cambia il GECOS field)

/etc/yp.conf: file di configurazione per il settaggio del nome di dominio NIS e dell'indirizzo del server

/var/yp/nicknames: contiene una lista di aliases per i nomi delle mappe

10.2.5 Server Programs

ypserv: NIS server. Legge il file di configurazione /etc/ypserv.conf

yppush: copia il database NIS aggiornato da un server Master ad uno Slave. Normalmente viene avviato da /var/yp/Makefile (commentando la riga NOPUSH=true)

rpc.ypxfrd: permette di accellerare la copia delle mappe da un server master ad uno slave, consentendo la copia diretta dei file contenenti la mappa del master, anzichè facendo costruire la mappa allo slave autonomamente

rpc.yppasswd: si accorge del cambiamento delle password da parte di un utente e conseguentemente avvia l'agiornamento del DB

/var/yp/securenets: le reti dalle quali è possibile collegarsi al server

/usr/lib/yp: contiene vari script per la configurazione di partenza

Capitolo 11

Lightweight Directory Access Protocol

Protocollo client-server leggero per l'accesso a servizi di directory X.500. Funziona su protocolli orientati alla connessione ed è definito nella RFC 2251 nella sua terza versione.

Parte III

Security

Capitolo 12

Security Introduction

12.1 Principi Fondamentali

12.1.1 Availability

Previene la distruzione di servizi e produttività

12.1.2 Integrity

Previene la modifica non autorizzata di sistemi e informazioni

12.1.3 Confidenziality

Previene la diffusione non autorizzata di informazioni private

Capitolo 13

Risk Analysis

13.1 Concetti Fondamentali

13.1.1 Threats

13.1.2 Risk

13.1.3 Attack

13.1.4 Vulnerability

13.2 BS 7799

13.3 ITSEC

13.4 Common Criteria

Capitolo 14

Security Architecture

14.1 Hardening

14.1.1 Trusted Computing Based

14.2 Clustering

Capitolo 15

Network Security