Amministrazione di una rete

Introduzione all'amministrazione di rete

Con il termine amministrazione di rete si intende indicare l'insieme delle funzioni di gestione necessarie per garantire la corretta utilizzazione dei servizi, le funzionalità e la qualità di una rete.

È compito dell'amministratore di rete implementare le procedure e le politiche necessarie per garantire il corretto funzionamento della rete, a partire dall'autenticazione degli utenti fino al monitoraggio e al mantenimento delle corrette funzionalità degli apparati.

In questo corso verranno fornite informazioni sulle principali funzioni di amministrazione di rete, partendo proprio dall'autenticazione degli utenti e dalla regolamentazione nell'uso dei servizi, per giungere alle più tipiche funzioni di gestione e monitoraggio di rete.

Il problema dell'autenticazione

In un sistema informativo distribuito l'autenticazione riguarda la verifica dell'identità di un utente.

Sulla base di questa verifica il sistema permette o nega l'utilizzazione di risorse e/o l'esecuzione di procedure.

Gli schemi adottati per l'autenticazione sono:

- *User-to-host* (da utente a *host*). Metodi usati dall'*host* per identificare gli utenti.
- *Host-to-host* (da *host* a *host*). Tecniche utilizzate dagli *host* per convalidare l'identità di altri *host*, in modo da poter scoprire eventuali comunicazioni fraudolente.
- *User-to-user* (da utente ad utente). Metodi per essere certi che i dati elettronici originino effettivamente dal supposto utente e non da qualcuno che si spaccia per il mittente.

Tecniche di autenticazione

Le tecniche mediante le quali è possibile identificare *host* o *user* sfruttano quello che sei (*Something You Are* - SYA), quello che sai (*Something You Know* - SYK) o quello che possiedi (*Something You Have* - SYH).

Le caratteristiche di queste tre tecniche possono essere così schematizzate:

- SYH - L'utente viene identificato per mezzo di qualcosa che possiede, detto *token* che può essere incluso in una *smart card* o in un tesserino bancomat. Per potere essere autenticato e quindi accedere al sistema l'utente deve possedere il *token* e, di norma, essere a conoscenza di un segreto, ad esempio un *PIN* o una *password*. Questo tipo di metodologia di identificazione può presentare i seguenti problemi:
  - Il *token* può essere smarrito, clonato o al momento non disponibile.
  - Il *token* comporta un costo che in alcuni casi può anche essere elevato.
  - Il correto uso del *token* presuppone l'esistenza di una infrastruttura *hardware* e *software* che può essere piuttosto complessa.
- SYA - L'utente viene identificato per mezzo di qualcosa che è. Appartengono a questa categoria i meccanismi di identificazione biometrica. L'utente è autenticato sulla base di fisici precedentemente impostati e su di lui modellati. Questa metodologia di identificazione può presentare i seguenti problemi:
  - Alta percentuale di errore, stimabile nel 10%.
  - Rischio di intrusione nei sistemi di rilevazione.
  - Costo elevato delle attrezzature.

- SYK - L'utente viene identificato per mezzo di qualcosa che sa. È questo il metodo di riconoscimento a mezzo *password* segreta. È sicuro se ben realizzato ossia se abbinato a tecniche crittografiche quando le *password* viaggiano sulla rete, è efficiente, non è intrusivo e, soprattutto, è economico.

A seconda di come gli schemi precedenti vengono applicati, la tecnica di autenticazione può essere:

- a fattore unico;
- a due o più fattori.

L'autenticazione a fattore unico è basata sul possesso o la conoscenza di una singola entità, ossia un elemento che solo l'utente ha a disposizione (per esempio lo schema basato sulla conoscenza di nome_utente e *password*, dove il primo viene distribuito all'intera organizzazione, mentre la parola chiave è nota solo all'utente). Tali schemi non presentano un elevato grado di sicurezza: quando le *password* sono facili da ricordare, spesso lo sono anche da indovinare; occorre quindi definire scelte rigide per la scelta delle *password*, evitando facili schemi, quali *date* di nascita, iniziali di nomi, eccetera (inoltre è bene modificarle periodicamente).

Gli schemi a due fattori si basano sulla memorizzazione di un'entità e sul possesso di un'altra (esempio possesso di una *card* e conoscenza di un *PIN*). La sicurezza è migliore rispetto agli schemi a fattore unico; tuttavia risulta di scarsa praticità d'uso. Tra l'altro, oltre a poter dimenticare l'informazione, è possibile smarrire l'oggetto necessario all'autenticazione.

Per ragioni di semplicità ed economia la forma di autenticazione più usata, in *Internet*, è SYK a fattore unico con l'utilizzo della *password* segreta.

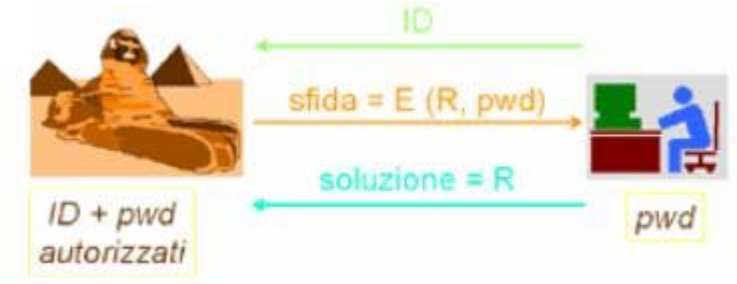
Protezione della password

L'autenticazione SYK basata su *password* normalmente utilizzata nei sistemi distribuiti odierni, per garantire un efficiente livello di sicurezza, richiede l'introduzione di tecniche per evitare la cattura delle *password*, ogniqualvolta queste vengano trasmesse in rete. Infatti utenti non autorizzati potrebbero essere in grado di intercettare le informazioni che viaggiano sulla rete e quindi appropriarsi delle *password* altrui.

Una valida e anche abbastanza semplice soluzione prevede di non fare transitare in rete le *password* in chiaro, ma di utilizzare informazioni che sono funzione della *password*.

Una possibile tecnica è quella delle sfide simmetriche o asimmetriche.

La sfida simmetrica opera come mostrato in figura:



Il *server* (lato sinistro della figura) riceve una richiesta di autenticazione con al suo interno un *ID* utente. Il *server* verifica che tale *ID* sia noto, ed invia in risposta il numero:

sfida = E(R, pwd)

in cui E è la funzione di cifratura, R è un numero casuale e pwd è il segreto condiviso tra *server* e utente (*password*). In questo modo, conoscendo lo *standard* di cifratura, e solo se si conosce il segreto pwd si sarà in grado di ricavare il numero R generato dal *server* in maniera casuale.

La sicurezza può essere ulteriormente aumentata se, ad autenticazione avvenuta, il processo viene ripetuto ad intervalli di tempo casuali durante la sessione di lavoro.

Nel caso di crittografia asimmetrica si utilizzano i certificati al posto della *password* (segreto condiviso). Si ottengono medesimi risultati che con il sistema simmetrico, ma si evita l'utilizzo di segreti condivisi, con inoltre il vantaggio di una gestione più semplice e conforme a *standard* internazionali come quello del certificato X.509 e delle infrastrutture di chiave pubblica (*Public Key Infrastructure* - PKI).

Si consideri l'esempio in figura:

Il *server* autenticatore (lato sinistro della figura) riceve una richiesta di autenticazione contenente un *ID* utente rappresentato dal certificato dell'utente stesso. Il *server* verificherà che tale certificato sia noto ed autorizzato, fed invierà una risposta data da:

sfida = E(R, Kpub)

in cui E è la funzione di cifratura, R è un numero casuale e Kpub è la chiave pubblica dell'utente. In questo modo, conoscendo lo *standard* di cifratura, e il segreto Kpub potrà essere ricavato il numero R generato dal *server* in maniera casuale.

Un attacco mediante intercettazione è inefficace, perchè intercettare il numero R, non fornisce sufficienti informazioni per poter replicare la sessione, dal momento che al prossimo tentativo il numero R generato dal *server* sarà diverso dal precedente.

Questi schemi di protezione delle *password* sono alla base di architetture più complesse quali quelle di *Kerberos* e *Secure Socket Layer* (SSL) che vengono oggi comunemente implementate per la realizzazione di autenticazioni sicure in ambienti di rete e a cui è dedicato uno specifico approfondimento.

Amministrazione di sistema

I moderni sistemi operativi di rete, ed in particolare i due già scelti come esempo nel modulo 7, ossia *Red-Hat Linux* e *Windows* 2000, sono sistemi operativi multiutente. Più utenti possono pertanto accedere contemporaneamente al sistema ed utilizzarne le risorse, quali dati contenuti nelle memorie di massa (direttori e *file*) o programmi.

Questo accesso può avvenire direttamente dalla *console* del calcolatore oppure tramite rete.

In questi sistemi, al fine di garantirne la sicurezza, ad ogni risorsa sono associati determinati permessi, in modo da restringerne l'accesso ai soli utenti abilitati.

Nel seguito vedremo quali sono le modalità per creare gli utenti, associare loro delle proprietà ed abilitarli o meno all'uso delle risorse del sistema stesso.

Nel caso di *Windows* faremo riferimento alla versione *Windows* 2000 *server* in inglese.

Prima però di andare nel dettaglio operativo su questi argomenti introduciamo il concetto di *directory service* che, come vedremo, trova applicazione nella stessa gestione degli utenti.

Amministrazione di sistema Linux

In *Linux* la gestione degli accessi viene attuata a livello del nucleo (*kernel*) del sistema operativo, ad esempio nel momento in cui un programma, attraverso una funzione di libreria, richiama la funzione *open* su un *file*.

Ogni programma in esecuzione sul sistema (processo) acquisisce i permessi dell'utente che lo ha mandato in esecuzione (*owner*, proprietario). Questo vale anche per i programmi che non vengono lanciati direttamente, come quelli attivati mediante lo schedulatore di sistema (*cron*) oppure avviati da un servizio di rete. Si pensi ad esempio ad uno *script* lanciato dal *server Web* oppure ai programmi di *delivery* lanciati dal *server* di posta elettronica al momento della ricezione di un nuovo messaggio.

Per questo motivo anche i programmi che offrono i servizi di rete o sovrintendono alle funzionalità di sistema, i cosiddetti demoni, sono comunque associati ai permessi di un utente *UNIX*. Comunemente un demone viene avviato al *boot* del sistema mediante uno *script* in `/etc/rc.d` utilizzando i permessi dell'amministratore di sistema (*root*) e come prima operazione esegue una chiamata al sistema operativo (*chroot*) che ne imposta come proprietario un utente non privilegiato. Spesso viene utilizzato allo scopo un utente con *username nobody* oppure un utente dedicato (ad esempio *mail* per il sottosistema che gestisce la posta elettronica oppure *webmaster* per il *web server*).

Definizione di un account di utente

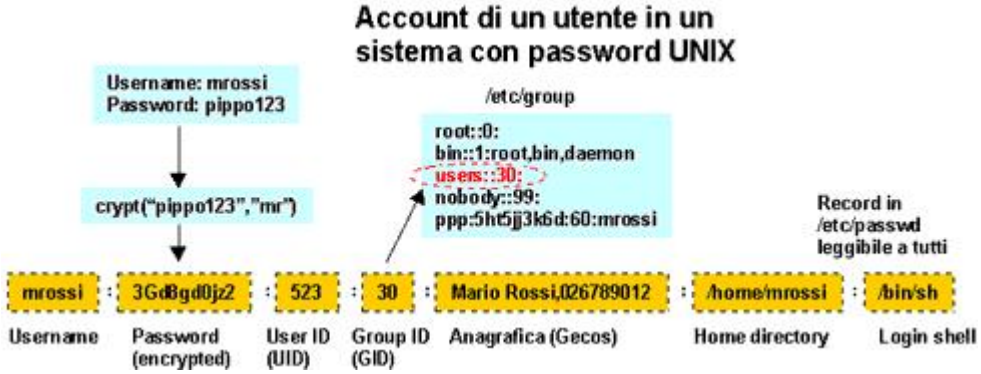
In un sistema *Unix* gli utenti vengono definiti mediante un'apposita stringa di informazioni contenuta nel *file* `/etc/passwd`. Ad ogni utente definito nel sistema è associata una stringa alfanumerica che lo identifica univocamente, il cosidetto *username*. In realtà, nelle funzioni interne del *kernel*, i diversi utenti sono definiti mediante un valore numerico univoco, lo UID (*User Id*).

Oltre allo UID, ad ogni utente è associato un GID (*Group ID*), che identifica a quale gruppo di utenti esso appartiene. Si tratta di un valore numerico che viene tradotto in forma simbolica mediante il *file* `/etc/group`. Esso viene utilizzato, in modo analogo allo UID, per marcare la proprietà di una risorsa nella gestione dei permessi dei gruppi. Il GID associato ad un utente in `/etc/passwd` è quello che esso acquisisce per *default* al momento del *login*, tuttavia un utente può acquisire i permessi di gruppi diversi mediante il comando *newgroup*.

Il *file* `/etc/password`, oltre ad associare uno *username* col rispettivo UID, contiene anche un profilo dell'utente, composto nell'ordine dai seguenti campi separati dal carattere :

- *username* associato all'utente;
- *password* (codificata);
- UID dell'utente;

- GID con cui l'utente opera di *default* (si tratta di un indice nella tabella dei gruppi `/etc/group`);
- Gecos, ovvero i dati anagrafici dell'utente, modificabili mediante il comando `chfn`;
- *home directory*, ovvero la *directory* di lavoro propria dell'utente, su cui esso dispone dei permessi di scrittura necessari per creare propri *file*;
- comando da eseguire al *login* sul sistema dell'utente. Generalmente si tratta di un interprete di comandi (*shell*), ma può essere qualunque comando. Molti sistemi per sicurezza limitano il contenuto di questo campo ai soli programmi listati in `/etc/shells`.



Tipologie di utenti

L'utente amministratore del sistema è quello con UID=0, a cui è associato lo *username root*, detto anche superutente o *superuser*. Esso può accedere a qualunque risorsa del sistema, anche non essendone il proprietario.

Poiché *root* dispone pressoché di tutti i permessi, è buona norma evitarne l'utilizzo nella pratica comune, se non per le operazioni strettamente connesse alla manutenzione del sistema. Questo sia per ridurre il rischio di compiere errori che possano danneggiare il sistema, sia per evitare di eseguire senza protezioni eventuali programmi difettosi oppure cavalli di Troia.

L'utilizzo di un *account* non privilegiato consente invece di limitare eventuali danni ai soli *file* su cui l'utente dispone dei permessi di scrittura.

UNIX mette a disposizione i seguenti meccanismi per porre dei limiti alle azioni degli utenti nel sistema:

- quote di disco: permettono di limitare lo spazio su disco a disposizione di un singolo utente o di gruppi di utenti. Mediate il comando `edquota` è possibile porre per ogni utente dei limiti sia sul numero di *file* che sulla occupazione di disco, per ognuno dei *filesystem* presenti sul sistema. È possibile definire dei limiti assoluti, oltre i quali non viene più permessa la scrittura, oppure dei limiti *soft* che possono essere superati per un certo periodo di tempo in caso di necessità (ad esempio durante lo scaricamento o la compilazione di un *software*).
- Limitazione nell'utilizzo della CPU: il comando `ulimit`, che può essere eseguito automaticamente al *login* sul sistema dell'utente mediante lo *script* `/etc/profile`, permette di controllare l'utilizzo delle risorse di calcolo da parte di un utente. È possibile definire il massimo numero di processi lanciabili all'interno di una sessione di lavoro, il numero degli accessi contemporanei, la massima dimensione di un programma caricabile in memoria, la quantità di memoria utilizzabile, il tempo macchina utilizzabile, nonché altri parametri.

Gestione dei permessi

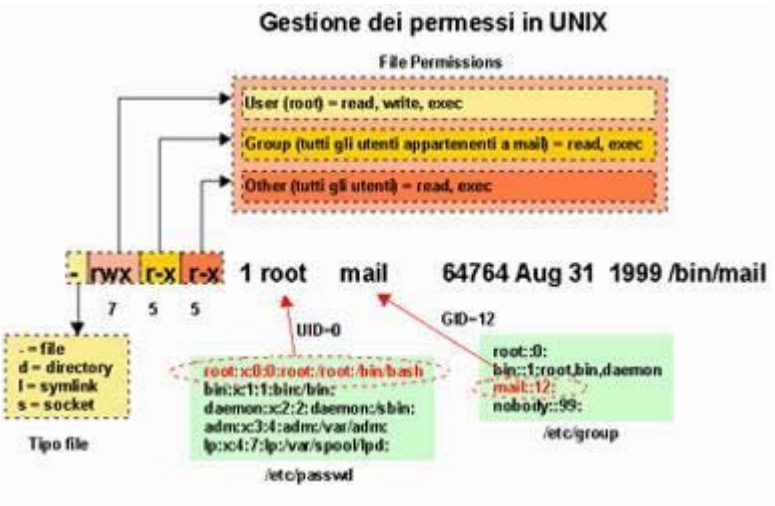
In *UNIX* la gestione dei permessi dei file avviene assegnando ad un *file* un utente proprietario e un gruppo proprietario. Nella gestione interna vengono utilizzati i valori numerici di UID e GID e

questi vengono convertiti in valori simbolici solamente per convenienza dell'utente (ad esempio dal comando `ls`).

Al *file* è anche associata una serie di bit che identificano i seguenti permessi (esistono ulteriori permessi speciali che non tratteremo in questa sede):

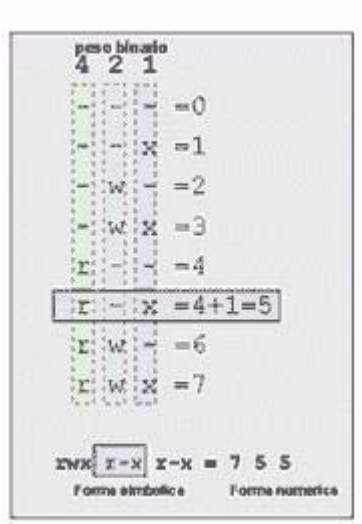
- lettura (r);
- scrittura (w);
- eseguibilità (x).

Il concetto di eseguibilità varia a seconda che esso sia applicato ad un *file* oppure ad una *directory*. Nel primo caso indica che si tratta di un programma eseguibile o di uno *script*. Nel secondo invece x assume il significato di permesso di accesso alla *directory*, da non confondere col permesso di lettura, che continua ad essere gestito mediante r.



I tre permessi vengono applicati ai seguenti gruppi di utenti:

- proprietario del *file* (u);
- utenti appartenenti allo stesso gruppo del proprietario (g);
- tutti gli altri utenti (o).



Esistono due metodi per rappresentare i permessi:

- simbolico: viene utilizzato ad esempio nell'*output* del comando `ls` e consiste nello scrivere in successione i tre gruppi di permessi (utente, gruppo, altri) con il loro valore letterale, sostituendolo con il simbolo - se il permesso non è attivo. Ad esempio rwx rw- r-x.
- Numerico: i gruppi di permessi di cui sopra vengono interpretati come numeri binari e successivamente vengono trasformati in decimale. Il permesso precedente rwx rw- r-x diventa pertanto 111 110 101, ovvero 765.

Le chiamate al sistema che accedono ai *file* confrontano lo UID e il GID con cui è marcato il *file* con quelli associati al processo che sta tentando di accedervi e, in base alla maschera dei permessi, decidono se concedere o meno l'accesso.

I principali comandi per gestire i permessi sui *file* sono i seguenti:

- `chown`, `chgrp`: gestione della proprietà di un *file*;
- `chmod`: gestione dei permessi;
- `su`, `newgrp`: utilizzo dei permessi di un altro utente o gruppo.

Ulteriori informazioni sugli utenti, sulla gestione dei permessi e sui relativi comandi di gestione esulano dagli scopi di questo capitolo e possono essere reperite su qualunque manuale di base di *UNIX*.

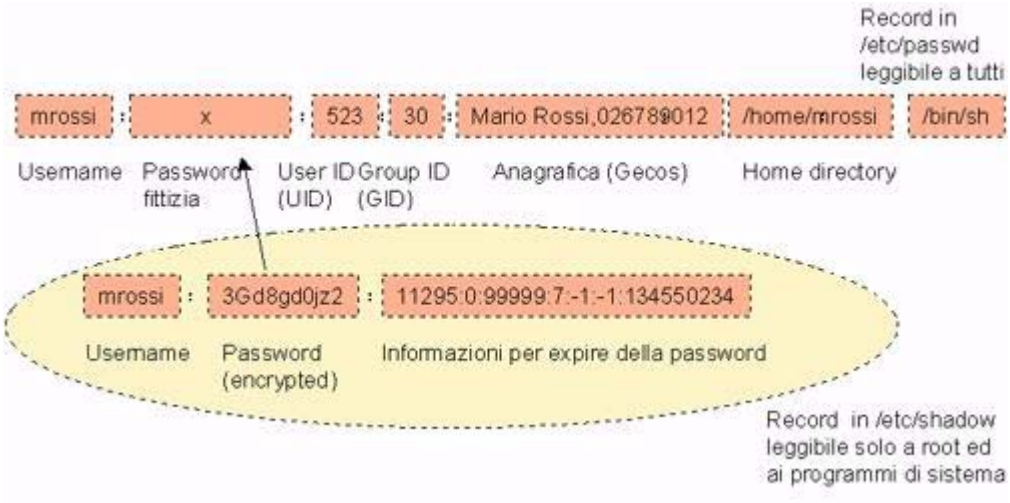
Shadow password

Tradizionalmente in *UNIX* l'autenticazione avviene assegnando ad ogni utente del sistema un *account* composto da una coppia *username/password*. Le informazioni sugli *account* vengono conservate nel *file* `/etc/passwd`, assieme alle *password* codificate. Al momento dell'accesso di un utente ad un servizio, ad esempio quando l'utente esegue il *login* in una sessione di lavoro interattiva sulla macchina, il codice contenuto nel programma che gestisce il servizio richiede lo username e la *password* dell'utente e, utilizzando la funzione di sistema `crypt()` genera la versione codificata della *password*, che viene confrontata con quella presente in `/etc/passwd`. Se le due *password* codificate coincidono, viene permesso l'accesso all'utente.

Essendo il confronto effettuato non sulle *password* in chiaro, bensì su quelle codificate, non è necessario che l'algoritmo di codifica sia bidirezionale: come standard viene utilizzata una versione modificata di DES, ma in alcuni sistemi è possibile scegliere altri tipi di codifiche come MD5.

Pertanto dalla *password* codificata non è possibile risalire a quella in chiaro, se non procedendo per tentativi, ad esempio provando a codificare una lista di *password* banali e confrontando il risultato con la *password* codificata che si trova in `/etc/passwd`. Tale metodo di attacco, difficile da realizzare ai tempi dei primi sistemi *UNIX*, è invece di fatto possibile utilizzando la potenza di calcolo dei calcolatori attuali, anche grazie alla poca lungimiranza di alcuni utenti nella scelta delle *password*.

La vulnerabilità del meccanismo delle *password* nei sistemi *UNIX* deriva dalla scelta progettuale di mantenere nello stesso *file* che contiene le *password* codificate anche i dati relativi agli *account*, che devono essere mantenuti leggibili a tutti gli utenti.



Non potendo limitare la lettura del *file* `/etc/passwd`, pena il malfunzionamento di alcuni programmi, per ovviare al problema alcuni anni or sono si è introdotta la variazione nota sotto il nome di *shadow password*. Essa consiste nel porre le *password* codificate nel *file* separato `/etc/shadow`, leggibile solamente a *root* ed ai programmi di sistema, continuando a mantenere in `/etc/passwd` solamente le informazioni non importanti per la sicurezza e sostituendo la *password* ivi contenuta con una etichetta senza valore. Contemporaneamente sono state aggiunte alcune funzionalità riguardanti la gestione della scadenza delle *password*.

Questo trucco ha permesso di risolvere il problema con il minimo impatto per il sistema. In particolare, non essendo stato intaccato il formato generale di `/etc/passwd`, tutti i programmi che accedevano a questo *file*, per ottenere dati che non fossero le *password*, hanno continuato a funzionare tranquillamente.

Si sono invece dovuti modificare per adeguarli al nuovo metodo tutti i programmi con necessità di eseguire delle autenticazioni, comprese molte utilità di sistema (*login*, servizi di rete, ...) e programmi applicativi.

Le *shadow password* vengono gestite mediante un insieme di *utilities* standard:

- `pwconv`, `pwunconv`: conversione da/a `/etc/password` a *shadow password*;
- `pwck`, `grpck`: verifica *password* e gruppi;
- `useradd`, `usermod`, `userdel`: metodi *industry*-standard per la gestione degli utenti;
- `groupadd`, `groupdel`, `groupmod`: metodi *industry*-standard per la gestione dei gruppi;
- `gpasswd`: gestione del *file* `/etc/group`.

Pluggable authentication module

Oltre al passaggio da *password UNIX* tradizionali a *shadow password*, vi sono anche altre occasioni in cui vi è la necessità di variare il metodo di autenticazione:

- modifica della *policy* di sicurezza (esempio: restrizioni nel formato delle *password*);
- utilizzo di una codifica crittografica più sicura (MD5);
- autenticazione mediante un *server* centralizzato (**NIS**, **NIS+**, **LDAP**, ...);
- autenticazione mediante altre tecniche (S/*keys*, *ticket kerberos*, ...);
- integrazione con altri sistemi operativi (esempio: autenticazione mediante *server* NT);
- applicazioni particolari (esempio: autenticazione di accessi via modem mediante Radius).

Per evitare di dover ogni volta sostituire un numero anche significativo di programmi e utilità di

sistema con le nuove versioni in grado di utilizzare i nuovi metodi di autenticazione, è stato introdotto alcuni anni or sono il sistema PAM (*Pluggable Authentication Modules*), il quale consente una gestione modulare dei metodi di autenticazione.

Utilizzando PAM il codice del programma comprende al proprio interno solamente una libreria di funzioni indipendenti dai possibili metodi di autenticazione. La parte di codice che implementa i metodi di autenticazione risiede invece su dei moduli a parte, che nella pratica sono costituiti da DLL (*shared libraries*) che vengono collegate al programma in modo dinamico.

In questo modo si evita la proliferazione di versioni dei programmi specifiche per i diversi metodi di autenticazione usati e nel contempo si rende il *software* più semplice da sviluppare e meno soggetto a variazioni. Eventuali correzioni nel codice relativo ad un determinato metodo di autenticazione coinvolgono solamente il modulo PAM che implementa e non il programma applicativo. L'eliminazione di codice ridondante e la delimitazione delle funzioni riguardanti l'autenticazione su un numero ristretto di *file* facilmente aggiornabili, consente di identificare più chiaramente le aree in cui possono verificarsi eventuali problemi di sicurezza ed offre una migliore possibilità di intervento.

Condivisione di file in Linux - NFS

Il protocollo **NFS** (*Network File System*), originariamente sviluppato da *Sun Microsystems*, permette di condividere (esportare) *filesystem*, ossia porzioni di disco, verso altre macchine *UNIX*. Applicazioni possibili sono la condivisione delle *home directory* degli utenti fra un gruuppo di macchine oppure la realizzazione di sistemi *diskless*, in cui il *filesystem* di *root*, contenente il sistema operativo e tutte le impostazioni di sistema, viene montato da un *server* di rete.

In *Linux* per utilizzare la parte *client* è sufficiente avere un *kernel* compilato con il supporto per **NFS**. Il *mounting* avviene mediante un comando del tipo:

```
# mount -t nfs server01:/disk2 /test
```

dove *server*01 è il nome del *server* e /disk2 la *directory* da montare. Il parametro -t è opzionale. È possibile automatizzare il *mount* utilizzando la tabella `/etc/fstab`, nella quale vanno inseriti i comandi di *mount* da eseguire all'avvio del sistema.

Per quanto riguarda la parte *server*, **NFS** utilizza come trasporto il protocollo RPC (*Remote Procedures Call*). È pertanto necessario verificare di avere attivo il *portmapper* e i demoni `rpc.mountd` (gestore del montaggio) e `rpc.nfsd` (gestore delle richieste dei *client*).

È possibile definire i *filesystem* da esportare mediante la tabella `/etc/exports`, secondo la seguente sintassi:

directory indirizzo(opzioni)

Gli indirizzi delle macchine a cui permettere il *mount* dei *filesystem* possono essere specificati anche mediante espressioni regolari. Ad esempio, per esportare in lettura e scrittura la *directory* /disk2 verso una singola macchina si può utilizzare la seguente linea:

```
/disk2 lnxserver001(rw, no_root_squash)
```

L'opzione *no_root_squash* specifica che l'utente *root* del *client* deve essere mappato nell'utente *root* del *server* (di *default* verrebbe mappato nell'utente *nobody*).

Per esportare il CDROM a tutte le macchine della rete locale con indirizzi 192.168.10.x si può utilizzare la seguente linea:

```
/mnt/cdrom 192.168.10.0/255.255.255.0 (ro, insecure)
```

Dopo aver modificato `/etc/exports` è necessario riavviare il *server* **NFS**.

Automount

Nei sistemi *UNIX* ogni disco contenente un *filesystem* per essere accessibile deve essere montato in una *directory* mediante il comando *mount* (oppure al *boot* mediante `/etc/fstab`). `Autofs` permette di automatizzare tale l'operazione anche a sistema funzionante, in modo da non dover mantenere sempre montati i *filesystem* che non si utilizzano.

Per usufruire di tale funzionalità il *kernel* di *Linux* deve essere compilato con il supporto per `autofs` (in alternativa può essere caricato il modulo `autofs.o`). Inoltre sono necessari i programmi utente e il demone.

Tutte le richieste di accesso effettuate all'interno di una determinata *directory* (*master-directory*) vengono intercettate dal sistema e se la richiesta in questione riguarda una *subdirectory* presente nella configurazione del servizio *automount* ma attualmente non montata, questa viene montata automaticamente nel *filesystem*, in modo trasparente all'utente.

Anche l'operazione di rilascio viene gestita in automatico non appena la *directory* risulti inutilizzata per un certo periodo di tempo. Il *filesystem* da montare può essere un disco locale, un dispositivo rimovibile (cdrom, *floppy*) o un volume condiviso da un *server* di rete.

La configurazione di *automount* consta di un *file* di configurazione principale `/etc/auto.master` che contiene la lista delle *master-directory* nel formato

```
<master-dir> <file configurazione> <opzioni>
```

Per ogni *master-directory* viene definito il relativo *file* di configurazione ed eventuali opzioni, ad esempio *--timeout*=xxx con la quale si imposta il tempo di inutilizzo del *filesystem* prima che questo venga automaticamente smontato.

I *file* con configurazione di ogni *master-directory* ha il seguente formato:

```
<dir> <options> <fylesystem>
```

in cui per ogni *subdirectory* si specificano le opzioni da passare al comando *mount*, ad esempio:

```
floppy -fstype=auto :/dev/floppy
```

```
cdrom -fstype=iso9660,ro :/dev/cdrom
```

Amministrazione di un sistema Windows

In questa sezione affronteremo le problematiche principali relative alla gestione degli utenti in un ambiente costituito da un singolo Dominio *Microsoft Windows* 2000.

Inizieremo parlando di *Account* Utente. Vedremo come creare e configurare un *Account* Utente di Dominio, quali sono le sue principali proprietà e daremo le linee guida per la generazione dei *logon*

name e delle *password*. Definiremo le *Home Folder* ed i Profili Utente come meccanismo per la personalizzazione dell'ambiente dell'utente.

Parleremo poi di Gruppi di Utenti definendo i Tipi di Gruppi (Sicurezza e Distribuzione) ed il loro *Scope* (Globali, Locali di Dominio ed Universali) e analizzeremo i Gruppi Predefiniti. Descriveremo, infine, la strategia di applicazione. Passeremo poi a parlare di risorse e di come renderle disponibili sulla rete. Inizieremo parlando di condivisione di cartelle. Vedremo come creare una cartella condivisa, come specificare le proprietà di condivisione, quali sono i Permessi di Condivisione e quali sono le regole per la loro applicazione nel caso di permessi multipli.

Relativamente alla protezione locale delle risorse parleremo di permessi NTFS per *files* e cartelle. Ne definiremo le varie tipologie, le regole di applicazione in caso di permessi multipli e il modo in cui si integrano con i permessi di condivisione.

Account utente

Un *Account* Utente è un oggetto che contiene quelle che sono le credenziali uniche per ogni utente e che gli permettono di accedere ad una macchina e alle risorse del dominio a lui permesse.

Le tipologie di *Account* Utente presenti in *Windows* 2000 sono:

Local user account

Permette ad un utente di effettuare il *log on* su uno specifico *computer* ed accedere alle risorse di tale *computer*. Per accedere alle risorse di altri *computer* bisogna utilizzare un *account* definito su tali *computer*, poiché questa tipologia di *account* risiede nel *Security Accounts Manager* (SAM) di ogni *computer*.

Domain user account

Permette ad un utente di effettuare il *log on* sul dominio da una qualsiasi macchina del dominio ed accedere a tutte le risorse del dominio indipendentemente dal *server* che le ospita. Tale *account* risiede nell'*Active Directory directory service*.

Built-in user account

Permette ad un utente di eseguire processi amministrativi o avere accesso temporaneo alle risorse di rete. Esistono due *account* di questo tipo: *Administrator* e *Guest* che risiedono rispettivamente nel SAM o in *Active Directory* che si parli rispettivamente di *account Built-in* Locali o di Dominio. Questi *account* vengono creati automaticamente durante l'installazione di *Windows* 2000 e di *Active Directory*.

Per una gestione corretta, efficace e sicura degli *Account* Utente è bene avere innanzitutto le idee chiare circa ciò che riguarda:

- Le Regole per la generazione dei Nomi.
- La gestione delle Password.
- Le Opzioni che è possibile specificare per un *Account* Utente.

Poiché le scelte che vengono operate in tal senso, possono poi impattare notevolmente su quella che è la gestione di tali oggetti successiva alla loro creazione.

Creazione di account utente

Gli *Account* Utente di Dominio permettono ad un utente di effettuare *log on* sul dominio da una qualsiasi postazione appartenente al dominio e di accedere ad una qualsiasi risorsa indipendentemente da dove essa sia localizzata e previa autorizzazione. Tale *Account* Utente viene creato su un Controllore di Dominio e da questi replicato a tutti gli altri Controllori dello stesso dominio.

Windows 2000 fornisce tutti gli strumenti necessari per creare e gestire tale tipologia di *Account* Utente, e tali strumenti vengono installati di *default* su ogni Controllore di Dominio. Tuttavia è possibile installare tali strumenti su qualsiasi *computer* che esegua almeno *Microsoft Windows* 2000 *Professional*.

Lo strumento utilizzato per la creazione e successiva gestione degli *Account* Utente di Dominio è *Active Directory Users and Computers* che è, ovviamente, situato in *Start\Programs\Administrative Tools*. Di *default* gli *Account* Utente di Dominio vengono creati nel contenitore *Users* ma possono essere creati anche in qualsiasi altra Unità Organizzativa che l'amministratore abbia deciso di creare.

Tra le caratteristiche di base che vengono definite durante la creazione di un *account*, oltre alle cose indispensabili come il *Logon Name*, ricordiamo la strategia di controllo delle *password* e l'associazione all'*account* di una Cartella Personale (*Home Folder*).

Dunque, per creare un *Account* Utente di Dominio vanno eseguiti i seguenti passi:

- Eseguire *Active Directory Users and Computers* in *Start\Programs\Administrative Tools*
- Cliccare con il tasto destro sul contenitore che conterrà l'*Account* Utente di Dominio, selezionare *New* e scegliere *User*.
- Di seguito sono descritte le opzioni che è possibile configurare:
    - *First name*: Il nome dell'utente.>
    - *Initials*: Le iniziali dell'utente. Tale campo non è obbligatorio.
    - *Last name*: Il cognome dell'utente.
    - *Full name*: Il nome completo dell'utente. Tale nome deve essere unico nel contenitore in cui stiamo creando l'*account*. *Windows* 2000 completa questo campo automaticamente con le opzioni già specificate in *First name* e *Last name* e costituisce ciò che viene visualizzato in *Active Directory*.
    - *User logon name*: Il nome che l'utente utilizzerà per effettuare il *logon*, definito in base alle regole di generazione dei Nomi stabilite. Deve essere unico in tutta *Active Directory*.
    - *User logon name* (pre-*Windows* 2000): Il nome che verrà utilizzato per effettuare il *logon* da macchine precedenti a *Microsoft Windows* 2000. Tale campo è obbligatorio e deve essere unico in tutta *Active Directory*.

Selezionando *Next* si passa a definire la *password* e le regole inerenti la sua gestione.

Cliccando *Next* e poi *Finish* si termina il processo di creazione di un *Account* Utente di Dominio.

Proprietà di un account utente

Una volta che l'*account* utente è stato creato e sono state definite le sue caratteristiche principali, è possibile definire quelle che sono le proprietà dell'*Account*, le opzioni di *logon*, le proprietà di accesso remoto e le proprietà personali.

La finestra di dialogo *Properties* relativa ad uno specifico *account* utente consente di specificare tutte le caratteristiche di tale *account*. Tali informazioni sono memorizzate in *Active Directory*.

Tramite queste informazioni è possibile ricercare un *account* utente in *Active Directory* basandosi su una sua qualche caratteristica, come ad esempio l'indirizzo, l'ufficio di appartenenza, ed altro.

Per modificare ed impostare quelle che sono le proprietà di un *account* utente, utilizzare lo strumento *Active Directory Users and Computers* in *Start\Programs\Administrative Tools*, selezionare il contenitore che contiene l'*account* che si vuole configurare, fare click con il tasto destro del *mouse* su tale *account* utente e scegliete *Properties*.

Si avranno a disposizione le seguenti schede:

- *General*. Permette di specificare il nome dell'utente, una descrizione, l'indirizzo dell'ufficio, il numero di telefono, e le informazioni relative alla posta elettronica ed alla *home page*.
- *Address*. Permette di specificare l'indirizzo dell'utente, la città, lo stato o la provincia, il CAP.
- *Account*. Permette di specificare il *logon name*, la scadenza dell'*account*, le opzioni relative alla *password*, le impostazioni di *logon* ed altre opzioni.
- *Profile*. Permette di assegnare all'utente un Profilo *Roaming* ed una *Home Folder*.
- *Telephones*. Raggruppa tutte le informazioni relative ai recapiti telefonici dell'utente e permette di specificare una descrizione.
- *Organization*. Specifica il titolo dell'utente, la società per cui lavora, il dipartimento ed i suoi superiori.
- *Member Of*. Permette di specificare i gruppi cui l'utente appartiene.
- *Dial-in*. Permette di impostare i permessi, le opzioni di *callback*, l'assegnazione di un indirizzo *IP* statico e di *routes* statiche quando l'utente si connette da remoto.
- *Environment*. Permette di specificare quali applicazioni partono e a quali *devices* ci si connette quando parte *Terminal Services*.
- *Sessions*. Specifica i settaggi di *terminal Services* relativi a questo utente se ciò è permesso.
- *Remote control*. Specifica i settaggi della funzionalità di controllo remoto di *terminal Services Specifies* relativi a questo utente se ciò è permesso.
- *Terminal Services Profile*. Specifica il profilo di questo utente per *Terminal Services*

Impostare le Opzioni di *Logon* di un *account* utente vuol dire controllare in quali giorni ed in quali ore l'utente può accedere alle risorse del dominio e quali postazioni può utilizzare.

Questi settaggi si impostano utilizzando lo strumento *Active Directory Users and Computers* in *Start\Programs\Administrative Tools*, selezionando il contenitore che contiene l'*account* che si vuole configurare, facendo click con il tasto destro del *mouse* su tale *account* utente, scegliendo *Properties* e selezionando la scheda *Account*.

Di *default*, l'utente si può connettere a risorse di dominio 24 ore su 24, 7 giorni su 7. In ambienti ad elevata sicurezza è bene restringere tale possibilità, ad esempio in ambienti in cui sono definiti turni di lavoro.

Per impostare le *Logon Hours*:

Dalle proprietà dell'*account* selezionare la scheda *Account* e scegliere il pulsante *Logon Hours*

Una tabella suddivisa nei sette giorni della settimana e nelle 24 ore di una giornata indica quando l'utente può accedere a risorse di dominio (box BLU) e quando no (box BIANCO). Utilizzando i pulsanti *Denied* e *Permitted* è possibile modificare lo stato dei vari box.

Attenzione poiché quando le *Logon Hours* dell'utente si esauriscono, le connessioni che nel frattempo ha stabilito a risorse di dominio non vengono perse, ma non ne può creare delle altre.

Di *default* un utente in possesso di un *account* valido può fare *log on* al dominio da qualsiasi

postazione, ad eccezione dei Controllori di Dominio.

In ambienti ad elevata sicurezza, in cui sulle stazioni di lavoro ci sono memorizzati dati sensibili, potrebbe essere necessario controllare quali sono le macchine da cui l'utente può effettuare un *log on* al dominio.

Per specificare le stazioni da cui l'utente può fare *logon*:

Dalle proprietà dell'*account* selezionare la scheda *Account* e scegliere il pulsante *Log On To*.

Selezionando *The following computers* è possibile creare l'elenco delle macchine da cui l'utente può effettuare il *log on* scrivendone il nome in *Computer name* e cliccando poi *Add*. In *Windows* NT 4.0 tale lista era limitata a 8 nomi.

Copia di un account

Per semplificare il processo di creazione degli *account* utente è possibile ricorrere alla copia di un *account* utente esistente.

Durente la copia molte delle proprietà dell'*account* esistente vengono ereditate dal nuovo *account* e questo fa in modo di non essere costretti a ripetere per ogni *account* le stesse informazioni.

È possibile solo copiare *account* residenti su un Controllore di Dominio.

Vediamo quali sono le proprietà che vengono ereditate dal nuovo *account* utente dopo la copia:

- *General*. Nessuna.
- *Address*. Tutte, eccetto *Street Address*.
- *Account*. Tutte eccetto *Logon Name*.
- *Profile*. Tutte eccetto *Profile path* e *Home folder* che vengono modificate per rispecchiare il nuovo valore del *logon name*.
- *Telephones*. Nessuna.
- *Organization*. Tutte, eccetto *Title*.
- *Member Of*. Tutte.
- *Dial-in*. Nessuna.
- *Environment*. Nessuna.
- *Sessions*. Nessuna.
- *Remote control*. Nessuna.
- *Terminal Services Profile*. Nessuna.

Attenzione, poiché diritti e permessi dati direttamente all'*account* esistente, non vengono ereditati dal nuovo *account*.

Per fare la copia di un *account*:

- Utilizzare lo strumento *Active Directory Users and Computers* in *Start\Programs\Administrative Tools*, selezionare il contenitore che contiene l'*account* che si vuole configurare, fare click con il tasto destro del *mouse* su tale *account* e scegliere *Copy*.
- In *Copy Object - User* specificare il *logon name*, lo *user name* e le nuove informazioni per il nuovo *account* e poi selezionare *Next*.
- Impostare la *password* e, se è il caso, disabilitare l'opzione *Account is disabled* e selezionare *Next*.

Verificare che le informazioni immesse sono corrette e selezionare *Finish*.

Gruppi di utenti

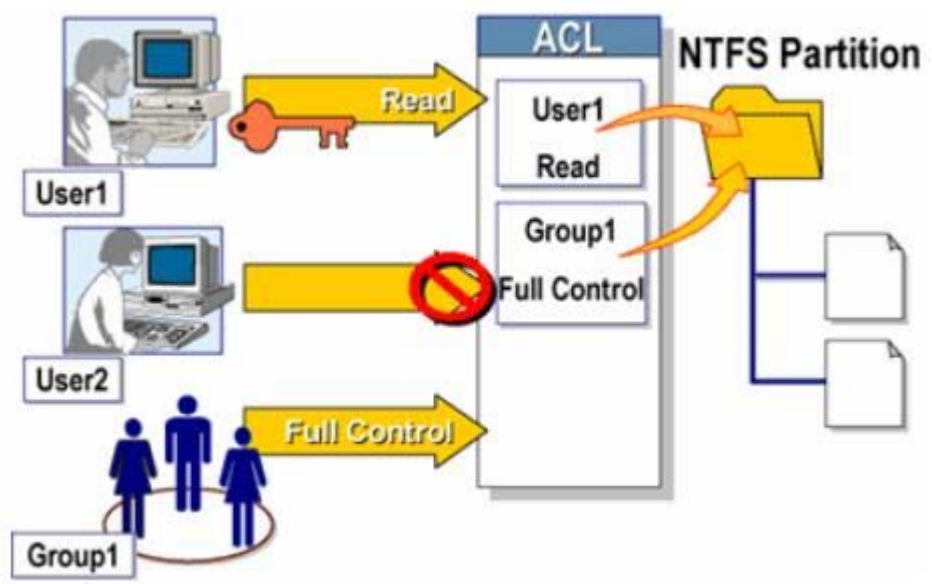
Così come in *Unix* anche in *Windows* gli *account* utente possono essere raggruppati in Gruppi di utente. Un Gruppo è un insieme di *account* utente.

Tramite un utilizzo opportuno dei gruppi risulta più agevole la concessione di permessi di accesso alle risorse e diritti agli utenti: tramite i gruppi riusciamo ad assegnare simultaneamente diritti e permessi a più utenti. Infatti se assegniamo un permesso o un diritto ad un gruppo, automaticamente essi vengono garantiti a tutti gli appartenenti al gruppo.

Quando aggiungiamo membri ad un gruppo, valgono le seguenti considerazioni:

- Quando un utente entra a far parte di un gruppo, automaticamente eredita tutti i permessi ed i diritti assegnati al gruppo.
- Se un utente viene reso parte di un gruppo mentre ha una sessione attiva, l'appartenenza a tale gruppo non ha effetto finchè l'utente non effettua un *log off* ed un successivo *log on*.
- Un utente può essere membro di più gruppi contemporaneamente.

Protezione su NTFS



Sulle partizioni NTFS, *Windows* 2000 permette di specificare Permessi NTFS per *files* e cartelle.

A differenza dei permessi di condivisione che valgono solo nel caso di accessi via rete e si possono specificare solo per cartelle, i permessi NTFS valgono localmente e si possono specificare per le cartelle ma anche per il singolo *file*.

È possibile specificare anche i Permessi NTFS sia per il singolo utente che per gruppi di utenti ed anche per loro è consigliata la strategia A G DL P.

NTFS permette di associare ad ogni *file* e cartella una *access control list* (**ACL**) che contiene la lista degli utenti, dei gruppi e dei *computer* che hanno accesso a tale risorsa ed il tipo di accesso concesso.

Affinchè un utente possa accedere a tale risorsa, la sua **ACL** deve contenere almeno una *access*

control entry (**ACE**) relativa al particolare utente o ad uno dei gruppi cui appartiene. In caso contrario l'accesso viene negato.

Tramite i Permessi NTFS specificati su una cartella è possibile controllare chi e come può accedere alla cartella ed ai *files* e sottocartelle in essa contenuti. In dettaglio, abbiamo a disposizione i seguenti Permessi NTFS per una Cartella:

NTFS folder permission Allows the user to

- *Read*. Vedere *files* e cartelle contenute in tale cartella e vedere i suoi attributi, *ownership* e permessi.
- *Write*. Creare nella cartella nuove cartelle e *files*, modificarne gli attributi e vedere *ownership* e permessi.
- *List Folder Contents*. Vedere i nomi dei *files* e delle cartelle contenute nella cartella.
- *Read & Execute*. Navigare attraverso le cartelle, lanciare eseguibili più i permessi di *Read* e *List Folder Contents*.
- *Modify*. Eliminare la cartella più i permessi di *Write* e *Read & Execute*.
- *Full Control*. L'unione di tutti i permessi precedenti.

Tramite i Permessi NTFS per un *File* è possibile controllare in dettaglio chi e come può accedere a tale *file*. Ovviamente tali permessi, se specificati, hanno precedenza rispetto a quelli che il *file* eredita dalla carella che lo contiene. In dettaglio, abbiamo a disposizione i seguenti Permessi NTFS per una Cartella:

- *Read*. Leggere il *files* e visualizzarne gli attributi, l'*ownership* ed i permessi.
- *Write*. Modificare il *file*, modificare gli attributi e vedere *ownership* e permessi.
- *Read & Execute*. Eseguire applicazioni più i permessi di *Read*.
- *Modify*. Modificare ed eliminare il *file* più i permessi di *Write* e *Read & Execute*.
- *Full Control*. L'unione di tutti i permessi precedenti.

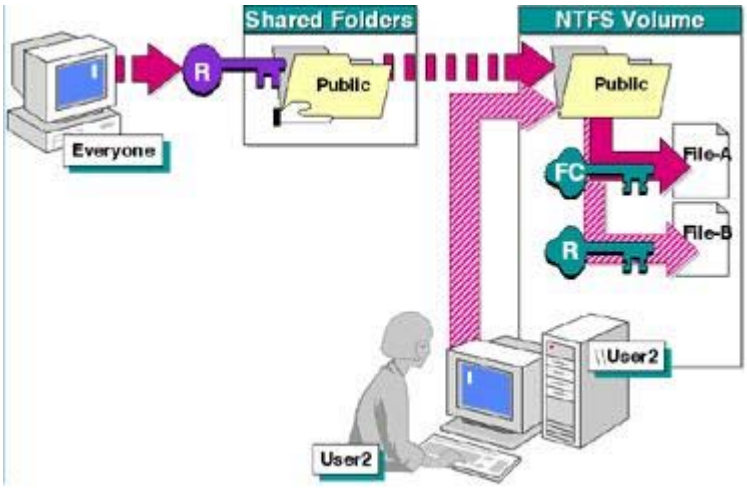
Quando una partizione o un volume viene formattato NTFS, *Windows* 2000 automaticamente assegna sulla *root* il permesso NTFS *Full Control* al gruppo *Everyone*, che dunque avrà di *default* tale permesso NTFS su tutti i *files* e cartelle create in quella partizione o volume, a meno che non diversamente specificato.

Interpretazione dei permessi

Nell'interpretare i permessi, potrà capitare di incombere permessi contraddittori: ad esempio in una riga si consente l'accesso per un utente al quale era stato negato l'accesso del gruppo di cui appartiene. Il seguente è l'imperativo in grado di sciogliere l'equivoco:

The Most Restrictive Permission Is the Effective Permission

letteralmente il permesso più restrittivo, è quello effettivo. Quindi in caso di contrasto di due ergole appartenenti alla stessa lista, prevarrà la scelta della regola più restrittiva.



Linea guida nell'assegnazione di permessi NTFS:

- Rimuovere il permesso *Full Control* da gruppo *Everyone*.
- Assegnare il permesso *Full Control* al gruppo *Administrators*.
- Assegnare al *Creator Owner Full Control* delle sue cartelle dati.
- Educare gli *Users* nell'assegnare i permessi NTFS ai propri *file*.

Home directory:

- Creare una cartella centrale denominata *Users*.
- Condividere la cartella *Users*.
- Rimuovere il permesso *Full Control* dal gruppo *Everyone* ed assegnarlo al gruppo *Users*.
- Usare la variablie d'ambiente %*Username*% per creare le *home directory*.

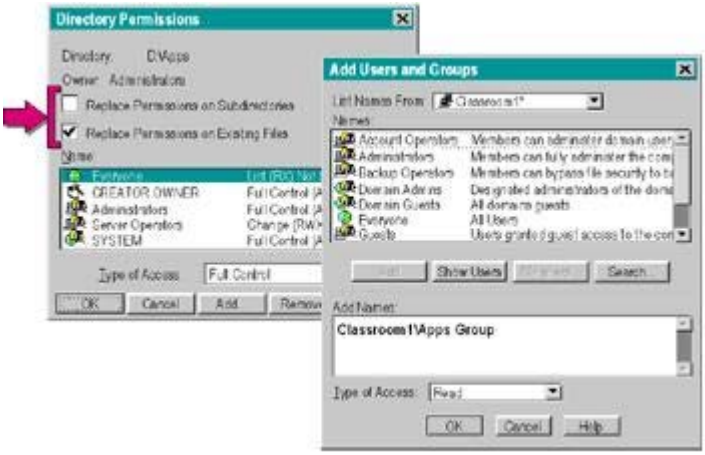
Assegnazione dei permessi

Per poter assegnare i permessi NTFS ad un qualsiasi ggetto si devono rispettare alcuni requisiti fondamentali, quali:

- essere *Owner*;
- godere dell'insieme di permessi *Full Control* o quantomeno *Special Access* (*Change Permission or Take Ownership*).

Permessi NTFS di *default*

Il gruppo *Everyone* viene automaticamente assegnato con permessi *Full Control* ed i nuovi file ereditano i permessi della cartella nella quale vengono creati.



Best Practices

- Assegnare i permessi NTFS prima di condividere le risorse.
- Rendere gli eseguibili degli applicativi *Read-Only* per tutti gli *Users*.
- Utilizzare la variabile d'ambiente %*Username*% per creare le *directory home* per gli utenti.
- Assegnare all'*account Creator Owner* i permessi *Full Control* per le cartelle dati.
- Usare nomi lunghi solo se si accede alle risorse localmente.

Condivisione di file in Windows

Le Cartelle Condivise sono uno strumento per permettere agli utenti di accedere al contenuto di cartelle tramite la rete.

Le cartelle condivise possono contenere applicazioni, dati pubblici e dati personali, permettendo quindi un'amministrazione centralizzata di tali informazioni.

È evidente come sia necessario però poter controllare con una certa accuratezza chi e come accede a tali cartelle, cioè occorre poter specificare dei Permessi di Condivisione relativamente ad un utente o ad un gruppo di utenti.

Per condividere una cartella bisogna innanzitutto appartenere ad uno dei gruppi che hanno il diritto di poter condividere cartelle relativamente al *computer* in cui la cartella risiede.

Quando condividiamo una cartella, dobbiamo inoltre poter controllare gli accessi a tale cartella assegnando opportuni permessi a ben precisi utenti e gruppi. Inoltre, potrebbe essere necessario limitare il numero di accessi contemporanei che è possibile effettuare a tale condivisione. Infine, in qualsiasi istante dobbiamo essere in grado di interrompere la condivisione, modificare il nome di condivisione, modificare i permessi.

Creare cartelle condivise

Quando si condivide una cartella:

- si assegna ad essa il nome di condivisione;
- si assegna eventualmente un commento che ne descriva il contenuto ed il proposito della condivisione;
- si limita eventualmente il numero di utenti che si possono connettere contemporaneamente a tale condivisione.
- si assegnano i permessi a gruppi di utenti o singoli utenti.

- opzionalmente, è possibile condividere la stessa cartella più volte.

Per effettuare queste operazioni, selezionare la cartella in *Windows Explorer* e cliccando con il tasto destro scegliere *Sharing*.

Utilizzando la scheda *Sharing* è possibile specificare le seguenti opzioni:

- *Share this folder*. Selezionare tale opzione per condividere la cartella.
- *Share name*. Specificare il nome che verrà assegnato alla condivisione e che verrà utilizzato dall'utente e dalle applicazioni ogni qualvolta intendano accedere ad essa. Di *default* viene proposto il nome della cartella. Tale campo è obbligatorio. Nella scelta di tale nome, tenere comunque conto che alcuni *client* potrebbero non essere in grado di visualizzare tutti i caratteri utilizzati.
- *Comment*. Descrizione, opzionale, che ha lo scopo di identificare il contenuto e/o lo scopo della condivisione.
- *User Limit*. Specificare il numero massimo di accessi concorrenti permessi a questa condivisione. Tale campo non è obbligatorio, poichè il valore di *default Maximum Allowed* fissa a 10 il numero massimo permesso per *Windows* 2000 *Professional* e nel caso di *Windows* 2000 *Server* è pari al numero di licenza acquistate.
- *Permissions*. Tale pulsante permette di assegnare permessi di accesso a tale condivisione ad utenti singoli o gruppi di utenti. Tale opzione non è obbligatoria poiche di *default* viene assegnato il permesso di *Full Control* al gruppo speciale *Everyone*.

Permessi di accesso alla condivisione

Ad ogni utente, gruppo di utenti o *computer* può essere garantito o negato il permesso di accedere alla condivisione, ed in ogni caso tale assegnazione è valida solo per accessi da remoto e non per accessi locali.

Dunque i permessi di condivisione permettono di proteggere la risorsa per accessi da remoto ma non per accessi locali, inoltre essi si applicano a cartelle e non ai singoli *files* (valgono per tutto ciò che è contenuto nella cartella).

È possibile assegnare i seguenti permessi:

- *Read*. Visualizzare i nomi delle cartelle, dei *files*, i dati contenuti nei *files*, gli attributi ed eseguire eseguibili.
- *Change*. Oltre a tutto ciò che è garantito da *Read*, è possibile creare cartelle, aggiungere *files*,

modificare *files*, modificare gli attributi, eliminare *files* e cartelle.
- *Full Control*. Oltre a tutto ciò che è garantito da *Change* è possibile modificare i permessi, acquisire la proprietà dei *files* (*Ownership*). Di *default* tale permesso è assegnato al gruppo speciale *Everyone*.

Per quanto detto appare immediatamente evidente come, se non in casi eccezionali, quando si crea una condivisione la prima cosa da fare è rimuovere i permessi assegnati di *default* al gruppo *Everyone* ed assegnare ad ogni utente o gruppo di utenti i permessi strettamente necessari affinchè possano eseguire la loro attività.

A tale proposito fare molta attenzione al fatto che i permessi di condivisione sono cumulativi: quando un utente accede ad una condivisione e in questa condivisione sono stati specificati sia permessi per l'utente che per alcuni o tutti i gruppi cui l'utente appartiene, allora il permesso risultante per quell'utente è la somma di tutti i permessi (in pratica il più ampio).

Tale regola di cumulatività ammette una sola eccezione: se solo uno dei permessi è *Deny*, allora tutti gli altri permessi vengono sovrascritti ed il permesso risultante sarà un *Deny*. Alla luce di ciò si raccomanda di utilizzare il *Deny* con molta parsimonia, praticamente esclusivamente per gestire le situazioni in cui un utente appartiene ad un gruppo che risulta avere dei permessi di condivisione per una certa risorsa e si vuole che tali permessi non siano validi per lo specifico utente.

Vediamo ora in dettaglio come impostare i permessi di accesso ad una condivisione, o come modificarli se già impostati.

È possibile assegnare permessi di condivisione per una cartella residente sia su una partizione o vulume formattati come **FAT**, che **FAT32** o NTFS. Si ricordi che, in quest'ultimo caso è necessario che l'utente abbia anche gli opportuni permessi NTFS.

Per assegnare i permessi di condivisione ad utenti, gruppi e *computer*:

Selezionare la cartella in *Windows Explorer* e cliccando con il tasto destro scegliere *Sharing*.

Nella scheda *Sharing* scegliere *Permission* ed accedere alla finestra di diaologo *Permissions*.

Selezionare *Add* ed in *Select User Groups or Computers* selezionare tramite *Look in* il dominio di cui interessa visualizzare gli utenti, i gruppi ed i *computers*.

Selezionare l'utente, o il gruppo o il *computer* che interessa tramite *Allow* impostare il permesso che interessa.

Per modificare le proprietà di una condivisione, selezionare la cartella in *Windows Explorer* e cliccando con il tasto destro scegliere *Sharing*. Di seguito le modifiche che è possibile apportare:

Do not share this folder.

Interrompere la condivisione. Tutti i settaggi, compresi i permessi, andranno persi.

Modificare il nome di condivisione.

Selezionare *Do not share this folder* per interrompere la condivisione, selezionare *Apply* e poi selezionare *Share this Folder* inserendo il nuovo nome di condivisione. Ovviamente nel frattempo avremo perso tutti i permessi precedentemente impostati.

Permissions.

Per modificare i permessi già impostati o aggiungerne di nuovi.

New Share.

Condividere nuovamente la cartella con un nome ed impostazioni diverse.

Remove Share.

Compare solamente se la cartella è stata condivisa più di una volta e permette di rimuovere una delle condivisioni.

Se quando selezioniamo *Do not share this folder* un qualche utente ha una connessione attiva, il sistema operativo ci avverte che un utente ha una connessione attiva e che se procediamo potrebbe perdere dei dati.

Condivisioni nascoste

Windows 2000 condivide automaticamente una serie di cartelle con lo scopo di permettere alcune funzionalità di sistema e permettere all'amministratore di svolgere le sue attività.

Tali condivisioni sono tutte caratterizzate dal fatto che il loro nome di condivisione termina con il simbolo $. Tale caratteristica impedisce che la condivisione venga visualizzata quando l'utente sfoglia l'elenco delle condivisioni di quel *server*, cioè rende tale condivisione una Condivisione Nascosta. Visto lo scopo per il quale sono concepite, sono conosciute come Condivisioni Amministrative.

Alcuni esempi di condivisioni nascoste sono tutte le unità logiche corrispondenti alle varie partizioni e volumi, la cartella di sistema, la cartella che contiene i *drivers* delle stampanti condivise, la condivisione utilizzata per i meccanismi di *interprocess communication* (IPC) utilizzati dai programmi:

C$, D$, E$, e così via ...

Tali condivisioni nascoste permettono all'amministratore di connettersi a qualsiasi partizione e volume di una macchina in modo tale da poter svolgere attività amministrative.

Admin$.

La cartella di sistema (C:\Winnt, di *default*). Permette di accedere alla cartella di sistema senza che sia necessario conoscere la cartella fisica corrispondente.

Print$.

Permette ai *client* di scaricare i *client* della stampante condivisa dal *server* su cui la condivisione risiede. Corrisponde al *path* fisico `Systemroot\System32\Spool\Drivers` e viene creata quando condividiamo la prima stampante. I gruppi *Administrators*, *Server Operators*, e *Print Operators* hanno *Full Control* mentre il gruppo *Everyone* ha *Read*.

IPC$.

Permette l'implementazione dei meccanismi IPC.

È possibile in ogni istante condividere ulteriori cartelle e fare in modo che siano condivisioni nascoste. Tali ulteriori condivisioni non risultano però essere delle condivisioni amministrative.

L'unico modo per accedere ad una condivione nascosta è indicare per esteso il suo *path* UNC, cioè `\\server\condivisione$`.

Servizi di directory

Un servizio di *directory* è una sorta di base dati distribuita di informazioni o di puntatori alle informazioni (per esempio, utenti, gruppi, risorse). Una volta implementata la *directory*, è possibile creare applicazioni distribuite che la utilizzano. Ad esempio l'elenco telefonico è un servizio di *directory* per la rete telefonica in cui la base dati contiene contiene i dati delle persone e le loro proprietà associate, che sono i numeri telefonici e l'indirizzo. Una semplice applicazione potrebbe ricavare il numero telefonico dato il nome dell'utente.

Il protocollo X.500 sviluppato dalla ISO è lo *standard* internazionale per i servizi di *directory*. Il protocollo X.500 specifica lo schema di *default* con cui costruire la base dati e descrive alcune classi di oggetti e i loro attributi associati. In particolare X.500 specifica che i dati (detti *Directory Information* Base o DIB) siano organizzati secondo un albero logico detto *Directory Information* Tree (DIT). Quindi le *directory* basate sul protocollo X.500 permettono di creare oggetti che contengono altri oggetti e supportano relazioni gerarchiche. L'albero delle informazioni viene costruito in base ai valori di attributi delle informazioni che possono essere di tipo assolutamente generale.

In una *directory* X.500 deve essere possibile reperire gli oggetti secondo schemi di ricerca, per esempio basati sui nomi degli attributi di un oggetto. Ogni elemento del DIB può essere ritrovato utilizzando due diverse notazioni:

- *Distinguished Name* (DN) che identifica univocamente l'elemento nell'albero;
- *Relative Distinguished Name* (RDN) che identifica univocamente l'oggetto all'interno di uno specifico contesto, ossia come elemento all'interno di una parte dell'albero.

X.500 specifica le regole con cui gli elementi del DIB possono essere ordinati, filtrati ed elaborati in genere. La ricerca di elementi all'interno del DIB viene svolta da un agente detto *Directory User Agent* (DUA) che accedere ai dati utilizza un protocollo specifico detto DAP (*Directory Access Protocol*).

Come vedremo una sua versione semplificata detta **LDAP** (*Lightweight* DAP) viene utilizzata per accedere a servizi di *directory* in *Linux* e *Windows*.

Ad esempio in un ambiente scolastico la *directory* del personale della scuola potrebbe essere un albero organizzato tramite attributi che identifichino le mansioni del personale stesso. Partendo dalla radice (*ROOT*) del DIT si potrebbe avere un primo attributo legato all'area operativa (A) ed un secondo legato all'attività specifica nell'ambito dell'area operativa (B) ed infine un terzo che identifica la persona per nome e cognome (C).

- *Root*
    - A=Amministrazione
    - A=Docenti
        - B=Italiano
            - C=Rossi
            - C=Bianchi
        - B=Matematica
            - ...

        - ...
    - A=Tecnici
    - ...

In questo DIT il Prof. Rossi può essere identificato utilizzando il *distinguished name* DN= {A=Docenti,B=Italiano,C=Rossi}, oppure all'interno della parte di albero relativo ai docenti di italiano dal relative *distinguished name* RDN={C=Rossi}.

Come vedremo la gestione di un sistema distribuito può avvalersi di uno o più *directory service* per la catalogazione dei nomi e degli attributi degli elementi del sistema, siano essi utenti, calcolatori o quant'altro.

Domain name system

Il *Domain Name System* (DNS) è uno dei più noti ed utilizzati servizi di *directory*. Esso risulta fondamentale per il buon funzionamento dell'attuale rete *Internet*.

Come noto l'instradamento a livello di protocollo *IP* si basa sull'indirizzo *IP* degli *host*. Per ragioni di comodità e di più facile memorizzazione gli esseri umani associano agli *host*, ed in particolare ai *server* dei nomi a parole.

Il DNS altro non è che un servizio di *directory* che permette di associare ad un nome simbolico di un *host* il suo numero *IP*.

Il DNS segue l'architettura di un servizio di *directory* vista precedentemente, per cui ha una organizzazione ad albero:

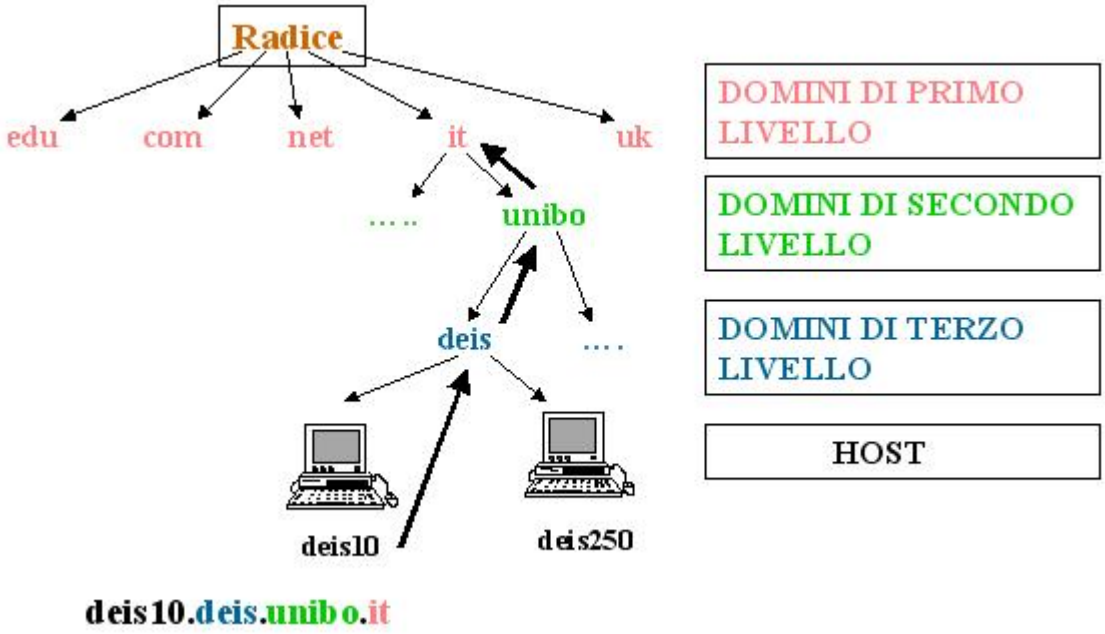
- Internet è partizionata in aree logiche dette domini con un nome univoco.
- I singoli domini possono a loro volta essere suddivisi in sottodomini.

Non esiste limite al numero di ripartizioni di un dominio o sottodominio.

Una rete *IP* appartiene solitamente ad un dominio o sottodominio e il suo posizionamento nell'albero della *directory* si riflette nella forma che assumono i nomi degli *host*.

I nomi sono composti da stringhe di caratteri separati da punti:

- la struttura dei nomi segue l'organizzazione gerarchica a partire da destra, dove si trova il nome di maggior valore (del dominio primario), seguito da quelli di valore sempre inferiore;
- le stringhe sono abbreviazioni convenzionali che indicano luogo fisico o ente di appartenenza dell'*host*;
- il numero di stringhe è virtualmente illimitato, al contrario del numero *IP*.



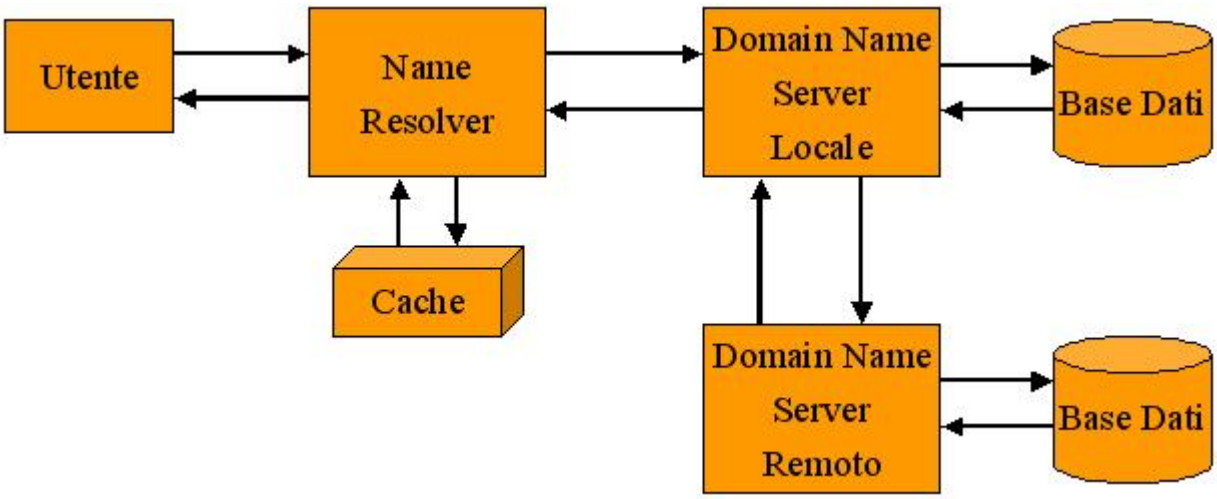
Utilizzazione di DNS

In una rete *IP* esiste normalmente almeno un *server* DNS, ossia un calcolatore dove è attivo il servizio DNS e nel quale risiede l'archivio con i nomi degli *host* di quella rete.

Negli *host* sono implementati metodi automatici per tradurre nomi in numeri, detti *Name Resolver*. Il *name Resolver* si preoccupa di questa traduzione, dialogando con il *Name Server* che è un programma specializzato per la risoluzione di nomi appartenenti ad altri domini.

Tutti i DNS servers utilizzano memoria *cache* per ridurre il traffico DNS sulla rete.

Di *default* DNS usa UDP: ma se la risposta supera 512 *bytes*, viene troncata e si esegue una ulteriore interrogazione tramite *TCP*, che supporta la frammentazione.



In questo modulo non si tratta dell'installazione del servizio DNS in quanto questa è già stata trattata nel **modulo 7**.

Directory services in Linux

Nei sistemi *Linux* un servizio di *directory* può essere implementato utilizzando il *Network*

Information Service o **NIS**.

Il **NIS** è un sistema, inizialmente sviluppato da *Sun* col nome di *Yellow Pages* (yp), che permette la gestione centralizzata delle informazioni amministrative in ambiente *UNIX* (*password*, gruppi, *hosts*, ...), che in questo modo possono essere condivise da tutti i *client* appartenenti ad uno stesso dominio **NIS**. Pertanto un utente può collegarsi su macchine diverse utilizzando il medesimo *account* senza dover tenere sincronizzate fra i diversi *computer* le tabelle delle *password*. Le tabelle **NIS** prendono il nome di mappe.

Le tabelle vengono rese disponibili da un *server master* e possono essere replicate su uno o più *server* con funzioni di *backup* (*slave*). La modifica dei dati contenuti nelle zone è possibile da macchine remote opportunamente abilitate.

NIS+ è una estensione di **NIS** che permette una miglior gestione di domini con un numero elevato di *client*. Con questa nuova implementazione è stata anche migliorata la sicurezza dell'intero sistema con una riprogettazione dei meccanismi di scambio dati.

Installazione del NIS

Se si vuole abilitare un *client* all'accesso ad un *server* **NIS**, è necessario attivare il demone `ypbind`, mediante uno *script* di avvio. Il dominio **NIS** viene impostato mediante il comando *domainname*, mentre la modalità di accesso al *server* viene definita nel *file* `/etc/yp.conf` nel seguente modo:

domain <dominio-**NIS**> *server* *<host>*

domain <dominio-**NIS**> *broadcast*

ypserv <*host*>

Nei *client* è necessario inoltre definire mediante il meccanismo di NSS (*Name Server Switch*), l'ordine con cui deve essere consultato il *server* **NIS** rispetto ai *file* di sistema. È buona norma consultare per primi i *file* di sistema, in modo da evitare che da un malfunzionamento di questo derivi l'impossibilità di collegarsi alla macchina. Nel caso della tabella delle *password* questo comportamento può essere ottenuto inserendo la seguente linea in `/etc/nsswitch.conf`:

passwd: *files* nis

Occorre inoltre aggiungere in coda a `/etc/passwd` la seguente linea:

+::::::

Di norma gli *account* amministrativi come *root*, bin, *wheel*, *demon* ed altri non vengono condivisi tramite **NIS**, per motivi di sicurezza e di *performance*, ma vengono invece utilizzati quelli definiti nel *file* `/etc/passwd` locale.

Per ulteriori informazioni su NSS e **NIS** si può fare riferimento anche alla unità didattica relativa all'autenticazione nei sistemi *UNIX*.

Configurazione del *server* **NIS**

Il *server* **NIS** per mantenere le informazioni non utilizza i classici *file* di testo, come ad esempio `/etc/passwd`, bensì dei *file* speciali detti mappe, contenenti coppie di valori nella forma chiave-valore, ordinati per chiave in modo da poter ricercare velocemente i dati.

Ad esempio partendo da `/etc/passwd` vengono generate le due mappe *passwd.byname* e *passwb.byuid*, che permettono rispettivamente ricerche veloci per nome utente e per numero identificativo (UID).

Le mappe vengono di norma salvate in `/var/nis` oppure in `/var/yp`, a seconda della configurazione che si utilizza. Per generare le mappe a partire dai *file* di sistema ogni implementazione mette a disposizione un comando apposito, ad esempio *makedbm*. Generalmente viene fornito un *Makefile* che permette la rigenerazione della mappe semplicemente spostandosi nella *directory* dove si trovano le tabelle e digitando *make*.

Per definire quali informazioni un *server* **NIS** può fornire alle macchine appartenenti ad un dominio si utilizza il *file* `/etc/ypserv.conf`, mentre gli *host* di fiducia possono essere impostati mediante `/etc/yp/securenets`.

Il demone che fornisce il servizio di *server* **NIS** è `ypserver`, che viene attivato in modalità *standalone* mediante uno *script* di avvio. Essendo **NIS** basato su RPC è necessario che sia attivo nel *server* il servizio *portmapper*.

Aggiornamento dei dati nelle mappe **NIS**

Per l'aggiornamento dei dati nel sistema **NIS** non è sufficiente utilizzare i classici comandi *UNIX*, ad esempio `passwd`. Vediamo ad esempio cosa succede se l'utente cambia la propria *password* su una macchina del dominio **NIS**. I casi possibili sono due:

- se la macchina non è il *server*, il *file* locale verrà aggiornato ma non verrà mai utilizzato, in quanto la macchina è configurata per prelevare le *password* via **NIS**.
- Se la macchina è il *server* **NIS**, questo dispone potenzialmente della nuova *password* ma le mappe contengono ancora la *password* precedente. Pertanto la nuova *password* funziona solo sul *server*, fintantoché non vengono rigenerate le mappe.

La soluzione consiste nell'utilizzo di comandi specifici per **NIS**, in questo caso `nispasswd`. Nel *server* è necessario attivare il servizio `rpc.passwdd`, che permette agli utenti di cambiare *password* da remoto. È anche possibile fare in modo che `rpc.passwdd` tenga automaticamente aggiornate le mappe a seguito di una modifica su uno dei *client*. In questo caso il servizio deve essere attivo quindi sia sul *server* che sui *client*.

Configurazione di un server slave

Per la configurazione di un *server* di tipo *slave* è sufficiente attivare normalmente le funzioni di **NIS** *server* (escluso `nispasswd`) e configurare il sistema come un *client* del *server master*. Per la replica

delle mappe si utilizza il comando

```
/usr/lib/yp/ypinit -s &lt;master-NIS&gt;
```

generalmente mediante uno degli *script* `/usr/lib/yp/ypxfr*`.

LDAP

LDAP (*Lightweight Directory Access Protocol*) è un altro sistema di gestione centralizzata di informazioni, il quale offre un approccio più generico rispetto a **NIS**, in quanto non si limita ai soli dati amministrativi ma è utilizzabile anche per gestire dati generici quali *bookmarks* o indirizzari.

L'aggettivo *lighweight* (leggero), suggerisce una progettazione del sistema che privilegia le prestazioni. L'architettura di **LDAP** è infatti studiata per avere tempi di risposta molto veloci nella ricerca e lettura dei dati, a scapito di una maggiore lentezza nelle operazioni di scrittura e aggiornamento. Questo non rappresenta un problema grave, in quanto le operazioni di scrittura avvengono di norma meno frequentemente rispetto a quelle di lettura (si pensi al numero di volte in cui la propria agenda telefonica viene consultata rispetto a quelle in cui viene modificata).

LDAP gestisce insiemi di dati strutturati secondo una gerarchia ad albero (*directory*), nei quali i singoli elementi possono essere oggetti che possiedono diversi attributi. Gli elementi di una *directory*, nodi intermedi o foglie dell'albero, possono anche essere riferimenti ad altre *directory* residenti su altri *server* **LDAP**. Ogni elemento deve avere un nome che permetta di individuarlo in modo univoco nella *directory*.

Alcuni dei tipi possibili utilizzabili per definire gli attributi di un oggetto sono i seguenti:

- bin: dato binario;
- ces (*case exact string*): stringa in cui vengono distinte le lettere maiuscole dalle minuscole;
- cis (*case ignore string*): stringa in cui le lettere maiuscole e minuscole sono trattate allo stesso modo;
- tel: numero telefonico;
- dn (*distinguished name*): nome univoco;

Un *server* che fornisce in *UNIX/Linux* il servizio **LDAP** è `slapd`. Esso mantiene i dati in un formato denominato LDBM, simile al DBM ma ottimizzato per le ricerche (per ognuno degli attributi degli oggetti contenuti in una *directory* viene creato un indice).

Directory services in Windows

Il sistema operativo *Windows* si rifà a X.500 per il servizio di *directory*. In *Windows* 2000 viene implementato un servizio di *directory* detto *Active Directory*, già citato nel **modulo 7**, che mette a disposizione una *directory* di oggetti specifici al NOS (*Network Operating System*) che consente di gestire non soltanto gli utenti e le loro proprietà, ma anche vari tipi di funzionalità specifiche come i volumi DFS (*Distributed File System*), gli oggetti GPO (*Group Policy Object*) e l'infrastruttura PKI (*Public Key Infrastructure*). Il tipi di oggetti che possono essere contenuti in una *directory* sono virtualmente illimitati. In ogni caso, occorre tenere conto delle implicazioni relative a prestazioni e replicazione.

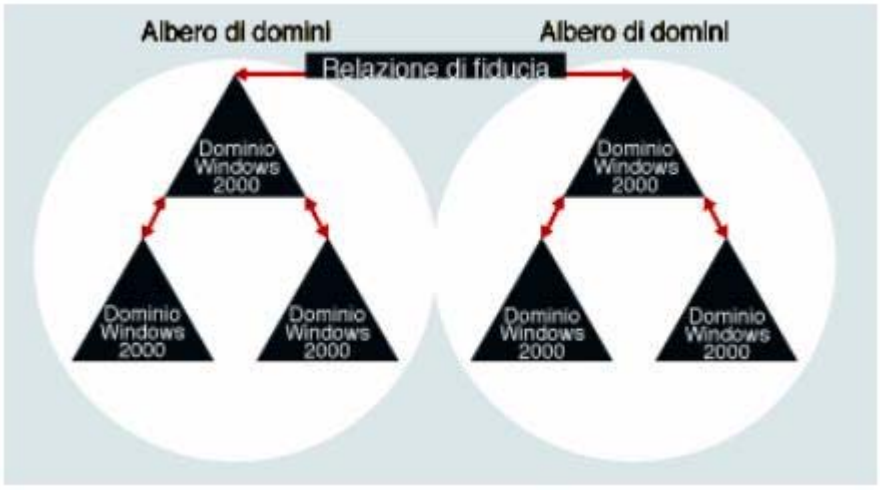
Sviluppando *Active Directory*, *Microsoft* ha dovuto prevedere la compatibilità a ritroso con gli ambienti NT 4.0. In conseguenza, molti concetti usati da *Active Directory* appaiono familiari a molti utenti abituati all'ambiente NT.

Domini. In *Windows* 2000 le politiche di sicurezza sono ancora legate al dominio, così come esistono ancora i gruppi *Domain Admins*. I domini di *Active Directory* non usano più lo *standard* di nomi **NetBIOS** a quindici caratteri; anche se questo tipo di nomi risulta ancora presente per offrire la compatibilità all'indietro, i dispositivi *Windows* 2000 riconoscono i domini *Active Directory* attraverso i loro nomi DNS (*Domain Name System*). In *Windows* 2000, infatti, il servizio di *default* per la gestione dei nomi è il DNS; tutti i domini *Active Directory* hanno un dominio DNS che serve a identificarli (per esempio *mycompany*.com).

In NT 4.0 possono esistere soltanto due livelli di gerarchia tra i domini, dal momento che le relazioni di fiducia non sono di tipo transitivo. Per esempio, si supponga di avere tre domini NT 4.0 - A, B e C - e di volere che il dominio B e quello C accordino la fiducia al dominio A, e che il dominio C accordi la fiducia a quello B. Si può creare una relazione di fiducia tra il dominio B e quello A e tra i domini C e A, ma è necessario creare una relazione di fiducia esplicita dal dominio C a quello B. *Windows* 2000 elimina questa limitazione. L'uso di *Kerberos* 5 quale protocollo di autenticazione di *default* implica che le relazioni di fiducia possano essere bidirezionali e transitive. In conseguenza, ci possono essere molti livelli di gerarchie dei domini. Per esempio, il dominio A può accordare la fiducia a quello B, il quale la accorda al dominio C, e così via.

Alberi e foreste di domini

Un albero è costituito da un insieme di domini che si accordano vicendevolmente la fiducia e che appartengono a uno spazio di nomi contiguo (per esempio, un albero di *directory* in cui ciascun dominio è un sotto-dominio del proprio dominio padre). Un esempio di spazio di nomi contiguo può essere un albero di domini che contiene alla radice il dominio *mycompany*.com, un dominio figlio chiamato *east.mycompany*.com sotto quest'ultimo, e un dominio figlio chiamato *finance.east.mycompany*.com sotto *east.mycompany*.com. In questo esempio, i tre domini formano uno spazio di nomi contiguo e, in conseguenza, un albero di domini.



Una foresta è invece costituita da un albero di domini (o un insieme di alberi di domini) caratterizzato da spazi di nomi contigui separati. Quando nell'ambiente viene installato il primo *controller* sul primo dominio del primo albero, è necessario specificare se appartiene a una nuova foresta oppure a una foresta già esistente. In *Active Directory*, tutti i domini di una foresta devono condividere il medesimo schema. *Windows* 2000 non permette di fondere più foreste o schemi; in conseguenza, se è necessario creare più foreste (per esempio, quando la società si fonde con un'altra che dispone già di una foresta *Active Directory*), occorre usare relazioni di fiducia non transitive e di tipo *downlevel* per collegare le foreste. In alternativa, si possono usare dei *tool* per spostare gli oggetti da una foresta all'altra. Se non si dispone di *tool* per gestire più foreste nell'ambito delle grandi aziende, è buona norma utilizzare il minor numero possibile di foreste.

DIT in active directory

In NT 4.0, il *database* SAM (*Security Accounts Manager*) contiene tutte le informazioni relative a utenti, *computer* e gruppi del dominio. Le sue dimensioni hanno limiti di scalabilità dovuti alla sua implementazione. Sui *controller* di dominio *Windows* 2000, il *database* SAM viene sostituito dall'albero DIT. Quest'ultimo si basa sul motore *database Jet* di *Microsoft* ed è simile al motore *Jet* usato da *Microsoft Exchange Server*.

Il *file* ntds.dit, contenuto nella *directory* `\%systemroot%\ntds`, è l'equivalente *Windows* 2000 del *file* SAM. Questo *file* contiene il *database* della *directory*. In generale, l'albero DIT è più esteso del *database* SAM dal momento che *Active Directory* contiene più informazioni e tipi di oggetti rispetto alla *directory* di NT 4.0. All'interno di un dominio, i contenuti del *file* ntds.dit vengono replicati su tutti i *controller*. Si potrebbe ritenere che la migrazione da NT 4.0 a *Windows* 2000 comporti un maggiore traffico di replicazione tra i *controller* di dominio; *Windows* 2000 usa tuttavia un modello completamente diverso da quello di NT 4.0 per replicare le modifiche alla *directory*.

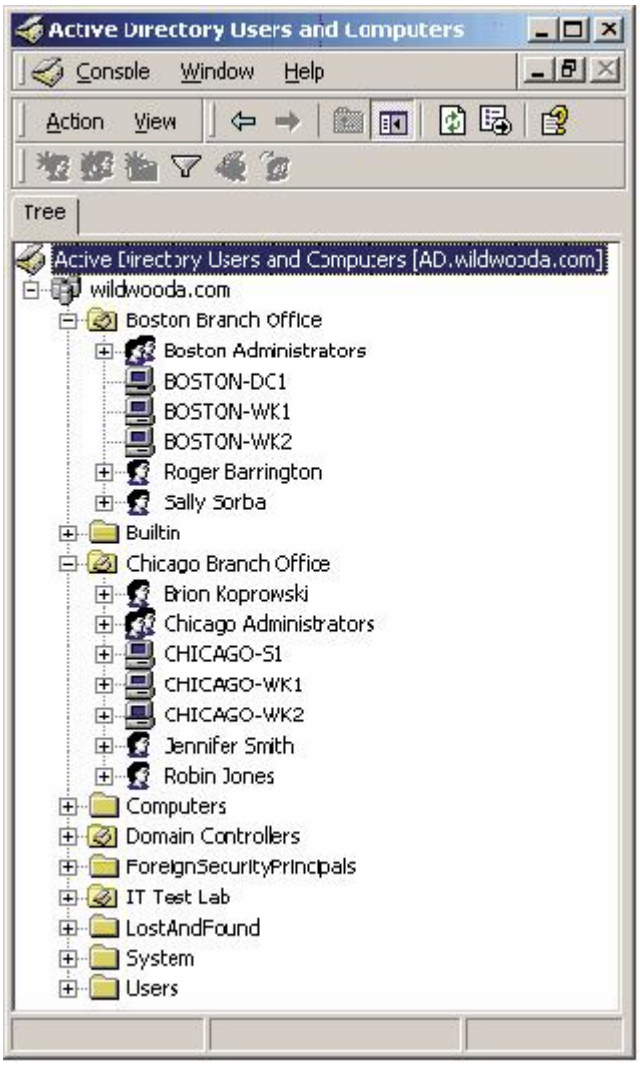
Active Directory fornisce a *Windows* 2000 molte funzionalità che diversificano questo sistema operativo da NT 4.0 e lo rendono più facile da usare nella gestione delle grandi aziende. Queste nuove funzionalità comprendono:

- il *Global Catalog*;
- le unità OU;
- i gruppi espansi;
- la replicazione della *directory*;
- una nuova struttura dei *controller* di dominio.

Global Catalog. Il *Global Catalog* è un nuovo concetto, introdotto per la prima volta con *Windows* 2000. Il catalogo è un indice separato di oggetti in una foresta *Active Directory*. Per *default*, questo indice non contiene tutti gli oggetti presenti nel *database* di *Active Directory*, ma soltanto una parte dei loro attributi. Il *Global Catalog* permette agli utenti di individuare rapidamente gli oggetti della *directory* all'interno della foresta aziendale, senza doversi recare presso il *controller* del dominio in cui risiede l'oggetto. Il *Global Catalog* viene usato al meglio quando sono presenti più domini e alberi, sparsi su varie reti. È necessario avere a disposizione sulla rete almeno un *server Global Catalog* affinché i *client* possano compiere l'autenticazione sui domini *Active Directory*.

Per *default*, il primo *controller* del primo dominio nel primo albero diventa il *server Global Catalog*. Per specificare manualmente altri *controller* di dominio come *server Global Catalog*, si può usare lo *snap-in* MMC (*Microsoft Management Console*) *Active Directory Sites and Services*.

Sebbene la maggior parte delle informazioni del dominio (come utenti e gruppi) venga replicata soltanto sul *controller* all'interno del dominio stesso, *Active Directory* replica il *Global Catalog* attraverso i confini del dominio e verso tutti i *controller* che sono dei *server Global Catalog*. Nel momento in cui viene implementato *Windows* 2000, è opportuno posizionare attentamente i *server Global Catalog*. Ciascuna macchina *client* deve avere un facile accesso al *Global Catalog*, in modo da ottimizzare la capacità del *computer* di compiere una ricerca all'interno della *directory*. Il *Global Catalog* sostituisce inoltre la lista GAL (*Global Address List*) in *Exchange* 2000 *Server* (in precedenza chiamato in codice Platinum).



Unità OU. Le unità OU permettono di delegare il controllo delle risorse di dominio *Windows* 2000. Quando si crea una unità OU nel dominio *Active Directory*, viene creato un confine amministrativo all'interno del quale è possibile delegare a un sottoinsieme di utenti la gestione degli oggetti contenuti in questa OU. Come è stato accennato in precedenza, ciascuna unità OU può contenere altre unità, oppure oggetti *leaf* come utenti, *computer*, o stampanti. È possibile nidificare qualsiasi numero di unità OU all'interno di altre; affinché la cosa risulti pratica, è tuttavia opportuno limitare a un massimo di dieci il numero massimo di OU nidificate nel dominio. Per creare le OU nidificate, si può usare lo *snap-in* MMC *Active Directory Users and Computers*.

Ad esempio si consideri l'unità OU chiamata US contiene quella California, che a sua volta contiene quella *Finance*. In quest'ultima è contenuto un oggetto utente chiamato *Joe User*. Si supponga che quest'ultimo sia l'amministratore locale del dipartimento *Finance* in California. Per delegare a *Joe User* il controllo dell'unità OU *Finance* su tutti gli oggetti posti al suo interno, si può usare la procedura di autocomposizione *Delegation of Control Wizard* dello *snap-in* MMC *Active Directory Users and Computers*. Per avviare questa procedura è sufficiente fare click con il pulsante destro del *mouse* in corrispondenza dell'unità OU, quindi selezionare *Delegate Control*. Successivamente occorre scegliere l'utente o il gruppo a cui delegare il controllo e specificare quali diritti dovranno avere sugli oggetti contenuti nell'unità OU. In alternativa, è possibile selezionare *Custom task* per assegnare i diritti usando un completo elenco. I diritti che si possono assegnare corrispondono alle liste **ACL** (*Access Control List*) di sicurezza sugli oggetti OU cui è stato delegato il controllo. Anche se è possibile modificare manualmente le liste **ACL** su una unità OU, utente, oppure gruppo, in modo da assegnare i diritti di sicurezza per i singoli oggetti, la procedura di autocomposizione *Delegation of Control Wizard* offre una semplice interfaccia **GUI** per delegare il controllo.

Gruppi *Windows* 2000. NT dispone di soltanto due tipi di gruppi: globale e locale. Questi gruppi esistono unicamente a fini di sicurezza (per esempio, per assegnare la sicurezza alle risorse) e possono contenere soltanto oggetti utente. *Windows* 2000 dispone invece dei gruppi globale e locale di dominio, oltre a un nuovo gruppo di sicurezza chiamato gruppo universale. I gruppi universali diventano disponibili quando i domini *Active Directory* vengono fatti passare dalla modalità mista a quella nativa. In modalità mista, un dominio *Windows* 2000 può contenere dei *controller* di dominio *Windows* 2000 e dei *controller* BDC (*Backup Domain Controller*) di NT 4.0. Nella modalità nativa, i domini non possono invece contenere i *controller* BCD di NT 4.0. Il passaggio alla modalità nativa è una funzione a senso unico: non sarà più possibile ritornare nelle condizioni precedenti.

I gruppi universali possono contenere quelli globali e altri gruppi universali provenienti da qualsiasi dominio della foresta. I gruppi globali sono invece specifici al dominio (un gruppo globale contiene utenti, *computer*, oppure altri gruppi globali provenienti unicamente dall'interno del medesimo dominio). Ovviamente, i gruppi globali di un dominio possono essere membri di gruppi locali di un altro dominio. I gruppi universali permettono inoltre di nidificare nella foresta i gruppi globali e universali di altri domini. In *Windows* 2000 è possibile creare dei gruppi di sicurezza che contengono oggetti macchina. In conseguenza, si possono impostare i permessi di accesso alle risorse usando gruppi basati sulle macchine e non soltanto sugli utenti.

Windows 2000 permette di creare dei gruppi non di sicurezza chiamati gruppi di distribuzione, che hanno un ambito analogo (ovvero locale, globale e universale) a quelli di sicurezza. Questi gruppi funzionano come le liste DL (*Distribution List*): non hanno un contesto di sicurezza ma permettono di raggruppare gli utenti per scopi come la posta elettronica.

Replicazione della *directory*. *Windows* 2000 usa un nuovo modello di replicazione per fare in modo che tutti i *controller* di dominio della foresta dispongano di informazioni aggiornate. Questo modello si basa sul concetto di replicazione *multimaster*. In NT 4.0 soltanto il *controller* PDC (*Primary Domain Controller*) mantiene una copia a lettura/scrittura del *database* SAM. In *Windows* 2000, invece, ciascun *controller* del dominio contiene una copia a lettura/scrittura dell'albero DIT. Gli utenti possono apportare a qualsiasi *controller* di dominio delle modifiche che vengono replicate sugli altri *controller*.

Update sequence number

Windows 2000 fa uso di una funzione chiamata numero USN (*Update Sequence Number*) per determinare se è necessario replicare le modifiche da un *controller* di dominio all'altro. Ciascun oggetto e le sue proprietà in *Active Directory* contengono un numero USN, che viene usato dai *controller* di dominio per stabilire quando devono avvenire le modifiche su un *partner* di replicazione. Durante un ciclo di replicazione, le modifiche (e non l'intero oggetto) vengono replicate per ogni proprietà. Per esempio, se il numero di telefono di un oggetto utente cambia sul *controller* di dominio 1, viene replicato sul *controller* di dominio 2 soltanto il nuovo numero (non l'intero oggetto utente). Se la modifica a una proprietà si verifica su due *controller* di dominio, il contrassegno di data e ora aiuta a fare in modo che venga presa in considerazione soltanto la modifica più recente. Per replicare le informazioni *Active Directory* e di dominio, i *controller* di una foresta usano tre contesti di assegnazione dei nomi di replicazione. Si può pensare ai contesti di assegnazione dei nomi come ai percorsi seguiti dalle informazioni replicate. Ciascun contesto di assegnazione dei nomi può seguire un percorso differente tra i *controller* di dominio nella foresta, inoltre replica informazioni diverse a seconda del loro ruolo. Il contesto di assegnazione dei nomi di dominio replica le modifiche DIT sui *controller* poste in all'interno del dominio; il contesto di assegnazione dei nomi dello schema replica le informazioni di schema su tutti i *controller* di dominio all'interno di una foresta; il contesto di assegnazione dei nomi della configurazione replica le informazioni di configurazione (come la tipologia di replicazione) su tutti i *controller* di dominio della foresta.

Active Directory usa i siti per consentire di controllare il traffico di replicazione tra ubicazioni caratterizzate da collegamenti *WAN* lenti. I siti *Active Directory*, proprio come i siti *Exchange Server*, sono aree caratterizzate da un'elevata larghezza di banda di rete. All'interno di un sito, il processo KCC (*Knowledge Consistency Check*) che si trova in esecuzione su ciascun *controller* di dominio genera automaticamente la tipologia di replicazione del *controller* per ogni contesto di assegnazione dei nomi. Il processo KCC crea una tipologia ad anello tra i *controller* di dominio del sito. Con la crescita del numero dei *controller*, il processo KCC aggiunge tra di essi nuovi oggetti di connessione in modo da impedire un numero eccessivo di salti tra due *controller* di dominio qualsiasi. È possibile pianificare manualmente la frequenza di replicazione tra i siti, a seconda di quali siano le proprie esigenze di rete. Per definire i siti manualmente è necessario usare lo *snap-in* MMC *Active Directory Sites and Services*.

Occorre inoltre creare degli oggetti *subnet* che corrispondano a tutte le *subnet TCP/IP* presenti sulla rete, quindi associare queste *subnet* ai siti appropriati. Le *workstation* usano queste informazioni per individuare il *controller* di dominio più vicino ai fini di autenticazione, dal momento che preferiscono usare un *controller* all'interno del sito prima di iniziare a interrogare a caso il servizio DNS alla ricerca degli altri *controller* di dominio disponibili.

Backup e Restore

Le procedure di *backup* hanno lo scopo di mantenere disponibile copie aggiornate dei dati sia di utente sia relative ai sistemi operativi al fine di poter recuperare eventuali malfunzionamenti che portino alla perdita dei dati presenti nella loro naturale locazione. Il *backup* può essere effettuato direttamente sul calcolatore locale utilizzando ad esempio un *Hard Disk* aggiuntivo o il CD, oppure tramite rete copiando i *file* su di un dispositivo connesso ad un altro calcolatore, normalmente un *server* di rete.

In genere è bene effettuare il *backup* su di uno o più dispositivi distinti da quello su cui risiedono normalmente i dati stessi al fine di evitare che un guasto del dispositivo che lo renda inservibile provochi la perdita dei dati originali così come delle copie.

A questo scopo si utilizzano *Hard Disk* dedicati su *server* di rete, a loro volta protetti per esempio con sistemi RAID di cui si è parlato nel modulo 7, oppure dispositivi accessori quali dischi estraibili, cassette, dischi ottici, eccetera.

Per essere efficace un *backup* deve essere pianificabile a livello di amministrazione di sistema e in modo indipendente dalle azioni dell'utente. Infatti il singolo utente, occupato in altre attività può facilmente dimenticare di svolgere le azioni di *backup* ed esporsi quindi ad una potenziale perdita di dati.

È opportuno effettuare operazioni di *backup* in momenti in cui il sistema sia poco o per nulla utilizzato, ad esempio alla sera o nei fine settimana. Due sono le ragioni principali per questo:

- se gli utenti stanno modificando i dati quando viene effettuato un *backup*, quest'ultimo potremmo non salvare l'ultima versione dei documenti e quindi venire meno al suo scopo;
- per sua natura il *backup* comporta la copia di grandi moli di dati e quindi intensive operazioni di lettura e scrittura dai dischi e di trasferimento di dati sulla rete e quindi interferisce con il normale funzionamento del sistema.

Se effettuato quando i calcolatori non sono utilizzati il *backup* risulta più efficace e si minimizza la sua influenza sul normale funzionamento dell'ambiente di rete.

Inoltre per diminuire i tempi di *backup* si può ricorrere a strategie di compressione e/o di *backup* incrementale o differenziale:

- Un *backup* differenziale effettua il *backup* di ciò che è stato modificato rispetto ad un *backup* completo di riferimento.
- Un *backup* incrementale effettua il *backup* di ciò che è stato modificato rispetto al *backup* precedente, eventualmente anch'esso incrementale.

I *backup* differenziali ed incrementali copiano solamente una parte dei dati e risultano quindi più efficienti per occupazione di spazio di memoria e per velocità. D'altro canto la procedura di recupero dei dati risulta generalmente più complessa.

Strategie di backup

Una strategia di *backup* tipicamente combina diverse tipologie di *backup*: alcune di queste strategie privilegiano i tempi di *backup*, mentre altre privilegiano i tempi di ripristino.

Di seguito sono illustrati alcuni esempi:

- Esempio 1: Normale e Differenziale. Viene effettuato un *backup* Normale il Lunedi ed un *backup* Differenziale dal Martedi al Venerdi. In caso di fallimento bisogna ripristinare il *backup* Normale del Lunedi e l'ultimo Differenziale disponibile. Tale strategia favorisce i tempi di ripristino rispetto a quelli di *backup*.
- Esempio 2: Normale e Incrementale. Viene effettuato un *backup* Normale il Lunedi ed un *backup* Incrementale dal Martedi al Venerdi. In caso di fallimento bisogna ripristinare il *backup* Normale del Lunedi e tutti gli Incrementali disponibili, nell'ordine in cui sono stati effettuati. Tale strategia favorisce i tempi di *backup* rispetto a quelli di ripristino.
- Esempio 3: Normale, Differenziale e *Copy*. Questa strategia è la stessa dell'Esempio 1, con la differenza che al Mercoledi viene effettuato un *backup* di tipo *Copy* per catturare una immagine completa dei dati pur senza influire sulla strategia di *backup* in corso. Ha il vantaggio di avere un'immagine di metà periodo, qualora qualcosa dovesse corrompere il *backup* originale.

Backup in Unix, Dump e Restore

Il sistema operativo *Unix* permette di svolgere funzioni di *backup* utilizzando vari comandi.

Il comando *dump* viene utilizzato per il *backup* di *file* e *directory*. *Dump* esamina I *file* in un *filesystem*, determina automaticamente per quali è necessario il *backup* (ossia quelli che non sono già stati salvati in precedenza), e copia quei *file* su un mezzo di memorizzazione specificato che può esser un altro disco rigido, una cassetta, eccetera.

Il comando *restore* svolge la funzione inversa rispetto a *dump*, per cui legge un archivio di *file* creato da *dump* e ne estrae i *file* che non sono presenti sul *filesystem* considerato.

L'opzione fondamentale del commando *dump* è il cosiddetto livello di *dump*, specificato con un numero intero da 0 a 9. Il livello 0 assicura che tutto il *filesystem* specificato sia copiato per *backup*. Livelli superiori a 0 servono per impostare *backup* incrementali o differenziali.

Una possibile strategia di *backup* potrebbe essere quella indicata nella tabella seguente per un periodo mensile, in cui mensilmente viene effettuato un *backup* completo (livello 0), settimanalmente viene effettuato un *backup* incrementale rispetto alla settimana precedente (livello 3) e giornalmente viene effettuato un *backup* incrementale rispetto al giorno precedente (livello 9).

In questo modo giornalmente si salvano solamente i dati modificati nel giorno, settimanalmente i dati modificati nell'intera settimana e mensilmente l'intero parco dati.

Settimana	Domenica	Lunedì	Martedì	Mercoledì	Giovedì	Venerdì	Sabato
1	0	9	9	9	9	9	3
2	-	9	9	9	9	9	3
3	-	9	9	9	9	9	3
4	-	9	9	9	9	9	3
5	0	9	9	9	9	9	3

Ad esempio il *backup* del *filesystem* /*home* su di un dispositivo a cassetta (`/dev/sd0`) può essere effettuato con il comando:

```
/sbin/dump -0 -f /dev/sd0 /home
```

Mentre il rispristino viene effettuato con il comando:

```
/sbin/restore -f /dev/sd0
```

Entrambi i comandi sono dotati di una quantità di opzioni in dettaglio reperibili sulle pagine *man* oppure sulla documentazione del sistema.

Backup con TAR

In alternativa a *dump* è possibile effettuare *backup* utilizzando il comando `tar`. `Tar` permette di raggruppare un intero *filesystem* od una sua porzione (ossia un sottoinsieme di *file* e di cartelle) in un unico *file* archivio detto *tar file*, dal quale possono poi essere estratti in seguito.

Per esempio il comando:

```
/bin/tar cf ./backupRossi.tar /home/Rossi/Documenti
```

in cui l'opzione `c` comanda la creazione di un nuovo *file* il cui nome è specificato dall'opzione `f` seguita dal nome del *file* stesso (`backupRossi.tar` in questo caso), ha come risultato la creazione nella cartella corrente di un *file* di nome `backupRossi.tar` che contiene il *backup* dell'intera cartella dei documenti di Rossi, contenuta nella sua *home directory* al percorso specificato.

Qualora i dati della cartella Documenti di Rossi dovessero andare perduti interamente o parzialmente per una qualsivoglia ragione sarebbe possibile recuperarli a partire dal *file* `backupRossi.tar` utilizzando nuovamente il comando `tar` come segue:

```
/bin/tar xf ./backupRossi.tar /
```

dove l'opzione `x` comanda l'estrazione dei dati dal *file* di *backup*, specificato con l'opzione `f` seguita dal nome.

Utilizzando altre opzioni è possibile aggiungere *file* ad un *backup*, semplicemente visualizzare il suo contenuto, eccetera. Queste opzioni sono reperibili sulla pagina di manuale del comando `tar`.

Pianificazione dei backup con CRON

In *Unix*, con il demone (servizio) *cron* permette di eseguire comandi in modo automatico seguendo una precedente pianificazione. È possibile stabilire l'ora e il giorno in cui un particolare programma

va eseguito, anche in modo ricorrente (ad esempio tutti i giorni alle 21.00, oppure tutti i sabati sera alle 23.00).

Cron è quindi uno strumento molto adatto ad essere utilizzato per predisporre dei *backup* periodici di *filesystem* senza la necessità che gli utenti e/o l'amministratore di sistema siano presenti.

Normalmente il comando `crontab` viene utilizzato dal *superuser*. I *file* `/etc/cron.allow` ed `/etc/cron.deny` servono per specificare esplicitamente gli utenti ai quali è permesso o negato l'uso di *cron*.

Le informazioni relative al comando da eseguire ed a quando eseguirlo sono contenute in una apposita tabella (`/etc/crontab`), modificabile utilizzando l'*utility* `/usr/bin/crontab`, oppure nella *directory* `/etc/cron.d`. Per ciascun comando è possibile specificare mese, giorno, ora e minuto di esecuzione.

Backup in Windows

Windows 2000 comprende un *tool* di *backup* denominato *Backup* e situato in Start\Programs\Accessories\System Tools. Utilizzando tale strumento è possibile:

- Effettuare il *backup* di *files* e cartelle.
- Effettuare il *backup* del sistema.
- Schedulare delle attività di *backup*.
- Ripristinare *files* e cartelle.

Utilizzando tale strumento è anche possibile effettuare un *Emergency Repair Disk* (ERD).

È possibile effettuare cinque tipi di *backup*. La principale differenza tra essi riguarda la presenza o meno di marcatori (*marker*), conosciuti anche come attributi di archivio. La loro assenza segnala un *backup* avvenuto.

Si hanno dunque le seguenti tipologie di *backup*:

- *Normal*. Effettua il *backup* di tutti i *files* e le cartelle selezionate. Elimina i marcatori per segnalare l'avvenuto *backup*.
- *Copy*. Effettua il *backup* di tutti i *files* e le cartelle selezionate. Lascia i marcatori. *Server* per effettuare un *backup* completo senza influire su strategie di *backup* un corso d'essere.
- *Differential*. Effettua il *backup* di *files* e cartelle che hanno il marcatore e lascia i marcatori intatti.
- *Incremental*. Effettua il *backup* di *files* e cartelle che hanno il marcatore ed elimina tali marcatori.
- *Daily*. Effettua il *backup* di *files* e cartelle che hanno subito modifiche durante la giornata.

Backup di file e cartelle

Utilizzando lo strumento *Backup* situato in *Start\Programs\Accessories\System Tools* è possibile effettuare il *backup* di *files* e cartelle su volumi formattati **FAT**, **FAT32** o NTFS.

Tale strumento mette a disposizione un *wizard* che ci aiuta nella definizione del processo di *backup* e di ripristino, anche se è comunque possibile realizzare tali attività manualmente.

Quando si effettua un *backup* bisogna specificare, oltre al tipo di *backup*, le seguenti informazioni:

- I volumi, *files* e cartelle di cui effettuare il *backup*.
- La destinazione del *backup*.
- Opzioni di *backup* come, ad esempio, l'eventuale creazione ed il *path* di un *file* registro, una descrizione, eccetera.
- Se sovrascrivere i *backup* esistenti o aggiungere il nuovo *backup* a questi.
- Opzioni avanzate come, ad esempio, la verifica del *backup* e l'attivazione della compressione *hardware*.

Per poter effettuare il *backup* ed il *restore* di dati su un *computer Windows* 2000, bisogna avere appropriati privilegi e diritti, come di seguito indicato:

- Tutti gli utenti possono effettuare il *backup* dei *files* e delle proprie cartelle propri o per i quali hanno il permesso di *Read*.
- Tutti gli utenti possono effettuare il ripristino dei *files* e delle cartelle per i quali hanno il permesso di *Write*.
- Gli appartenenti ai gruppi *Administrators*, *Backup Operators* e *Server Operators* possono effettuare il *backup* ed il ripristino di tutti i *files* e tutte le cartelle indipendentemente dai permessi.

Backup dello stato del sistema

Utilizzando lo strumento *Backup* situato in *Start\Programs\Accessories\System Tools* è possibile effettuare il *backup* dello Stato del Sistema: se il sistema si corrompe, tramite tale *backup* ed il CD di installazione di *Windows* 2000 è possibile ripristinare lo stato del sistema come al momento del *backup*.

Il *backup* dello Stato del Sistema comprende le seguenti componenti:

- Registri.
- *Component Services class registration database*.
- *Files* di avvio del sistema.
- *Database* dei *Certificate Services*, se tali servizi sono installati e configurati (non per macchine *Windows* 2000 *Professional*).
- *Active Directory* (solo per Controllori di Dominio).
- Cartella Sysvol (solo per Controllori di Dominio).

Non è possibile selezionare singolarmente tali componenti.

Pianificazione temporale dei backup

Si definisce *Job* di *backup* un singolo processo di *backup* dei dati.

È estremamente importante pianificare ed eseguire attività di *backup* con una certa regolarità, allo scopo di poter ripristinare più dati possibili in caso di perdita degli stessi.

Per facilitare l'esecuzione regolare di *Job* di *backup*, lo strumento *Backup* è integrato con *Task Scheduler*, per cui è possibile schedulare *Job* di *backup* in maniera tale che essi vengano eseguiti regolarmente ed in periodi ben determinati, magari coincidenti con momenti di scarso carico di lavoro per il sistema e/o per la rete.

Per scheulare un *Job* di *backup* utilizzare la scheda *Scheduled Jobs* dello strumento *Backup* situato in *Start\Programs\Accessories\System Tools*, fare doppio click sul giorno in cui si vuole iniziare l'esecuzione del *Job* e completare il *wizard* specificando:

- Di cosa si vuole effettuare il *backup* (Tutti i *file*, Solo *files* e cartelle selezionate, Lo stato del sistema).
- Tipo di supporto utilizzato e nome del *file* di *backup*.
- Tipo di *backup*.
- Verifica del *backup*.
- Se aggiungere tale *Job* a *job* precedenti eventualmente contenuti sul supporto o sovrascriverli.
- Etichetta associata al *backup*.
- *Account* utilizzato dal *Job* (con opportuni privilegi).
- Nome del *Job*.
- Schedulazione.

Ripristino di file e cartelle

Utilizzando lo strumento *Backup* situato in *Start\Programs\Accessories\System Tools* è possibile effettuare il ripristino di *files* e cartelle in caso di perdita dei dati.

Tale strumento mette a disposizione un *wizard* che ci aiuta nella definizione del processo di ripristino, anche se è comunque possibile realizzare tali attività manualmente.

Quando si effettua un ripristino bisogna specificare le seguenti informazioni:

- I *files* e le cartelle da ripristinare.
- La destinazione del ripristino. Di *default* sarà corrispondente a quella da cui è stata effettuato il *backup*, ma può essere modificata.
- Opzioni di ripristino come, ad esempio, se sovrascrivere i *file* esistenti.

Bisogna ripristinare su NTFS *files* e cartelle che risiedevano su una partizione NTFS al momento del *backup*, allo scopo di non perdere tutti i settaggi inerenti i permessi, *Encrypting File System*, la definizione di *quote* ed altro.

Gestione di rete

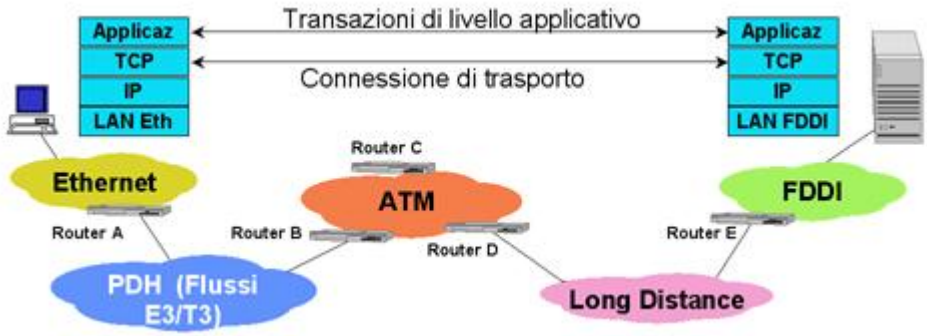
In enti e aziende di ogni tipo e dimensione è sempre più essenziale la disponibilità di un efficiente Sistema Informativo, tipicamente di tipo distribuito e quindi utilizzante tecnologie di rete.

Quando le dimensioni del sistema sono rilevanti (centinaia o migliaia di utenti), ed a maggior ragione se è anche distribuito territorialmente, le infrastrutture di rete utilizzate possono diventare realmente molto complesse. Tali infrastrutture sono prevalentemente costituite da un numero sempre crescente di LAN (*Local Area Network*), interconnesse tra loro localmente o per mezzo di collegamenti geografici. I responsabili della gestione di tali reti si trovano pertanto a dover gestire sistemi sempre più complessi con risorse umane tipicamente piuttosto limitate. Una delle strade percorribili per ottenere un incremento del grado di efficienza nell'uso di queste risorse consiste nell'utilizzo di tecnologie che consentano il controllo centralizzato delle strutture di rete e dei servizi realizzati per il loro tramite.

Gli obiettivi da raggiungere, legati alla necessità di gestire le applicazioni che stanno alla base delle attività aziendali, sono essenzialmente i seguenti:

- massimizzare il livello di disponibilità della rete;
- contenere al minimo i costi di gestione;
- ottimizzare le prestazioni e la qualità del servizio fornito;
- identificare per tempo le nuove esigenze al fine di pianificare l'evoluzione della rete.

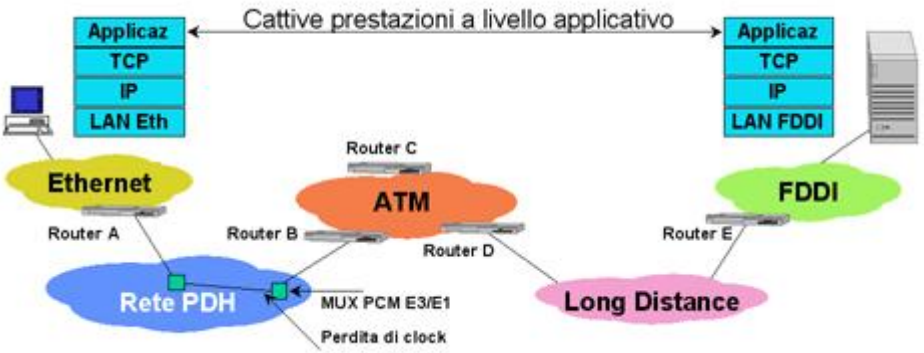
Uno degli aspetti più critici nella gestione di una rete complessa è la sua configurazione. I problemi da superare sono legati alla necessità di confrontarsi con reti sempre più eterogenee in cui vengono impiegati tecnologie ed apparati sempre più complessi (si pensi, ad esempio, all'impiego ormai diffuso di tecniche che virtualizzano le reti: LAN virtuali, Emulazione di LAN su ATM, reti private virtuali in ambito geografico, eccetera).



Il problema si complica ulteriormente quando, come ad esempio nel caso in figura, più gruppi di persone gestiscono domini diversi, cioè operano separatamente su porzioni diverse della rete: la LAN *Ethernet* di origine, il circuito diretto numerico (CDN) in ambito locale, il *backbone* nazionale ATM ad alta velocità, la rete internazionale, la LAN FDDI di destinazione. In tali condizioni è spesso difficile mantenere la consistenza nella configurazione degli apparati che andrebbero invece configurati in modo congruente. Tale problema è particolarmente sentito ogni volta che si rendono necessarie modifiche di configurazione, durante le quali è particolarmente difficile mantenere la consistenza tra le configurazioni dei diversi apparati. Ad oggi mancano, purtroppo, tecnologie per automatizzare la configurazione della rete e l'obiettivo di avere reti autoconfigurabili sembra ancora abbastanza distante.

Individuazione dei guasti

Un'altra grande criticità nella gestione di rete è legata all'individuazione dei guasti. Per evidenziare tale difficoltà si faccia riferimento, ad esempio, ad un guasto che causi una perdita di sincronizzazione intermittente sul *clock* di un flusso E3.



Tale guasto si propaga: la perdita di sincronismo a livello di flusso E3 (34 Mbps) causa la perdita di trame E3; si perdono pertanto trame E1 (2 Mbps) e sono quindi corrotti i flussi a 64 kbps su di esse trasportati. Saranno quindi affetti da un elevato tasso di errore un gran numero di connessioni TCP. Poichè il TCP utilizza un meccanismo di adattamento della dimensione della finestra di trasmissione, il risultato è una notevole riduzione del *throughput* su tutte le connessioni TCP trasportate. Aumentano quindi i ritardi da estremo ad estremo sulle applicazioni e, se gli applicativi sono sensibili a tali ritardi, un gran numero di transazioni applicative possono essere abortite, con il

conseguente degrado delle prestazioni percepite dagli utenti.

La gestione del problema, fatta usualmente in modo manuale, è la seguente: l'utente protesta con l'amministratore del sistema locale (*System Administrator*), poichè il *server* funziona correttamente il problema viene riportato al *network administrator* locale, poichè anche la LAN locale è a posto viene interpellato il gestore dei *router* di *backbone*, e così via. È quindi l'utente che fa da monitor della rete.

Questo esempio dimostra quanto sia usualmente difficile diagnosticare e risolvere il problema. La principale causa di tale difficoltà, anche se si passa ad un *monitoring* automatizzato della rete, risiede nel fatto che un singolo problema si propaga e spesso può generare un elevatissimo numero di sintomi. La propagazione va normalmente dal basso verso l'alto, cioè i problemi di livello fisico (perdita di sincronismo) si propaga verso i sistemi terminali e le applicazioni.

I problemi da risolvere sono:

- come correlare i sintomi per isolare il problema;
- come coordinare l'isolamento e la risoluzione del problema tra diversi domini.

Data la complessità del *Problem Management* e poiché la tendenza è verso un continuo aumento dell'esposizione delle aziende a fronte di problemi sulla rete, è evidente quanto diventi essenziale sviluppare dei sistemi che permettano di automatizzare tali funzionalità.

Riduzioni dei rischi

Per quanto le tecnologie di rete diventino nel tempo sempre più affidabili dal punto di vista *hardware*, l'esplosivo aumento della complessità delle reti stesse accresce il rischio di malfunzionamenti o inefficienze dovuti a errori di configurazione e problemi software. Aumento del rischio di guasti ed inefficienze causato da:

- aumento della dimensione e della complessità delle reti;
- aumento della velocità dei *computer*;
- aumento della velocità di cambiamento.

D'altro canto l'effetto dei malfunzionamenti diventa potenzialmente più grave a causa del maggiore impiego di applicazioni distribuite per assolvere compiti di importanza fondamentale in molte aziende. Oggi aereoporti, borse valori, banche e aziende di servizi finanziari, un numero crescente di aziende sanitarie e varie altre imprese dipendono pesantemente dal funzionamento della propria rete di telecomunicazione.

Quanto sia forte questa dipendenza è dimostrato dagli effetti catastrofici del collasso della rete sulla operatività delle aziende più disparate. Fra gli esempi più famosi si possono citare la paralisi aereoporto di NY nel 1992 e, sempre nel 1992, il blocco del sistema di *dispatching* per le ambulanze a Londra. Evidentemente, per un fornitore di servizi di rete questi fenomeni sono deleteri per il proprio *business*: basti pensare alle enormi perdite in borsa dovute al blocco della rete *America On Line* nel 1996.

Riduzioni dei costi di esercizio

Un secondo importante obiettivo della gestione di rete è il contenimento dei costi di esercizio. La rilevanza di questo aspetto si deduce immediatamente dalla analisi dei costi dei sistemi informativi aziendali: i costi di esercizio della rete e dei sistemi costituiscono almeno i due terzi del totale, e possono crescere in alcuni casi fino ad una percentuale del 90%. Tale fattore è destinato ad

accrescere ulteriormente la propria importanza in conseguenza dell'evoluzione tecnologica che produce un continuo calo del costo degli apparati, mentre il costo del personale, particolarmente se dotato di conoscenze tecniche specifiche, tende continuamente a crescere.

L'automatizzazione della gestione della rete tende quindi a ridurre il più importante fattore di costo di un sistema informativo, e non è un caso che, nello stesso scenario organizzativo ed economico, si stiano affermando soluzioni quali i *Network Computer* che mirano ad abbattere l'altro fattore che determina il costo complessivo di un sistema informativo, ossia il costo di gestione dei sistemi e delle applicazioni.

Considerando poi il caso specifico di un gestore di rete, è evidente che questo abbattimento dei costi consente di formulare offerte economicamente convenienti per gli utenti, e quindi vincenti in uno scenario caratterizzato da una competizione sempre maggiore.

Il modello di gestione Manager/Agent

Abbiamo già visto quali siano le funzionalità principali svolte da un sistema di gestione. Per svolgere tali funzionalità il centro di gestione interagisce con i *network element* da gestire attraverso un'infrastruttura di comunicazione dedicata al trasporto delle informazioni di gestione (rete di gestione sovrapposta alla rete gestita), oppure attraverso la stessa rete gestita. Tale colloquio, che può essere svolto a livello locale oppure geografico, si attua attraverso meccanismi di comunicazione che possono essere proprietari, cioè realizzati da un costruttore in modo specifico per la gestione dei propri apparati, oppure standardizzati.

Alla base della gestione di rete c'è l'introduzione negli degli apparati di rete di una strumentazione sempre più completa e sofisticata. Tale strumentazione è in grado di raccogliere una enorme quantità di dati dagli apparati: configurazione e parametri operativi dei singoli elementi che compongono ciascun dispositivo, dati di traffico, tassi di errore, eccetera. Il compito del gestore è quello di analizzare tale massa di informazioni, riconoscere eventuali stati di funzionamento anomalo, se ad esempio viene superata una determinata soglia di carico su un collegamento, ed effettuare le operazioni necessarie a ripristinare il corretto funzionamento della rete.

Nel seguito viene descritto il modello di comunicazione *Manager/Agent* di base, che può essere preso come riferimento nella descrizione tanto dell'*OSI Management* che della gestione di rete SNMP.

Manager e agent

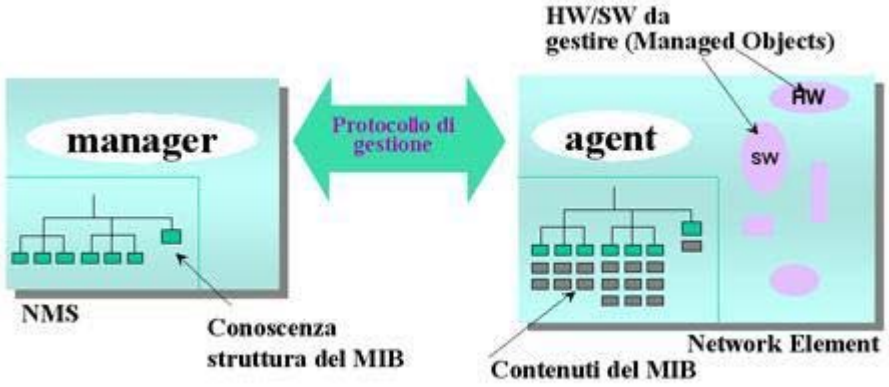
Alla base della gestione di rete c'è quindi un colloquio tra la stazione di gestione e l'apparato gestito. Tale colloquio si esplica in particolare tra due entità, realizzate per mezzo di processi *software*, denominate rispettivamente *Manager*, nel centro di gestione, ed *Agent*, nel nodo gestito. Il trasferimento di informazioni tra *Manager* ed *Agent* avviene in accordo ad un insieme di regole, sintattiche e semantiche, che costituiscono il protocollo di gestione. Il protocollo di gestione è un protocollo di livello applicativo che si appoggia sulla pila protocollare sottostante.

In ambito OSI tale protocollo si chiama CMIP (*Common Management Information Protocol*) ed in ambito SNMP si chiama appunto SNMP (*Simple Network Management Protocol*).

Il modello Manager/Agent/Managed object

Le informazioni che il *Manager* richiede ad un *Agent* sono relative ad Oggetti logici che rappresentano la realtà fisica del dispositivo. Tali oggetti sono contenuti in un *database* denominato MIB (*Management Information Base*), che ha una struttura ad albero.

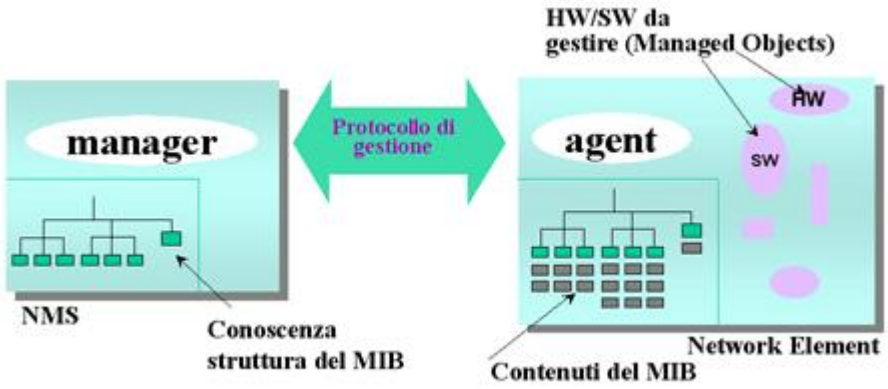
Le informazioni contenute nel MIB forniscono una rappresentazione logica del dispositivo e del suo stato; tale rappresentazione logica permette al *Manager* di accedere ad i dati di un dispositivo in modo non ambiguo. Il *Manager* conosce infatti la struttura del MIB e nel colloquiare con l'agente fa riferimento agli oggetti indicandone la posizione sul MIB. In tal modo il *manager* non è vincolato a conoscere la realtà fisica del dispositivo. Se ad esempio il *Manager* volesse conoscere il numero di pacchetti che un dispositivo ha ricevuto su una sua interfaccia, esso chiede all'agente di restituirgli il valore della variabile corrispondente sul MIB. Sia *Manager* che l'Agente hanno una conoscenza della struttura del MIB; l'agente, oltre alla struttura, possiede anche i dati che la popolano, acquisendoli dalla strumentazione dell'apparato.



Essenzialmente il meccanismo di base su cui è impostata la gestione di rete consiste quindi nell'interrogazione del *database* costituito dal MIB. Il *Manager*, tramite l'Agente, è anche in grado di modificare i dati presenti nel MIB, effettuando quindi operazioni di configurazione dell'apparato. Un Agente, in seguito alla richiesta del *Manager*, è anche in grado di eseguire dei comandi (ad esempio eseguire un operazione di reinizializzazione dell'apparato). I dispositivi possono anche inviare all'applicazione di gestione degli allarmi che indicano il verificarsi di eventi particolarmente significativi nell'apparato gestito.

Il modello di riferimento nel colloquio tra stazione di gestione e dispositivo gestito è quindi detto: modello *Manager* / *Agent* / *Managed Object*. In tale modello:

- il *manager* effettua interrogazioni sui MIB dei diversi dispositivi in rete da lui gestiti, riceve le risposte dagli agenti, riceve dagli agenti notifiche asincrone di particolari eventi verificatisi nel dispositivo, fornisce alle applicazioni di gestione le interfacce di programmazione (API) attraverso le quali le applicazioni possono richiedere al *manager* l'esecuzione di specifiche operazioni (*polling* periodico di un dispositivo, eccetera);
- l'agente mantiene aggiornato il MIB acquisendo informazioni sul dispositivo gestito attraverso la sua strumentazione, risponde alle richieste del *manager*, esegue sugli oggetti gestiti le operazioni richieste dal *manager*, notifica al *manager* eventi asincroni (guasti alle interfacce, eccetera);
- i *managed object* sono entità logiche che rappresentano aspetti specifici del dispositivo fisico, di interesse per ciò che riguarda la gestione di rete (stato delle interfacce, contatori di traffico, tassi di errore, indirizzi, eccetera).



Strumenti di monitoring

Sulla stazione di gestione sono presenti, come si è già visto, le applicazioni di gestione. Queste sono programmi che, utilizzando le API fornite dal *Manager*, possono interrogare l'apparato gestito per fornire all'operatore umano un'interfaccia, per quanto possibile amichevole, che permetta il *monitoring* ed il controllo delle funzionalità della rete. Tali applicazioni mettono a disposizione dell'operatore strumenti per apprendere in modo automatico la struttura della rete e visualizzarne la mappa, per controllare e visualizzare lo stato dei diversi elementi di rete, per misurare il traffico ed i tassi di errore sui collegamenti, per leggere o modificare il valore di una qualsiasi variabile del MIB. Molte applicazioni di gestione forniscono, inoltre, una rappresentazione visiva immediata dell'apparato gestito.

Limiti odierni della gestione

Anche se negli ultimi anni, come si vedrà meglio in seguito, sono stati evidenti i progressi fatti nella gestione di rete, molti sono ancora quelli da fare se si vuole far fronte in modo efficiente all'enorme aumento della complessità e delle dimensioni delle reti. Fare gestione di rete oggi significa infatti osservare e controllare lo stato di ciascun *network element* separatamente. Ma la rete non è una mera collezione di elementi, bensì un insieme di attività *end-to-end* che andrebbero analizzate come tali. Non è ad esempio sufficiente sapere se un determinato *router* sta funzionando bene, ma servirebbe piuttosto avere strumenti per capire come si comporta una certa applicazione informatica in azienda, identificare gli eventuali colli di bottiglia per il buon comportamento di quella applicazione, ottimizzare, nel suo insieme il comportamento di tutte le applicazioni in rete.

Oggi non ci sono strumenti automatici per gestire la rete: i centri di gestione si limitano a raccogliere informazioni dagli apparati gestiti ed a presentarle, in forma più o meno amichevole, all'operatore. Le decisioni sulle azioni da intraprendere sono effettuate dall'operatore in base ai dati raccolti e, soprattutto, alla propria esperienza: la rete è gestita dagli uomini, non dalle stazioni di gestione.

Un altro problema è che la gestione di rete, così come è concepita oggi, è poco scalabile: in un modem ci sono poche variabili 15-20, in un *router* 4000-8000, in un nodo ATM potrebbero esserci anche decine di migliaia di variabili: come è possibile tenere sotto controllo a livello centralizzato l'enorme mole di informazioni significative presenti in una rete di grandi dimensioni? Gli obiettivi da perseguire sono:

- la realizzazione di sistemi di rete *plug&play*, che possano cioè essere inseriti in rete senza bisogno di complesse operazioni di configurazione e che si adattino automaticamente ai cambiamenti;
- l'introduzione in rete di meccanismi che permettano di ottimizzare in modo automatico il comportamento della rete e delle applicazioni che la utilizzano;
- la definizione di strumenti che consentano l'identificazione automatica dei guasti.

Mentre per i primi due punti la strada da compiere è ancora lunga, per ciò che riguarda il *fault management* automatizzato ci sono già strumenti, basati su sistemi esperti o su altre tecnologie informatiche, che permettono una identificazione automatica degli inconvenienti attraverso una correlazione dei sintomi da essi generati.

Da quanto detto dovrebbe emergere con chiarezza che è di fondamentale importanza gestire la rete con accuratezza. Una rete costituita da apparati tecnologicamente all'avanguardia ma dotata di un sistema di gestione insufficiente si comporta molto peggio di una rete meno avanzata, ma ben gestita.

Gli inconvenienti che derivano dal trascurare gli aspetti di gestione possono essere l'impossibilità di garantire la qualità dei servizi offerti, elevati tempi di indisponibilità, costi di manutenzione incontrollabili, una eccessiva complessità nella realizzazione di espansioni e aggiornamenti o altri ancora. Nella scelta della apparecchiatura da acquistare deve essere posta quindi la massima attenzione anche su quali strumenti sono disponibili per gestirle.

Manager di managers

Per integrare su un'unica stazione la gestione di una rete complessa si possono seguire diversi approcci. Uno di questi consiste nel realizzare un sistema di integrazione che, dialogando con le stazioni di gestione proprietarie, fornisca su un unico elaboratore un'interfaccia per la gestione di tutta la rete indipendente dal particolare tipo di apparato gestito. Tale stazione di gestione centralizzata può essere vista come un *manager* di *managers*: cioè come una stazione di gestione che interagisce con le stazioni di gestione proprietarie anziché con gli elementi di rete. Questo approccio richiede però che il sistema di integrazione sia realizzato in modo da poter colloquiare con tutte le stazioni di gestione proprietarie. Ciò si basa però sull'impiego di un sistema *software* specificamente realizzato per il contesto in cui deve operare, quindi con elevati tempi e costi di sviluppo.

È un aspetto molto critico, in questo caso, anche la necessità di seguire, con continui aggiornamenti del *software*, le evoluzioni dei sistemi di gestione proprietari: ad ogni nuova funzionalità introdotta dal costruttore sarà necessario, per poterla fruttare, un aggiornamento del *software* del sistema integratore. Con tutte le limitazioni di cui si è accennato, questo approccio è però, in alcuni casi, l'unico praticabile.

Unificazione della gestione

Volendo invece unificare realmente la gestione, su un'unica stazione che veda in modo omogeneo tutti gli elementi di rete, sarà necessario standardizzare il colloquio tra *manager* ed agente, sia dal punto di vista procedurale, sia per ciò che riguarda la sintassi con cui viene effettuata la codifica delle informazioni trasportate. È cioè necessario standardizzare il protocollo di comunicazione di

livello applicativo per le informazioni di gestione.

La standardizzazione del protocollo di gestione permette al *manager* di colloquiare direttamente con tutti gli apparati da gestire. Per gestire a livello centralizzato tutta la rete è però anche importante che i diversi apparati possano essere visti dalla stazione di gestione in modo omogeneo. Poiché la struttura degli apparati è logicamente rappresentata, come già visto nel capitolo precedente, attraverso il *Management Information Base* (MIB), l'unificazione della gestione richiede anche la disponibilità di un MIB standard che possa rappresentare in modo omogeneo, almeno per le funzionalità di base, tutti gli elementi di rete. Attraverso il MIB standard le applicazioni di gestione di tipo generale possono accedere in modo omogeneo alle funzionalità di base per la gestione di tutti i dispositivi.

È però spesso necessario agire anche su aspetti specifici dei diversi apparati; tali aspetti, che sono descritti da MIB proprietari, sono normalmente gestiti da applicazioni di gestione specializzate fornite dai costruttori degli apparati. Perché tali applicazioni possano essere facilmente rese disponibili sulle diverse stazioni di gestione è importante disporre di interfacce di programmazione (*Application Programming Interface* - API) standard che rendano indipendenti le applicazioni dalla piattaforma *software* di gestione.