Ministero dell'Istruzione, dell'Università e della Ricerca Servizio Automazione Informatica e Innovazione Tecnologica

Modulo 9

Amministrazione e gestione di una rete

ForTIC

Piano Nazionale di Formazione degli Insegnanti sulle Tecnologie dell'Informazione e della Comunicazione

Percorso Formativo C

Materiali didattici a supporto delle attività formative

2002-2004

Promosso da:
- Ministero dell'Istruzione, dell'Università e della Ricerca, Servizio Automazione Informatica e Innovazione Tecnologica
- Ministero dell'Istruzione, dell'Università e della Ricerca, Ufficio Scolastico Regionale della Basilicata

Materiale a cura di:
- Università degli Studi di Bologna, Dipartimento di Scienze dell'Informazione
- Università degli Studi di Bologna, Dipartimento di Elettronica Informatica e Sistemistica

Editing:
- CRIAD - Centro di Ricerche e studi per l'Informatica Applicata alla Didattica

Progetto grafico:
- Campagna Pubblicitaria - Comunicazione creativa

In questa sezione verrà data una breve descrizione del modulo.

Gli scopi del modulo consistono nel mettere in grado di:

- Conoscere le procedure per garantire la sicurezza di una rete.
- Gestire gli *account*.
- Progettare, installare e mantenere una struttura di directory.
- Saper assegnare agli utenti appropriati diritti.
- Saper effettuare procedure di *backup*, *recovery* e controllo.

Il modulo è strutturato nei seguenti argomenti:

- **Amministrazione di una rete**
  - Implementare appropriate procedure per garantire la sicurezza di una rete.
  - Gestire gli *account* degli utenti incluso script di login.
  - Progettare, installare e mantenere una struttura di directory.
  - Assegnare agli utenti i diritti appropriati per accesso a file, applicazioni e risorse.
  - Usare un sistema di *account* su una rete.
  - Effettuare procedure di *backup* e *recovery* e controllo.
  - Discutere gli aspetti connessi con le varie tecniche di autenticazione degli utenti.

Introduzione

Amministrazione di una rete

Franco Callegati
Walter Cerroni

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.

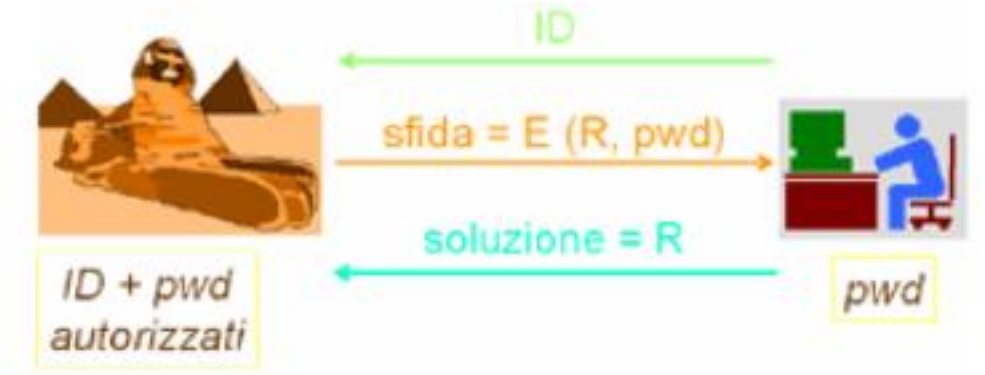
Protezione della password

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:



Schema di dialogo per autenticazione con sfida utilizzando crittografia simmetrica

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:

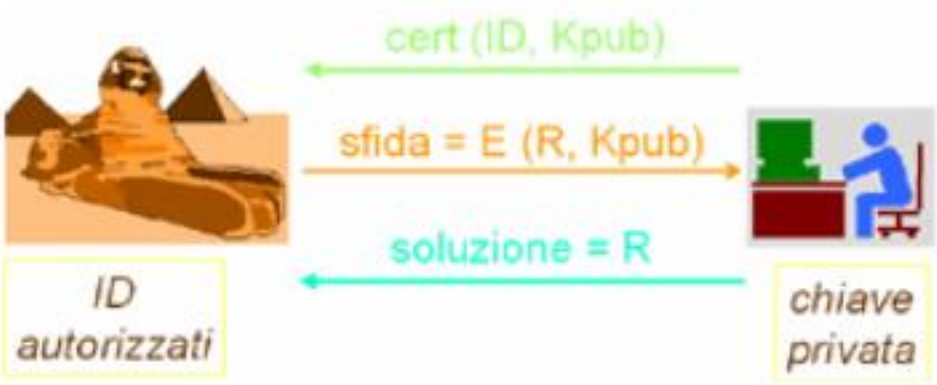
sfida = E(R, pwd)

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:



Schema di dialogo per autenticazione con sfida utilizzando crittografia asimmetrica

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*).

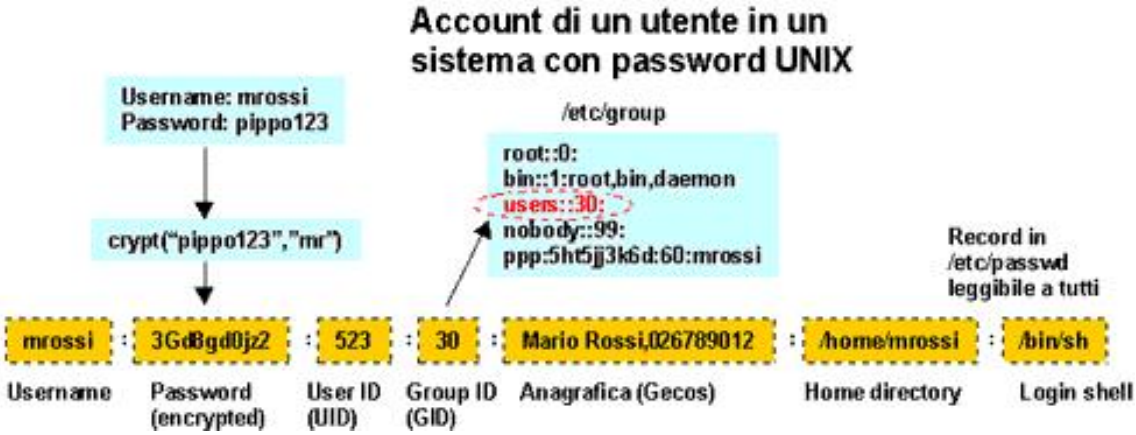
Oltre allo UID, ad ogni utente è associato un GID (*Group ID*), che identifica a quale gruppo di utenti esso appartiene. Si tratta di un valore numerico che viene tradotto in forma simbolica mediante il *file* /etc/group. Esso viene utilizzato, in modo analogo allo UID, per marcare la proprietà di una 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.



Esempio di stringa di definizione di un account di utente in un sistema Unix

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.

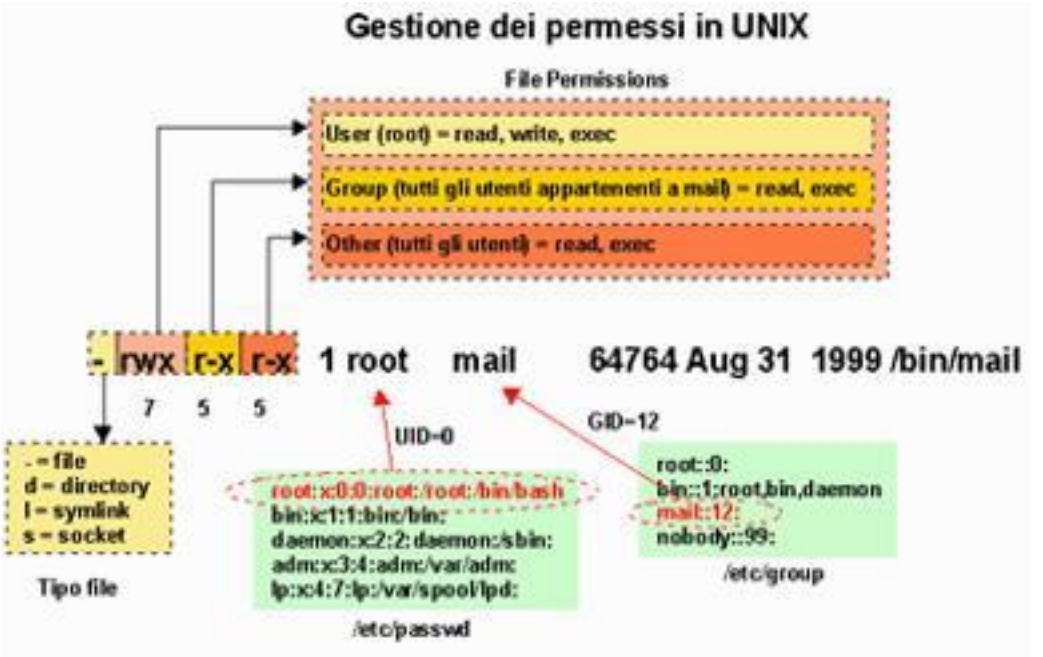
Gestione dei permessi

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.

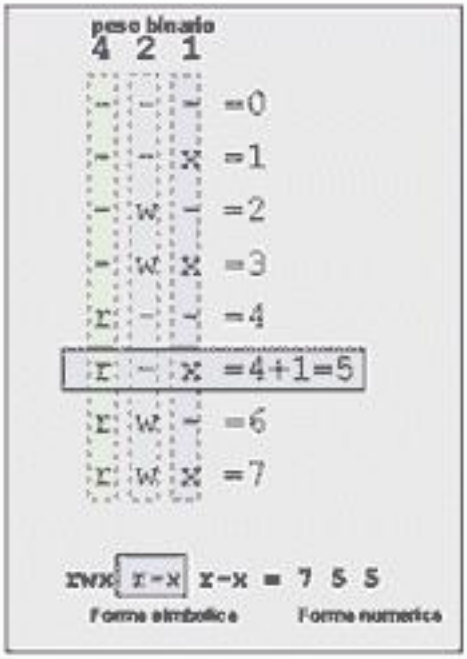


Proprietà di un file Unix (output di ls -1)

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).



Rappresentazione dei permessi

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

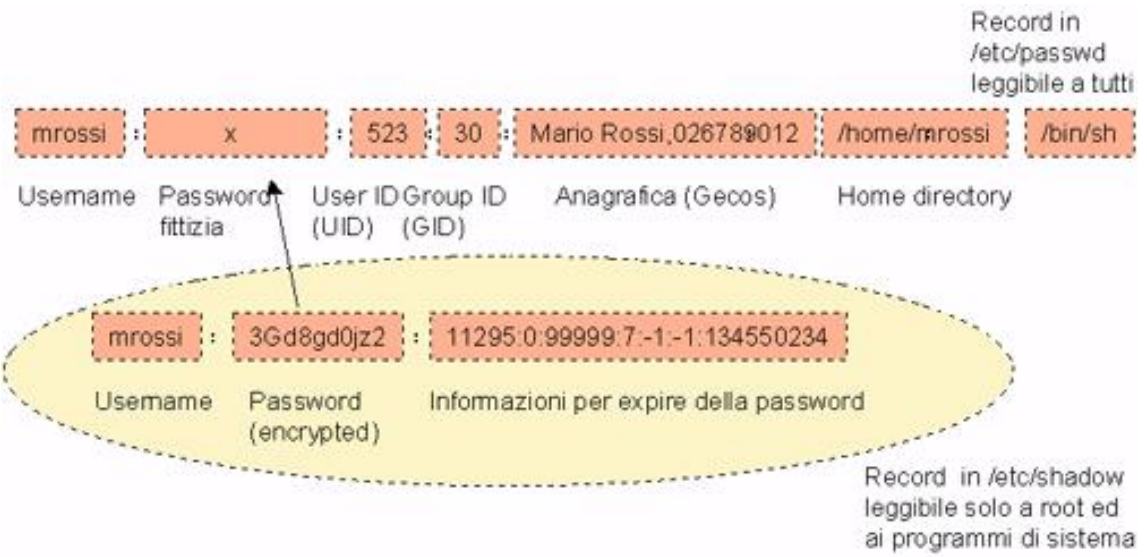
Tradizionalmente in *UNIX* l'autenticazione avviene assegnando ad ogni utente del

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.



Account di un utente in un sistema con shadow password

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

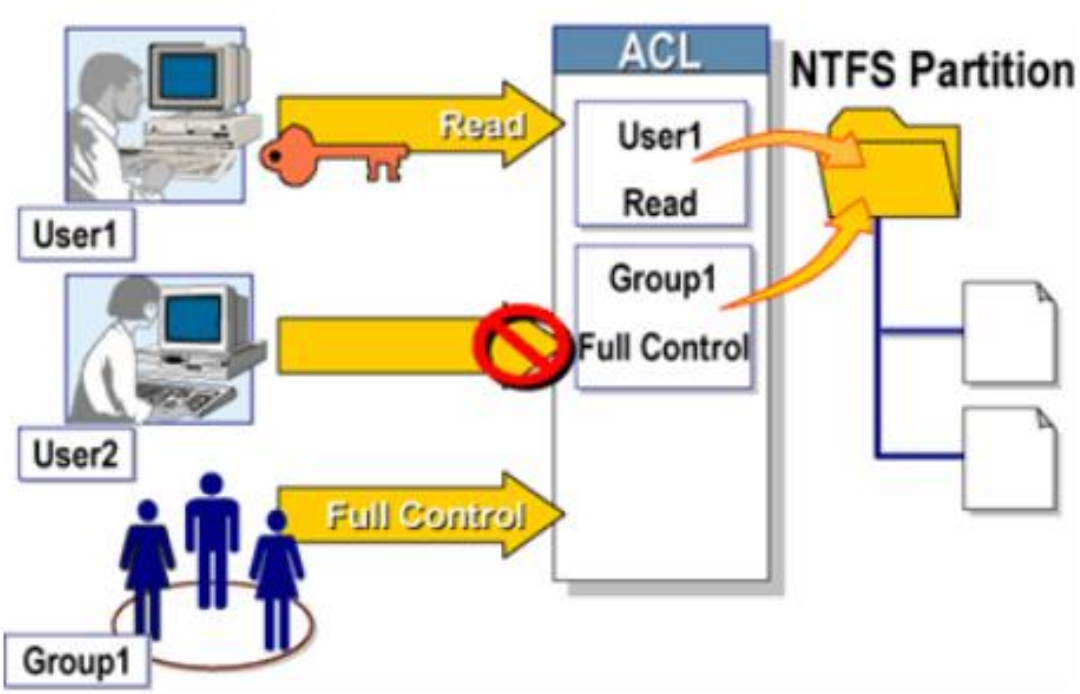
Così come in *Unix* anche in *Windows* gli *account* utente possono essere raggruppati in Gruppi di utente. Un Gruppo è un insieme di *account* utente.

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



Esempio di controllo di accesso ad una cartella condivisa

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.

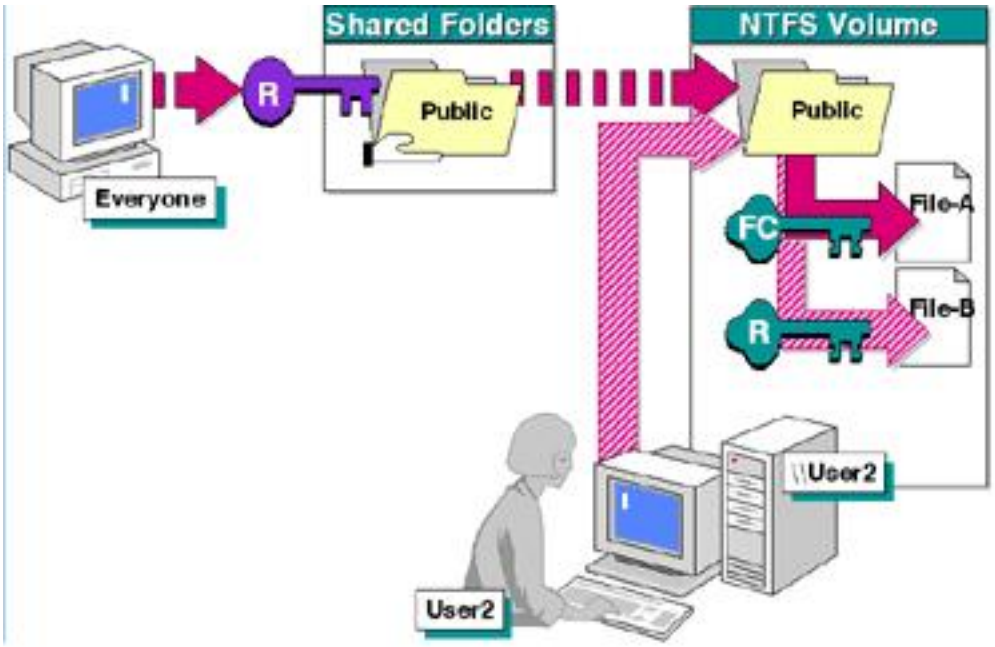
Quando una partizione o un volume viene formattato NTFS, *Windows* 2000 automaticamente assegna sulla *root* il permesso NTFS *Full Control* al gruppo *Everyone*, che dunque avrà di *default* tale permesso NTFS su tutti i *files* e cartelle create in quella partizione o volume, a meno che non diversamente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



Interpretazione dei permessi

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*.

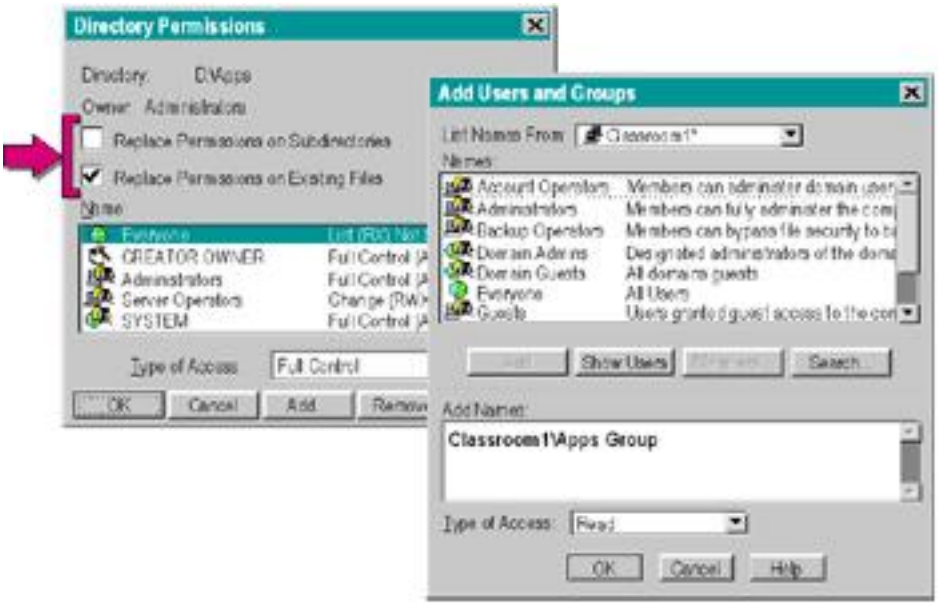
Assegnazione dei permessi

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.



Assegnazione dei permessi

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

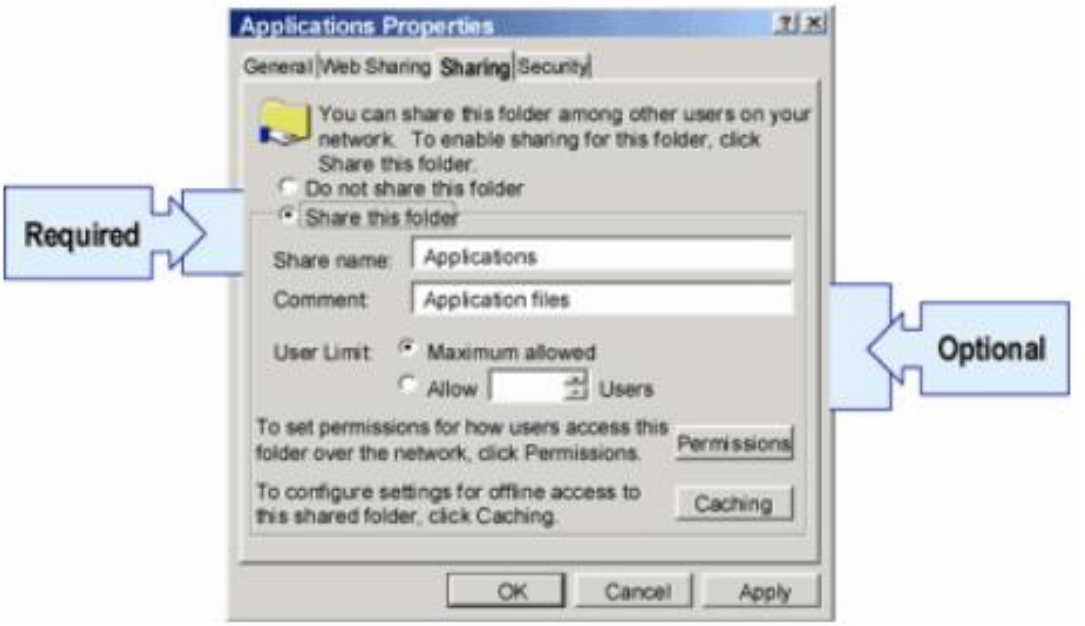
Quando si condivide una cartella:

- 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*.



Finestra di set up della condivisione

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:

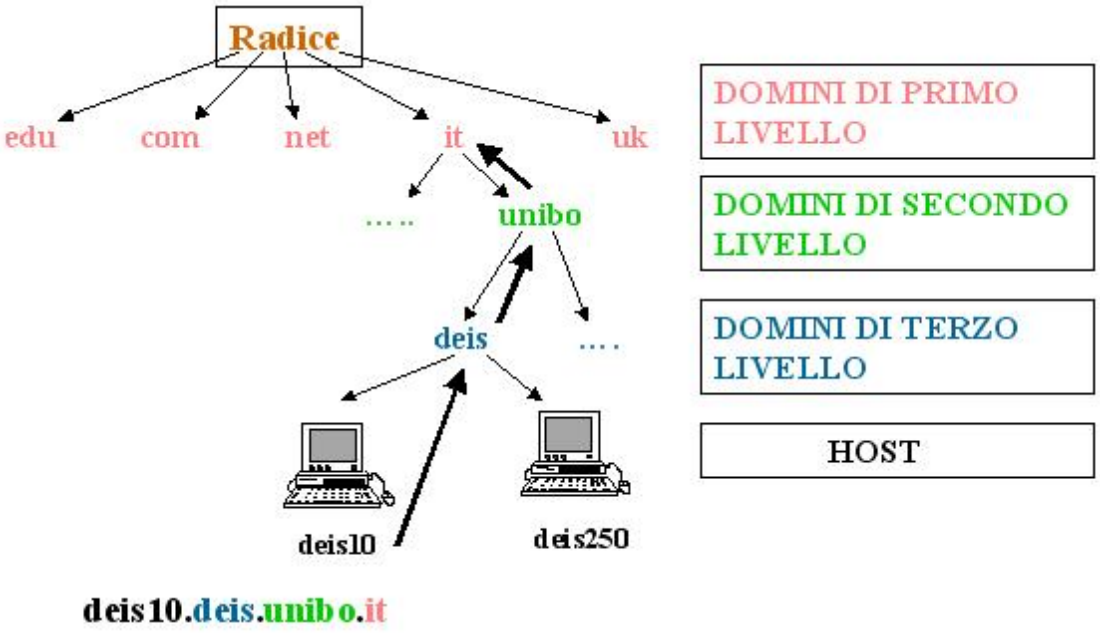
- Internet è partizionata in aree logiche dette domini con un nome univoco.
- I singoli domini possono a loro volta essere suddivisi in sottodomini.

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*.



Esempio di nome simbolico nella gerarchia DNS

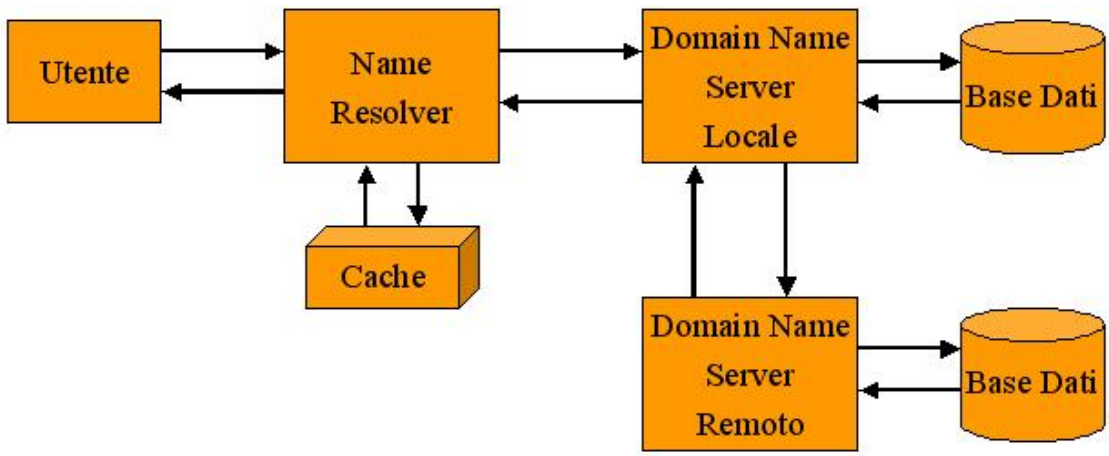
Utilizzazione di DNS

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.



Esempio di ricerca nel DNS utilizzando il Name Resolver

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.

Esempio di accesso utente su di un client che utilizza il NIS

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 <master-**NIS**>

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.

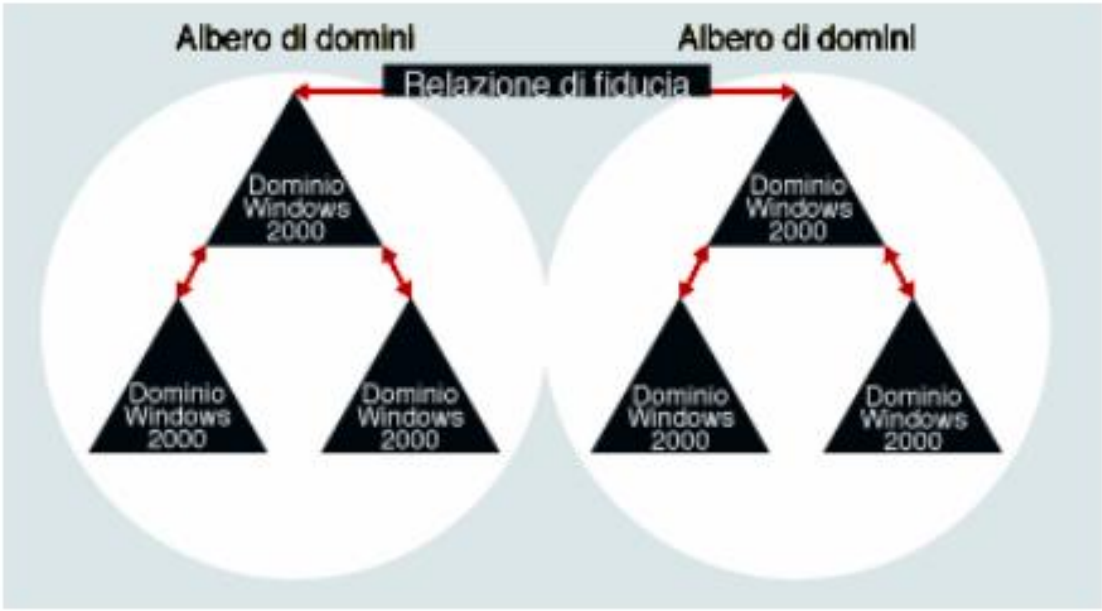
Domini. In *Windows* 2000 le politiche di sicurezza sono ancora legate al dominio, così come esistono ancora i gruppi *Domain Admins*. I domini di *Active Directory* non usano più lo *standard* di nomi **NetBIOS** a quindici caratteri; anche se questo tipo di nomi risulta ancora presente per offrire 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.



Alberi di domini che stabiliscono relazioni di fiducia fra loro

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*.

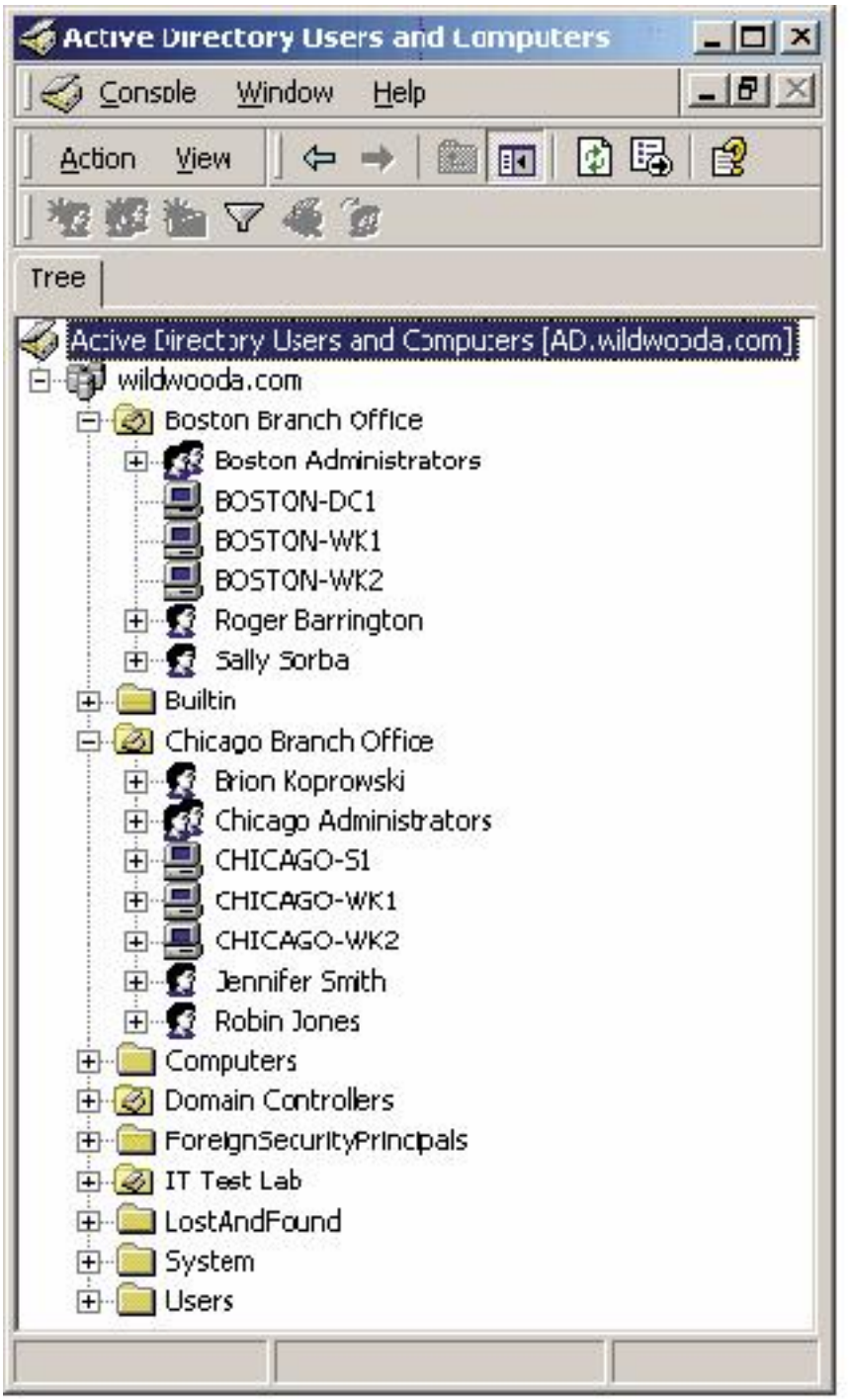
Active Directory fornisce a *Windows* 2000 molte funzionalità che diversificano questo sistema operativo da NT 4.0 e lo rendono più facile da usare nella gestione delle grandi aziende. Queste nuove funzionalità comprendono:

- 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).



Finestra Active Directory Users and Computers

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.

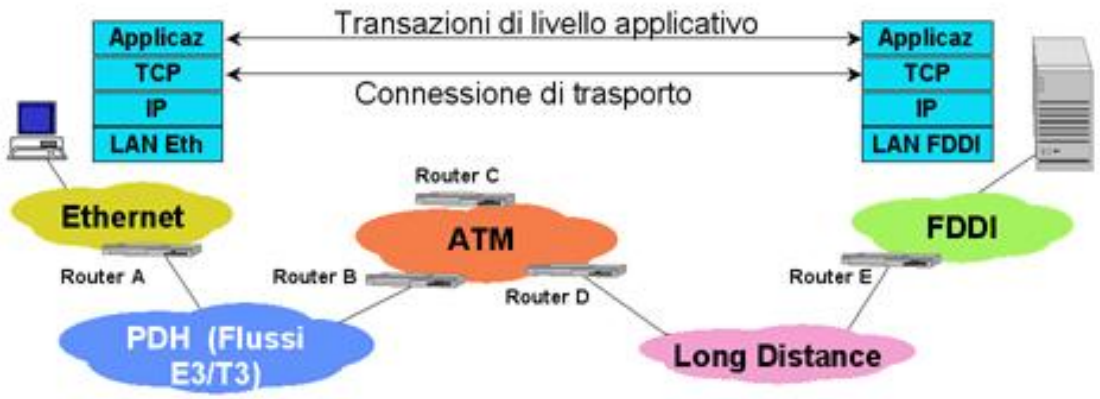
Quando le dimensioni del sistema sono rilevanti (centinaia o migliaia di utenti), ed a maggior ragione se è anche distribuito territorialmente, le infrastrutture di rete utilizzate possono diventare realmente molto complesse. Tali infrastrutture sono prevalentemente costituite da un numero sempre crescente 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).

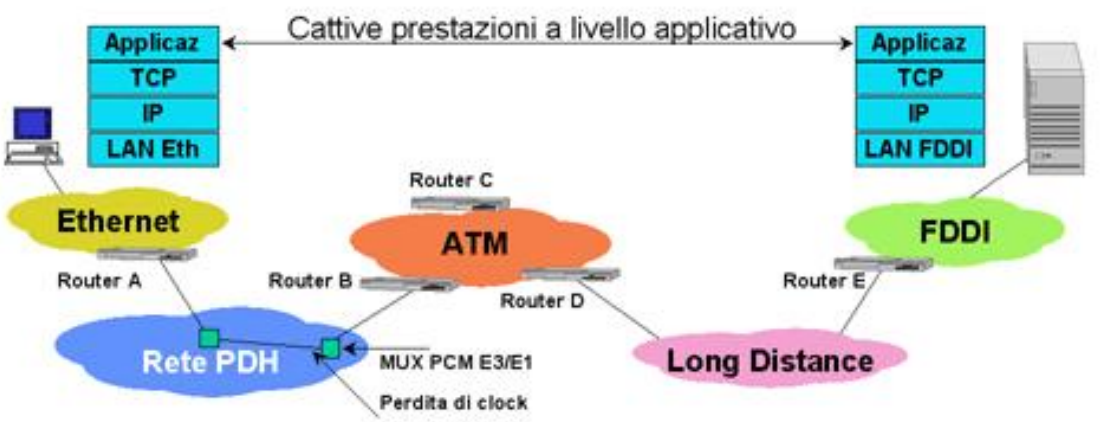


La gestione di rete

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.



Individuazione dei guasti

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.

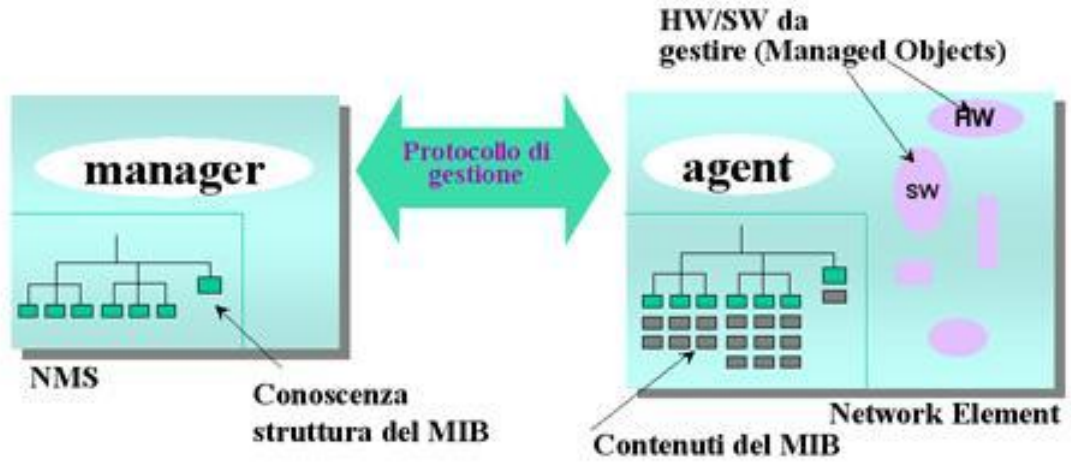
Manager e agent

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.



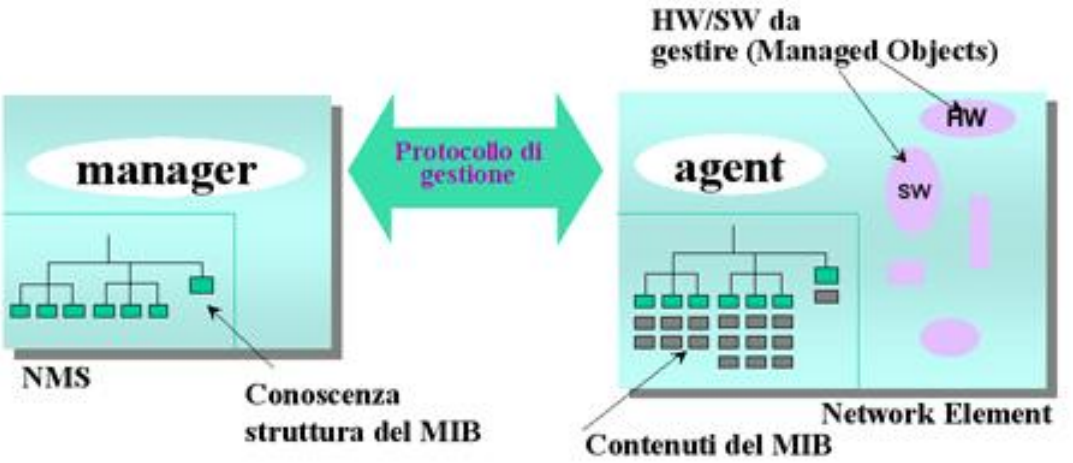
Il modello Manager/Agent/Managed object 1

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).



Il modello Manager/Agent/Managed object 2

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.

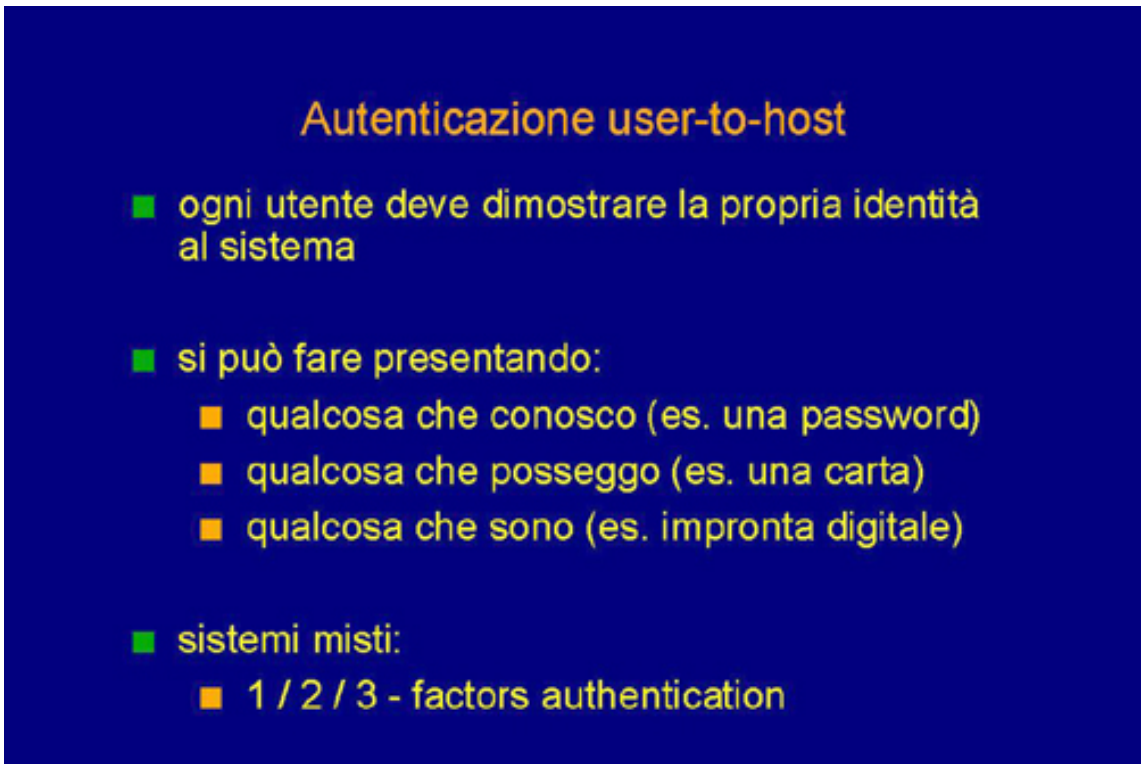
Approfondimento

Implementazione di sistemi di autenticazione

Franco Callegati

9.1.5 (Usare un sistema di account su una rete) - 9.1.7 (Discutere gli aspetti connessi con le varie tecniche di autenticazione degli utenti)

Autenticazione user-to-host

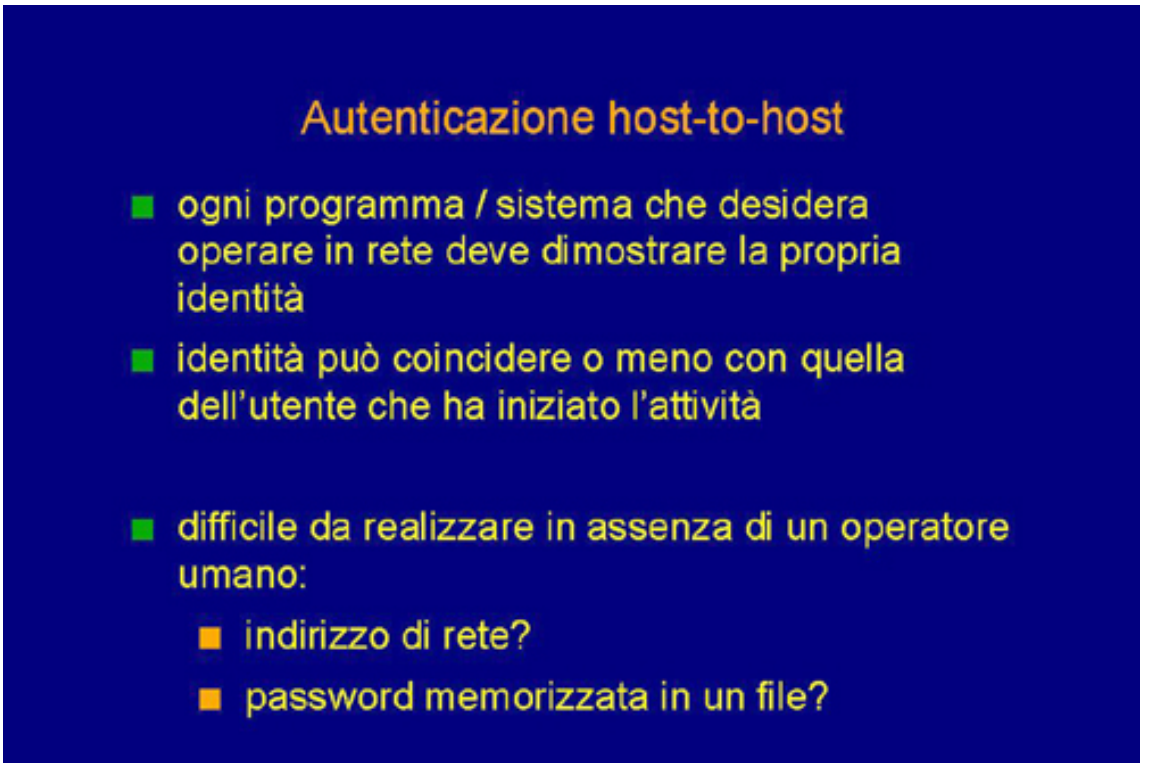


Autenticazione user-to-host

Alla base di tutti quanti i sistemi di sicurezza risiede l'identificazione delle persone e dei programmi che sono abilitati ad operare sul sistema. Abbiamo visto in precedenza che questo prende tecnicamente il nome di autenticazione. In particolare, è possibile avere due tipi di autenticazione all'interno di un sistema informatico. Il primo tipo di autenticazione è quella cosiddetta *user-to-host*, in cui ogni utente deve dimostrare la propria identità al sistema, ossia deve dimostrare di avere diritto di utilizzare una certa risorsa di calcolo, o di attivare una certa procedura di calcolo. Questo può esser fatto in almeno tre modi diversi. Il modo più banale è quello di dimostrare la propria identità dicendo qualcosa che soltanto quest'utente teoricamente dovrebbe conoscere, banalmente questo è il normale sistema di *username* e *password* ed è un sistema non molto sicuro. Per rafforzare la sicurezza di questo sistema potrebbe essere preferibile utilizzare qualcosa che si possiede, ad esempio una carta. Questo significa che l'accesso al sistema viene controllato non soltanto in base allo *username* e *password* che vengono digitati su una tastiera, ma viene controllato anche in base ai dati

contenuti in una carta di tipo magnetico, o di tipo elettronico, da introdursi nel sistema. Se desideriamo avere delle funzionalità di sicurezza ancora più forte, tipicamente si ricorre a qualcosa che io impersonifico, qualcosa che io sono, ovvero una mia caratteristica fisica. È tipico in questo campo l'uso delle impronte digitali, oppure del tono della voce, o di qualche altra caratteristica fisica: la dimensione della faccia, il colore degli occhi e così via. In generale è possibile avere dei sistemi misti: si parla in questo caso di sistemi a 1, 2, 3 *factors authentication*, ossia sistemi di autenticazione che, per permettere l'accesso al sistema, richiedono la combinazione di uno, due, oppure tre fattori dei tipi che abbiamo appena menzionato.

Autenticazione host-to-host

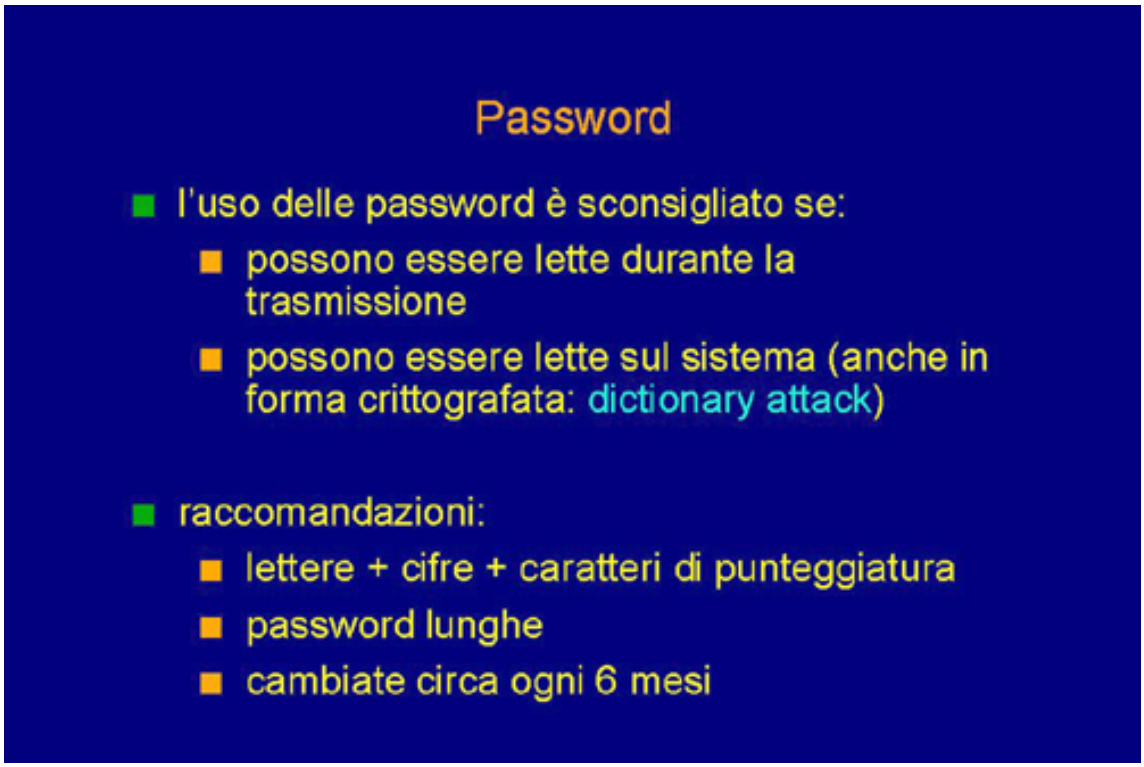


Autenticazione host-to-host

All'interno di un sistema informativo però non ci sono soltanto utenti che operano, ci sono anche delle procedure, dei processi e dei sistemi che molto spesso operano in modo automatico anche in assenza di un operatore di tipo umano. Anche questi processi, questi sistemi, devono essere autenticati ed in questo caso si parla di una autenticazione di tipo *host-to-host*. Non c'è più un essere umano che deve autenticarsi nei confronti di un elaboratore, ma c'è un elaboratore che deve autenticarsi nei confronti di un atro elaboratore per ottenere una funzionalità, un servizio o dei dati. In questo caso, l'identità del programma, del processo, che sta cercando di svolgere questa funzione può coincidere o meno con quello dell'utente che ha iniziato l'attività. È in ogni caso un campo molto delicato e molto difficile da realizzare, soprattutto in assenza di un operatore umano, perché tutte le tecniche che noi abbiamo sviluppato in passato sono basate sulla presenza di un operatore umano, non sono quindi adatte ad autenticare programmi o sistemi. Infatti è stato proposto l'utilizzo degli indirizzi di rete, ossia abilitare la funzionalità di un certo tipo solo per processi, programmi, che

provengono da una certa classe di rete. Oppure è stato proposto di memorizzare la *password* da utilizzarsi all'interno di un *file*, ma in questo caso la *password* non sarebbe protetta come dovrebbe esserlo, teoricamente, quando risiede all'interno del cervello umano. Quindi, in conclusione, l'autenticazione *host-to-host* è uno dei campi più delicati e ancora risolti nel modo meno soddisfacente, nell'attuale schema di sicurezza dei sistemi informativi.

Password

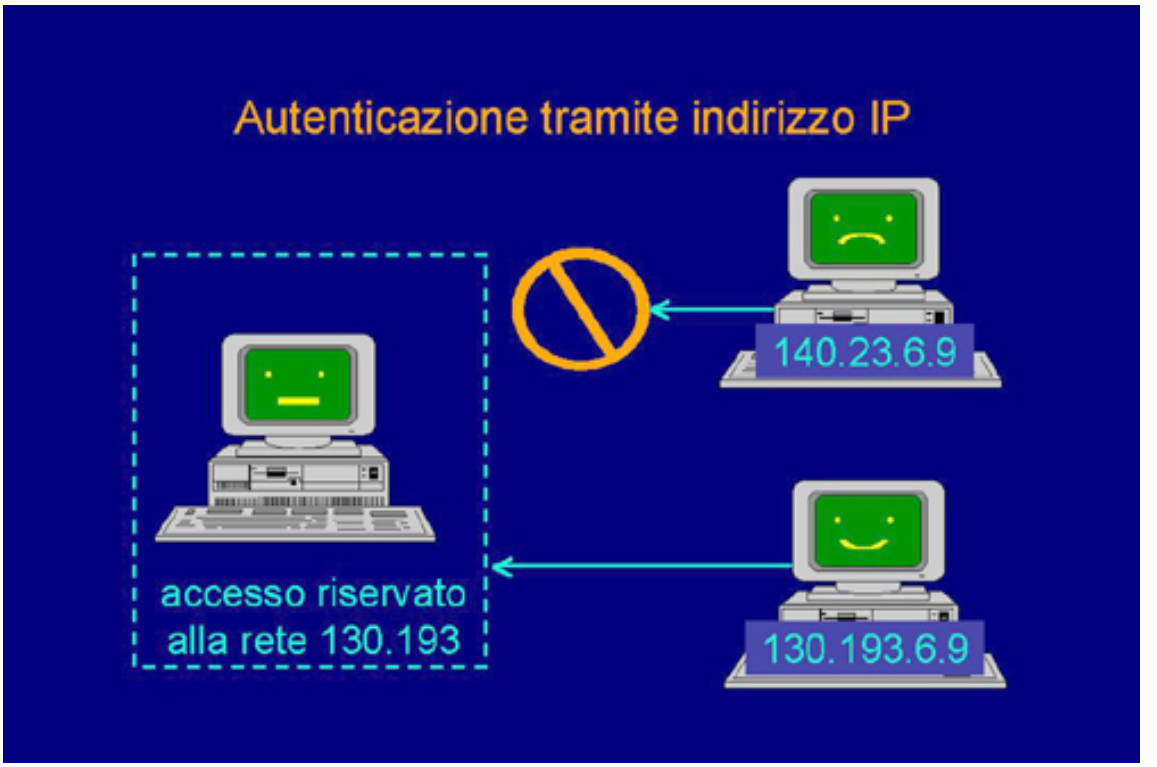


Password

Veniamo adesso ad esaminare un po' più nel dettaglio alcune delle tecniche con cui è possibile autenticare l'utente nei confronti del sistema. La classica tecnica che viene utilizzata è quella della *password*. L'utilizzo di *username* e *password* è fortemente sconsigliabile non in assoluto, ma solo se sussistono alcune possibilità. La prima possibilità è che la *password* possa essere letta durante la sua trasmissione in rete fra l'utente e il servizio a cui desidera accedere, perché in questo modo può essere facilmente catturata e quindi riutilizzata per fare delle altre sessioni di lavoro non autorizzate. Inoltre le *password* sono da sconsigliarsi se vengono mantenute su un sistema informativo al cui interno operano delle persone che possono leggere queste *password*. È chiaro che se le *password* sono conservate in chiaro in un *file* è veramente una operazione banale leggerle, ma anche nel caso in cui le *password* siamo mantenute in forma crittografata, ossia cifrata, ossia segrete, diventano pericolose. È infatti possibile svolgere un attacco di tipo basato su dizionario: questo significa che l'attaccante, dopo aver preso visione di quelle che sono le *password* crittografate cifrate, prova a cifrare in tanti modi diversi un dizionario di parole più comunemente usate. Siccome gli utenti hanno la tendenza ad usare come *password* delle parole semplici e di senso compiuto, tipo il proprio nome, il nome dei propri figli, il

nome dei propri parenti o del luogo in cui sono stati in villeggiatura durante l'ultimo anno, ecco che questo genere di attacchi sorprendentemente ha sempre un grande successo. Si stima che mediamente qualunque sistema informativo abbia un numero di *password* banali da indovinare compreso tra il 30 e il 40 per cento delle *password* totali utilizzate all'interno del sistema. Esistono quindi delle raccomandazioni specifiche da fare agli utenti che siano per qualche motivo obbligati ad usare le *password* tradizionali. La prima raccomandazione è quella di non utilizzare una *password* solo di caratteri alfabetici, ma di mischiare al suo interno lettere, cifre e caratteri di punteggiatura. Questo è sufficiente a parare gli attacchi di tipo dizionario, perché ovviamente all'interno di un dizionario ben difficilmente trovano posto parole di questo genere. Inoltre la *password* deve essere lunga: si consiglia un minimo di 8 caratteri, perché se la *password* è troppo corta diventa facile cercare di indovinarla. Analogamente non bisognerebbe mantenere la propria *password* per anni: si dice che mediamente al massimo ogni 4/6 mesi la propria *password* dovrebbe essere cambiata.

Autenticazione tramite indirizzo IP

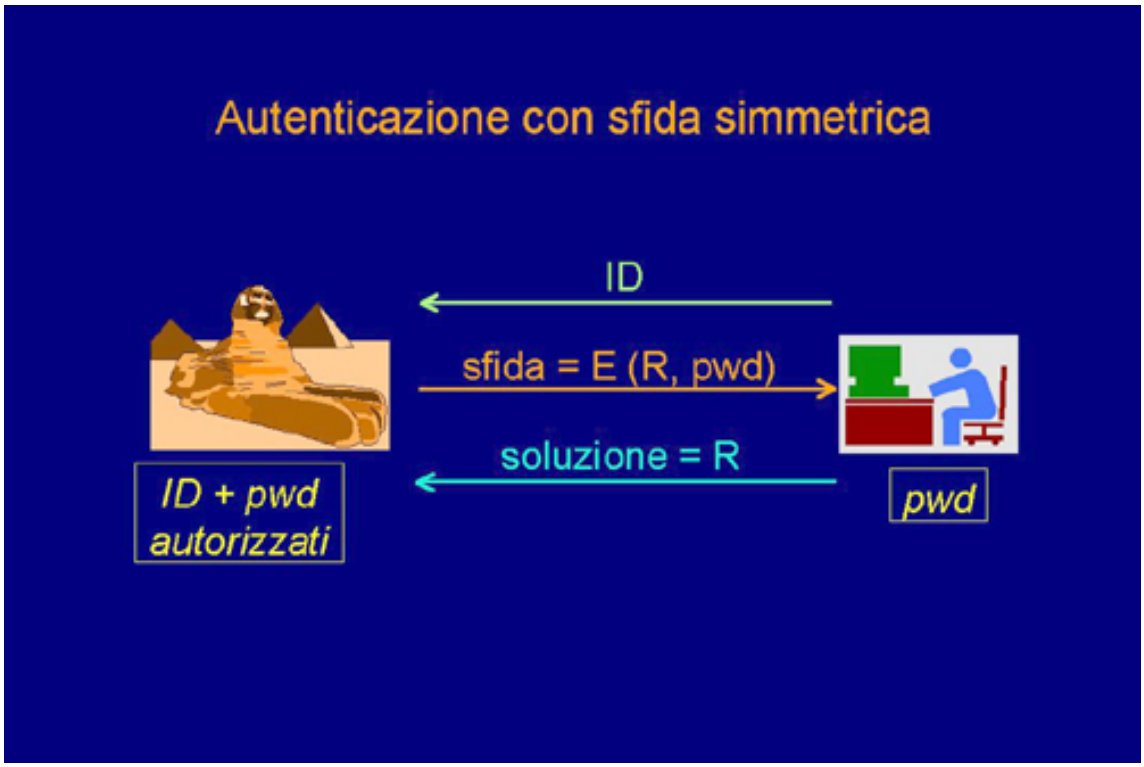


Autenticazione tramite indirizzo IP

Un'altra possibilità di autenticare sia un utente sia un processo che richiede un servizio, è quello di fare affidamento sugli indirizzi di rete, ossia identificare la postazione di lavoro da cui l'utente sta richiedendo un certo servizio. Questo comunemente viene fatto utilizzando gli indirizzi di rete IP. Ad esempio, in questo schema abbiamo un calcolatore che contiene dati riservati; in particolare questi dati sono riservati alle postazioni di lavoro che hanno indirizzi appartenenti alla classe 130.193. Ciò significa che se un utente è seduto davanti ad un *computer* che ha un indirizzo di questa classe, come in questo caso (130.193.6.9), allora le sue richieste di servizio verranno esaudite. Ma se il medesimo utente si sedesse di fronte ad un *computer* appartenente ad una

classe diversa, qui abbiamo un *computer* con indirizzo 140.23.6.9, allora la sua richiesta verrebbe automaticamente rifiutata dal sistema. Questo tipo di autenticazione è sicuramente un tipo di autenticazione debole, perché è molto facile falsificare gli indirizzi IP, è addirittura banale su un normale PC. Un elaboratore dotato del sistema operativo *Windows*, o anche di un altro sistema operativo, che sia sotto il controllo dell'utente che gli è seduto davanti, permette di impostare a proprio piacimento l'indirizzo IP e a questo punto io potrei impostare quell'indirizzo che mi da accesso a quei dati riservati. Quindi l'autenticazione basata sugli indirizzi di rete è da sconsigliare fortemente in ambienti con grado di sicurezza medio/alto.

Autenticazione con sfida simmetrica

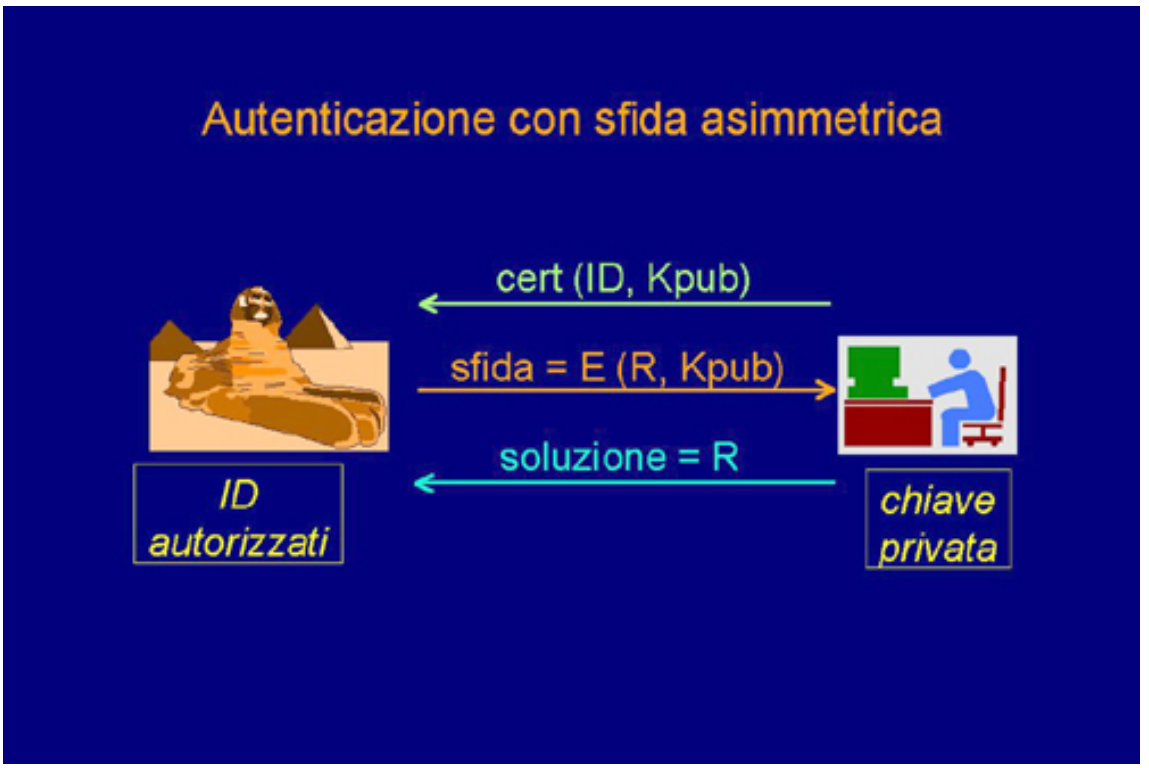


Autenticazione con sfida simmetrica

Veniamo ad alcuni sistemi invece che presentano delle caratteristiche di sicurezza molto migliori, per quanto riguarda l'autenticazione. Un modo molto bello di fare autenticazione utilizzando le *password* senza mostrarle in rete, sono i sistemi cosiddetti di autenticazione a sfida simmetrica. Ipoteticamente l'utente conosce la propria *password*, che è nota anche al sistema a cui noi vogliamo fare accesso, il quale contiene l'elenco degli identificativi (gli *username*) e le relative *password* che sono autorizzate ad accedere al sistema. Quando l'utente desidera accedere al sistema, invia un primo pacchetto in rete che specifica lo *username*, ossia l'ID, con cui lui desidera accedere al sistema. A fronte di questa richiesta di accesso, il sistema gli invia indietro un messaggio di sfida: banalmente la sfida consiste in un numero random R, un numero casuale, che è stato cifrato (E sta per *encrypted*) con la *password* relativa a quell'utente. Se l'utente è veramente chi dice di essere allora è in grado di decifrare questo pacchetto usando la propria *password* e quindi di rispondere alla sfida mandando la soluzione. La soluzione è semplicemente il numero casuale R che il

sistema aveva generato. Se il numero casuale non viene mai più ripetuto in qualunque transazione di tipo futuro, allora questo è un sistema di sicurezza perfetto per quanto riguarda l'autenticazione in rete. Vuol dire che chiunque stia osservando lo scambio di messaggi tra la postazione *client* e la postazione *server*, non può da questo scambio di messaggi catturare in alcun modo la *password*.

Autenticazione con sfida asimmetrica

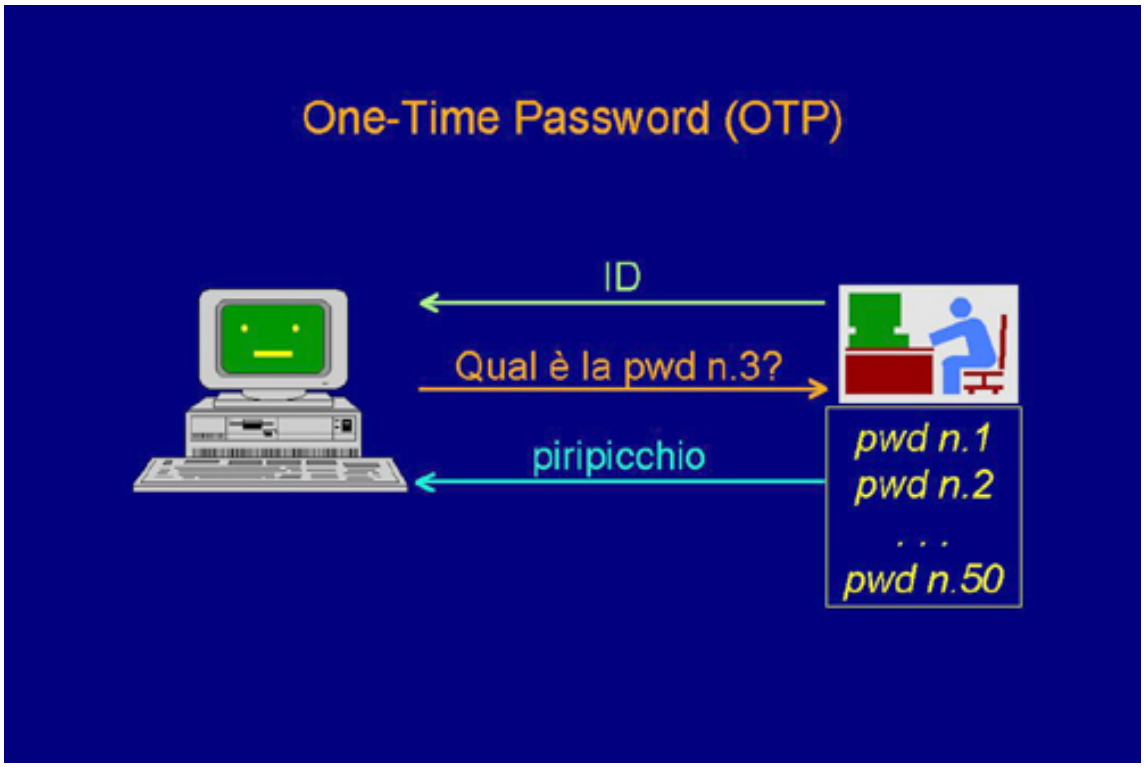


Autenticazione con sfida asimmetrica

Questo sistema può essere esteso e migliorato utilizzando gli schemi di crittografia asimmetrica. Nello schema di crittografia asimmetrica non è necessario che il sistema cui noi desideriamo collegarci conosca anche lui la nostra *password*, è sufficiente che lui conosca semplicemente la nostra chiave pubblica. Questi generi di sistemi si basano sul principio che vedete qui illustrato: ad ogni utente deve essere stata distribuita una coppia di chiavi private e chiavi pubbliche, in particolare l'utente dispone sulla sua postazione di lavoro della propria chiave privata. Il sistema cui noi vogliamo accedere memorizza l'elenco degli identificativi (degli *username*) che sono autorizzati ad accedere al sistema. Quando l'utente desidera accedere manda il proprio certificato a chiave pubblica contenente il proprio identificativo e la propria chiave pubblica. Il sistema, basandosi sulla chiave pubblica estratta da questo certificato, manda una sfida. La sfida consiste nel solito numero casuale R crittografato con la chiave pubblica del destinatario. A questo punto il destinatario, se è veramente chi dice di essere, possiede la chiave privata corrispondente ed è quindi in grado di mandare indietro la soluzione della sfida, che consiste nel decifrare questo pacchetto usando la propria chiave privata e quindi mandare indietro il numero *random* che era stato generato. Nuovamente, anche in questo caso, chi osserva semplicemente la rete non può in alcun modo desumere quale sia la chiave privata dell'utente e quindi non può in futuro

impersonificarlo.

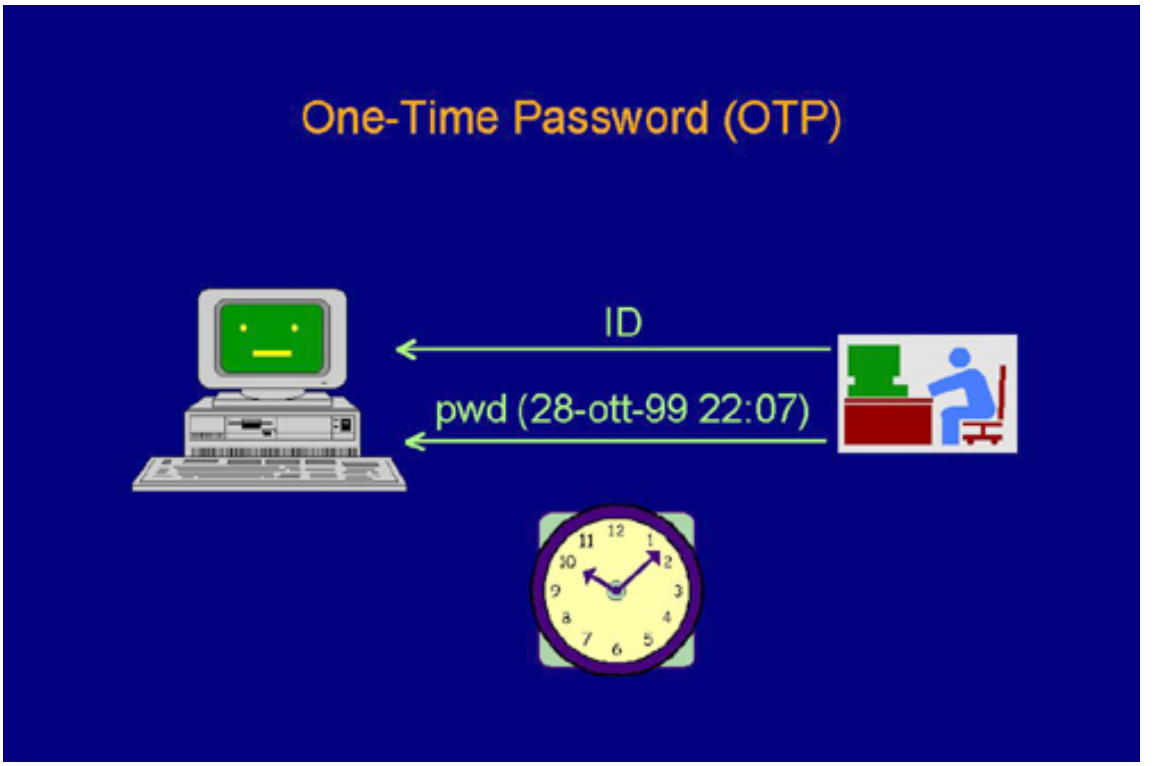
One-Time Password (1/2)



One-Time Password

Un altro modo per evitare che le *password* costituiscano un problema è utilizzare le cosiddette *one-time password*: le *password* valide soltanto una volta. Nello schema più semplice di questo tipo, il sistemista, quando abilita un utente ad accedere ad un sistema, gli fornisce un elenco di *password* numerate. Quando il nostro utente desidera accedere al sistema fornisce il proprio identificativo. Il sistema risponde chiedendo di presentare la *password* di un certo numero, nel nostro esempio la *password* numero 3. A questo punto l'utente preleva dalla propria lista la *password* numero 3 e la comunica in rete. È vero che questa *password* può essere intercettata e letta da chi ha il controllo della rete, ma se il sistema cui ci stiamo collegando non ci chiederà mai più la *password* numero 3, questo non ha nessuna influenza. Questo significa che una volta esaurite tutte quante le *password* che sono state consegnate all'utente, è necessario che l'utente si rechi presso il sistemista per aver un altro elenco di *password*. Questo fa sì che il sistema sia bello e interessante dal punto di vista teorico ma poco pratico a livello implementativo.

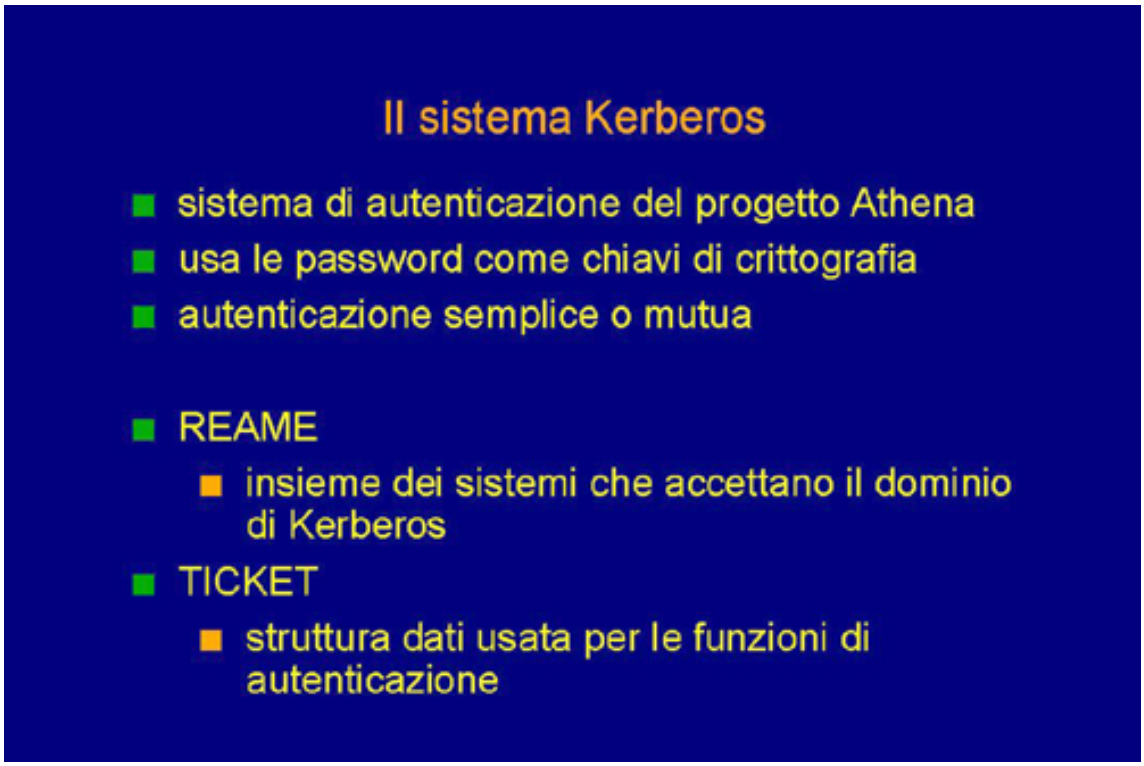
One-Time Password (2/2)



One-Time Password

Per questo motivo ne è stata proposta un'altra versione dove è possibile applicare il medesimo schema di *one-time password*, ossia di *password* non riutilizzabili, *password* usa e getta, basate questa volta non su un elenco di *password* preconfigurato, ma sulla data e ora. In parole povere all'utente viene fornito un programma, o un dispositivo elettronico, che gli permette di inviare in rete non soltanto il proprio identificativo, ma anche la *password* valida per quel preciso istante di tempo, ovviamente con una certa approssimazione. Vuol dire che, ad esempio, esistono dei dispositivi o dei programmi che cambiano automaticamente la *password* ogni minuto, una volta ogni 60 secondi. È chiaro che in uno schema di questo genere l'ora a cui fa riferimento il dispositivo o il calcolatore usato dal nostro utente e il sistema di elaborazione a cui noi stiamo cercando di collegarci, deve essere necessariamente la stessa. Se c'è una differenza di orario tra i due sistemi, ovviamente questo tipo di autenticazione fallirà. Visto che la *password* è relativa unicamente a questo istante di tempo, già un istante di tempo dopo non sarà più valida e quindi non sarà più una *password* che anche se riutilizzata possa dare accesso al sistema.

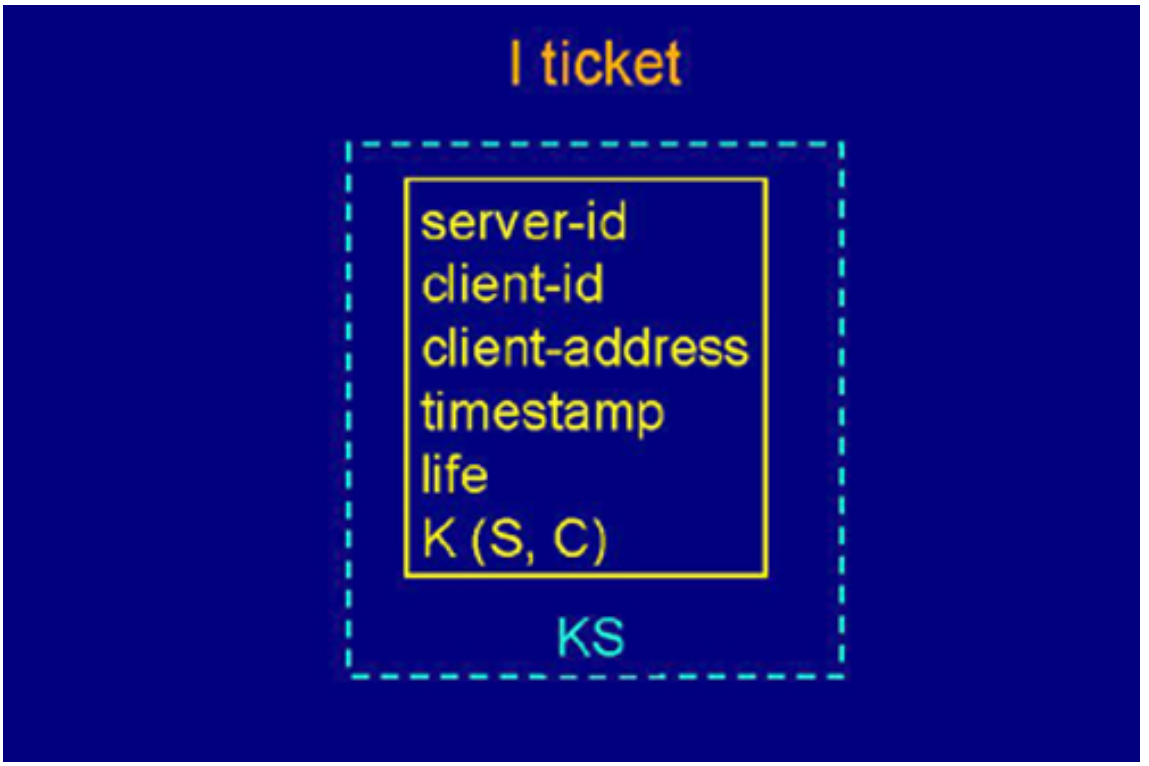
Il sistema Kerberos



Il sistema Kerberos

Infine concludiamo questa breve discussione dei sistemi di autenticazione parlando del sistema *Kerberos*. *Kerberos* è il sistema di autenticazione che è stato sviluppato al prestigioso *MIT* di *Boston*, l'istituto di ingegneria più grosso d'America, all'interno del sistema *Athena*. *Athena* è stato il sistema che il *MIT* di *Boston* ha messo in piedi quando ha voluto rimpiazzare i suoi grossi *mainframe* con una rete di *workstation Unix*. *Kerberos* costituisce il sistema di autenticazione usato in questa rete e si è poi diffuso oggigiorno ed è presente e realizzato su molti sistemi, non soltanto *Unix* ma anche *Windows*. Il concetto base del sistema *Kerberos* e di assegnare agli utenti delle *password* usandole solo ed esclusivamente come chiavi di crittografia. Nella versione originale di *Kerberos* si usavano le *password* come chiavi per il DES. Visto che oggigiorno il DES è un algoritmo che è stato riconosciuto essere debole, si possono usare le *password* come chiavi per algoritmi di crittografia più forti, quali 3DES o IDEA. Il sistema *Kerberos* è in grado di offrirci sia autenticazione semplice, ossia soltanto un utente che dimostra la propria identità al sistema, ma anche mutua autenticazione. Il sistema *Kerberos* si basa su due concetti fondamentali, il primo dei quali è il reame: l'insieme dei sistemi che accettano il dominio di *Kerberos*, ossia che offrono i loro servizi solo a chi si è autenticato tramite *Kerberos*. Il *ticket*, invece, è la struttura dati usata nel sistema *Kerberos* per tutte le funzioni di autenticazione.

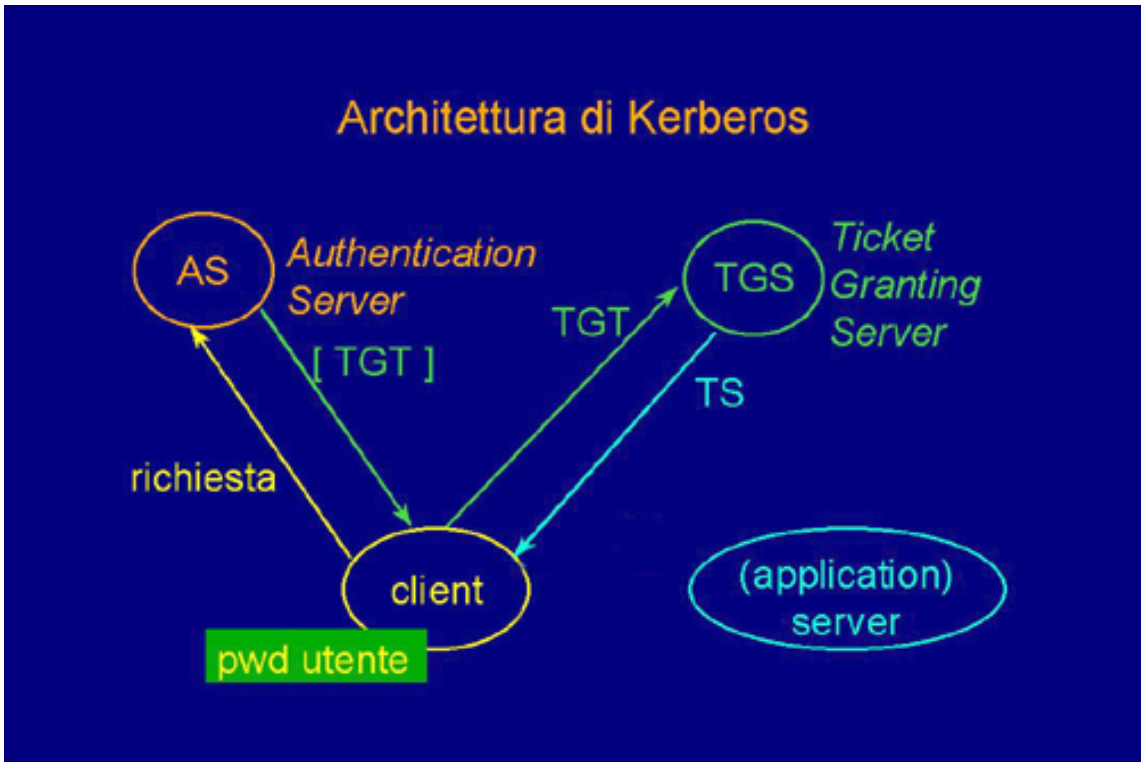
I ticket



I ticket

Osserviamo come è fatto un *ticket*: contiene alcuni dati identificativi, in particolare l'identificazione del *server* a cui è destinato da parte di un certo *client*. Il *client* tipicamente è l'utente che desidera ricevere un servizio da questo *server*. Il *client* è identificato sia in base l'identificativo dell'utente, sia in base all'indirizzo IP, ossia l'utente su una ben determinata postazione di lavoro. Inoltre il *ticket* è valido soltanto per un determinato lasso di tempo: è identificato in base alla sua data e ora di emissione e dalla sua vita, la sua durata. Il *ticket* contiene al suo interno anche una chiave di crittografia simmetrica condivisa fra il *server* e il *client*, questo permette eventualmente al *server* e al *client* di effettuare anche delle comunicazioni crittografate. Per evitare che i *ticket* possano essere manipolati dagli utenti, tutti questi dati vengono presi e cifrati con la chiave KS del *server* a cui sono destinati. Quindi anche quando un utente riceve un *ticket* per usufruire di un certo servizio, l'unica cosa che lui può fare è passare questo *ticket* al *server* da cui desidera avere questo servizio. Non può in alcun modo manipolarlo perché non conosce la chiave per aprire il *ticket*. In pratica è una sorta di biglietto messo dentro una cassaforte blindata.

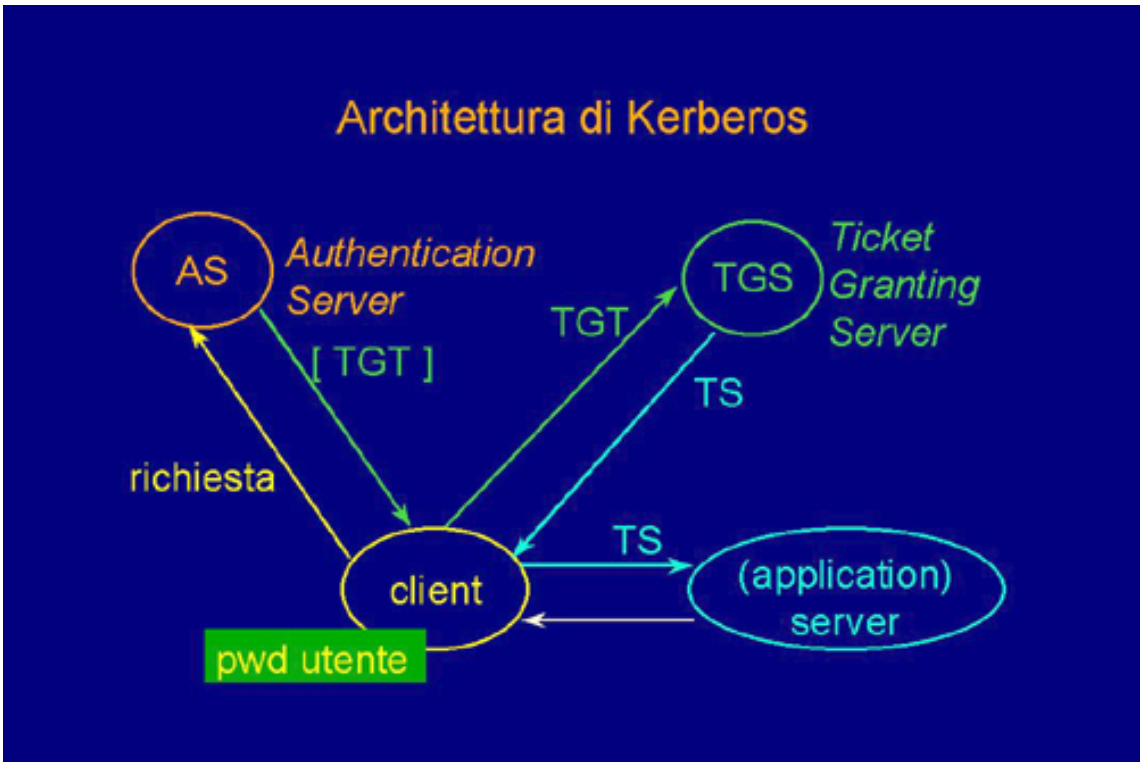
Architettura Kerberos (1/2)



Architettura Kerberos

Questa è l'architettura del sistema *Kerberos*. Il sistema *Kerberos* richiede vari tipi di elementi, in particolare richiede che esitano dei *client*, ossia delle postazioni di lavoro, kerberizzate: dotate di un opportuno *software* di tipo *client*. Quando un utente accede ad una postazione di tipo *client* fornisce localmente la propria *password*, la quale rimane memorizzata all'interno del *client* senza uscirne mai. Il *client* per accedere ad un servizio applicativo deve prima dotarsi del necessario *ticket* di autorizzazione. La procedura segue questo cammino: il *client* invia in chiaro una richiesta al *server* di autenticazione. Il *server* di autenticazione è il cuore di *Kerberos* e in particolare questo *server* emette il cosiddetto TGT, o superbiglietto (TGT vuol dire *Ticket Granting Ticket*), il biglietto che mi da diritto a ricevere gli altri biglietti. Notate che questo biglietto non viene mandato in chiaro, ma viene mandato chiuso dentro una cassaforte che è stata cifrata con la *password* dell'utente. Quindi se ho fatto una richiesta a nome di un'altra persona, non potrò aprire la busta che contiene il TGT, perché non conoscerò la *password*. Se invece sono veramente chi ho detto di essere, allora sarò in grado di aprire la busta e di ottenere il TGT. Il TGT deve successivamente essere presentato ad un altro servizio: il *Ticket Granting Server*, che è il processo demandato ad emettere i biglietti specifici per i servizi applicativi che desideriamo. Quindi, in parole povere, il TGT dimostra la mia identità nei confronti del sistema *Kerberos*, il TGS controllerà se io ho diritto ad accedere ad una certa applicazione, in caso positivo mi emetterà un *ticket* specifico per il servizio S che ho richiesto.

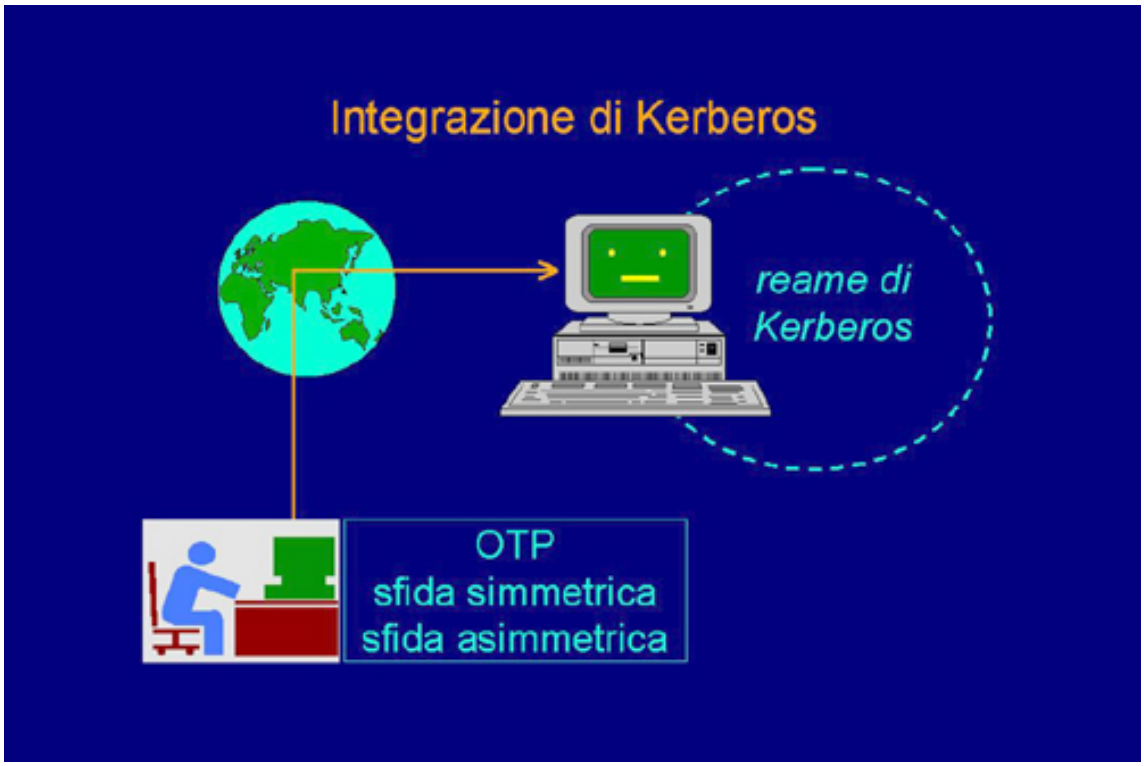
Architettura Kerberos (2/2)



Architettura Kerberos

Alla fine di tutta questa storia quello che farò sarà mostrare il biglietto specifico per questo *server* al servizio applicativo che io desideravo ottenere e il *server* applicativo potrà finalmente erogarmi il servizio. Siccome i *ticket* hanno una certa durata temporale, posso prendere il *ticket* in un certo istante di tempo, ad esempio al mattino quando faccio il primo collegamento al sistema, e poi mantenerlo memorizzato a lungo all'interno del mio *client*. Quindi non è necessario accedere in continuazione all'*Authentication Server* e al TGS, posso avere dei *ticket* di lunga durata, 4 o 8 ore, l'unica avvertenza che devo avere è cancellare i *ticket* dalla mia memoria nel momento in cui io ho finito la mia sessione di lavoro.

Integrazione di Kerberos



Integrazione di Kerberos

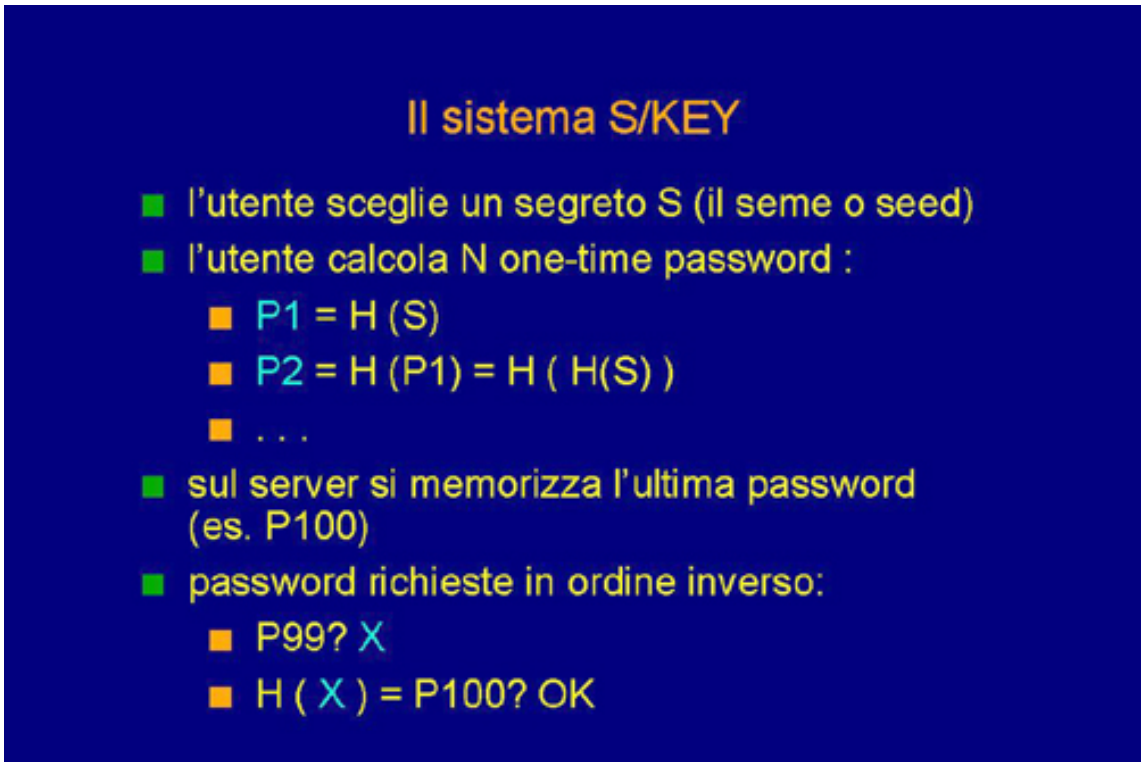
Il sistema *Kerberos* presuppone che la *password* sia inserita localmente in un sistema attaccato al reame di *Kerberos*. Nel caso che io faccia accesso a questo sistema da una postazione remota, ovviamente la mia *password* potrebbe essere intercettata mentre viaggia, supponiamo, da casa mia alla macchina kerberizzata. Allora in questo caso, per risolvere questo problema, *Kerberos* è stato integrato con dei sistemi di autenticazione che non mostrino la *password* durante il suo transito in *Internet*. Esistono delle versioni di *Kerberos* che utilizzano come interfaccia sistemi a *one-time password*, sistemi a sfida simmetrica o sistemi a sfida asimmetrica. In questo modo si estende l'applicabilità di *Kerberos*, che di per sé sarebbe ristretta al solo reame che tipicamente coincide con una rete locale, anche all'accesso remoto fatto via *Internet* tramite reti insicure.

Il concetto di OTP

Il concetto di OTP

È noto che le *password* sono un mezzo insicuro per dimostrare la propria identità ad un sistema di elaborazione a cui si desidera accedere. Nelle precedenti lezioni abbiamo visto che è possibile non utilizzare le *password* per autenticarsi nei confronti di un sistema di elaborazione, usando al loro posto delle tecniche alternative, per esempio le *one-time password* oppure i sistemi a sfida. In questa lezione quello che faremo sarà esaminare alcune tecniche per implementare le *one-time password* o i sistemi a sfida. Cominciamo con la prima di queste. In particolare, ricordo che le *one-time password* sono delle *password* utilizzabili una ed una sola volta; il problema che ci si pone, allora, è come fornire all'utente sempre nuove *password*, perché chiaramente queste verranno esaurite in brevissimo tempo. Ci sono delle soluzioni più artigianali, altre più tecnologiche: entrambe sono tecnicamente valide, quale applicare dipende semplicemente da qual è il tipo di convenienza, economica o organizzativa, verso la quale si punta. In particolare, la soluzione più semplice è quella di fornire all'utente le *password* scrivendogliene tante elencate su un foglietto ed aspettare che il sistema ci richieda una di queste *password*. Una soluzione più tecnologica consiste invece nel dare al nostro utente uno strumento automatico, che effettui il calcolo della *password* necessaria a collegarsi al sistema in un determinato istante di tempo.

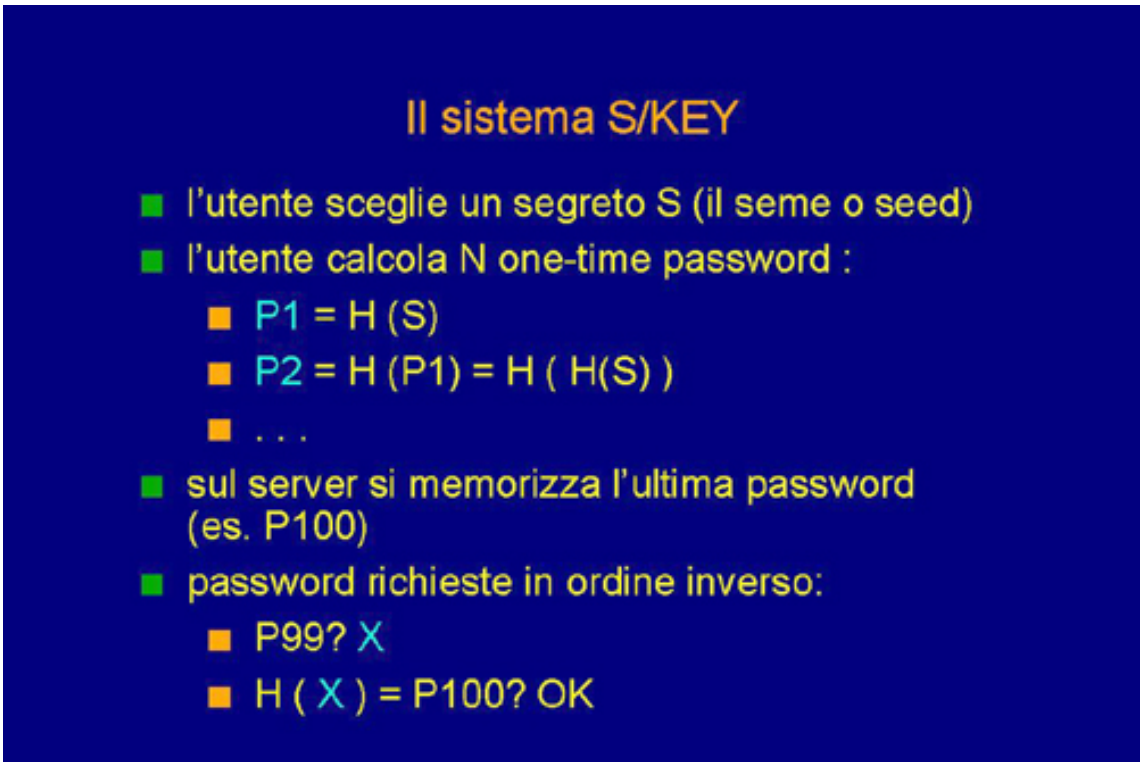
Il sistema S/KEY (1/2)



Il sistema S/KEY

Un'implementazione del primo tipo è quella fatta dal sistema chiamato S/*KEY*. Nel sistema S/*KEY* l'utente sceglie un segreto, il quale non è altro che un numero binario casuale ed è anche chiamato seme o *seed* (in inglese); dopodiché l'utente calcola autonomamente N, numero a piacere, di *password* utilizzabili una volta sola. La modalità con cui le calcola è quella indicata qui: la *password* numero 1 viene calcolata applicando una funzione di *hash* al seme S. La *password* numero 2 è data dalla stessa funzione di *hash* applicata alla *password* numero 1, ossia la funzione di *hash* applicata due volte al segreto S. In questo modo si può generare un numero di *password* lungo a piacere. Per poter poi accedere al *server*, sul *server* deve essere memorizzata l'ultima *password*. Ad esempio, supponendo di averne generate 100, verrà memorizzata sul *server* la *password* numero 100.

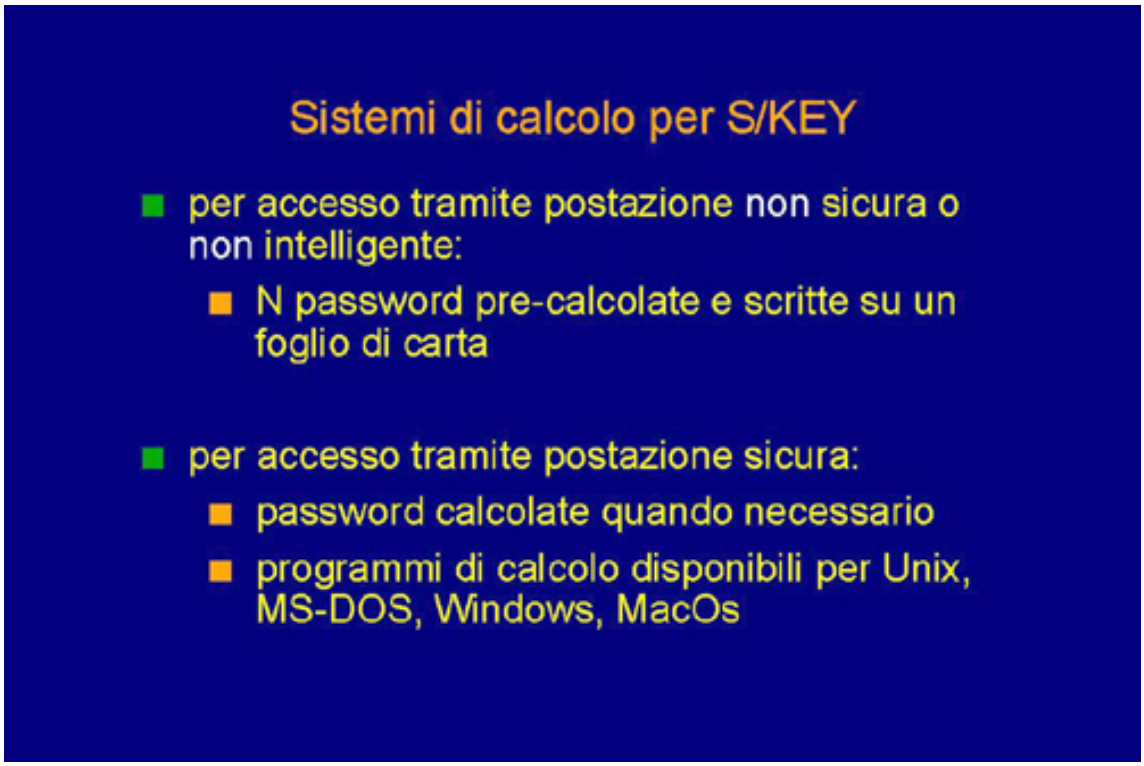
Il sistema S/KEY (2/2)



Il sistema S/KEY

Ogni volta che noi desidereremo fare *login* o accedere a un servizio fornito da questo *server*, lui ci chiederà le *password* in ordine inverso a quello con cui sono state generate. Ad esempio, ci chiederà la *password* numero 99, perché conteneva al suo interno la numero 100. Alla *password* numero 99 risponderemo con un certo numero X. Il *server* può verificare che questo numero X sia effettivamente la *password* numero 99, effettuando il calcolo che vedete indicato: applicando la funzione di *hash* al numero X fornito, si deve ottenere la *password* numero 100. La *password* numero 100 era memorizzata sul *server*, quindi confrontando il risultato di H(X) con la *password* già memorizzata, può decidere se l'utente ha il diritto ad accedere al servizio oppure no. Ovviamente ogni volta che una *password* è stata utilizzata viene buttata via, in questo caso al posto della *password* numero 100 verrà memorizzata la numero 99 e la prossima volta all'utente verrà richiesta la numero 98. Questo è uno schema mosto semplice: una volta che siamo arrivati alla *password* numero 0, ovviamente quello che occorre è che l'utente si generi di nuovo altre 100, altre 1.000, altre 10.000 *password* a suo piacimento e successivamente dia l'ultima di queste al *server* con cui desidera effettuare il collegamento.

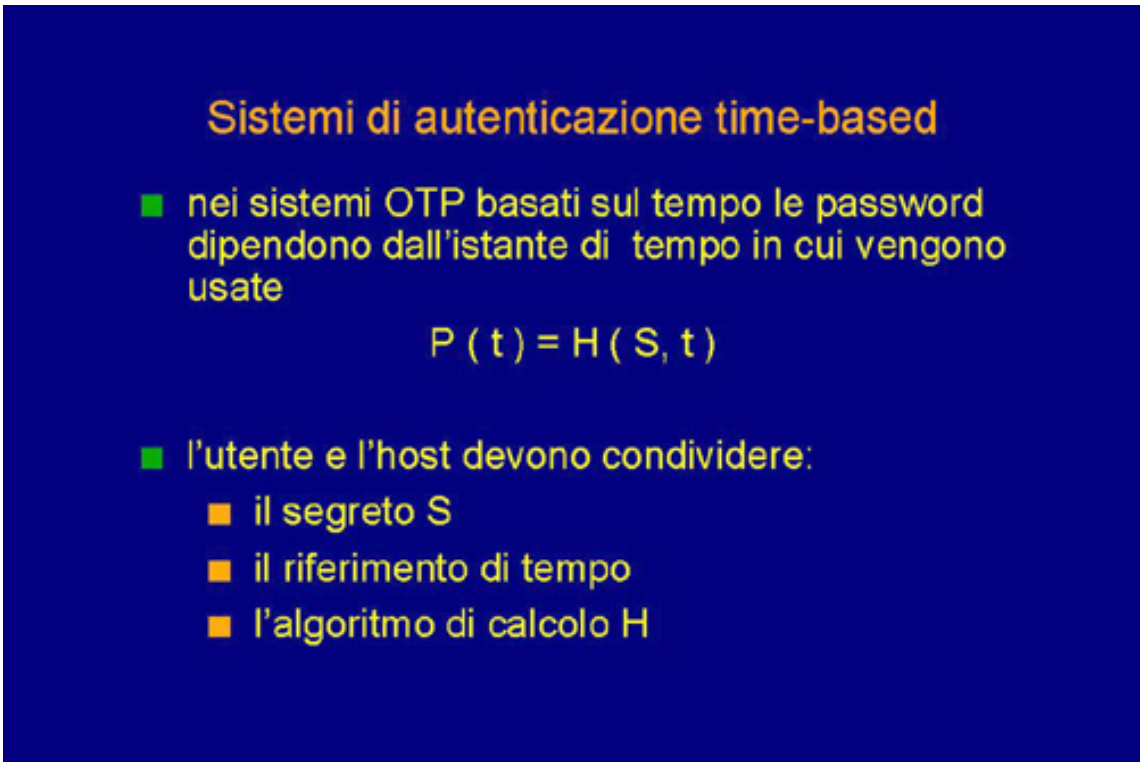
Sistema di calcolo per S/KEY



Sistema di calcolo per S/KEY

Allora, nel caso che noi desideriamo accedere a questo *server* tramite delle postazioni di lavoro non sicure, quindi delle postazioni di lavoro in cui non possiamo esser certi del *software* installato o del *hardware*, oppure vogliamo accedere tramite una postazione non intelligente non in grado di effettuare il calcolo della funzione H, l'unica soluzione è quella di usare queste N *password* precalcolate, ossia di scriverle su un foglio di carta. Questo ci da completa indipendenza dalla postazione di lavoro e dal suo livello di sicurezza. Se invece possiamo accedere al nostro sistema tramite una postazione sicura, quindi una postazione in cui noi ci fidiamo del *software* installato e del *hardware* tramite cui noi interagiamo con questa postazione, allora è possibile non doversi portar dietro questo foglietto ma semplicemente, fornendo il segreto alla postazione di lavoro, far si che sia lei ad effettuare il calcolo. In questo caso le *password* vengono calcolate, quando necessario, da un programma residente sulla postazione sicura. Sono disponibili in rete programmi di calcolo per sistemi *Unix*, ma anche per MS-DOS, *Windows*, *MacOs*. *S/KEY* è un sistema di *public domain* e come tale è stato ampiamente implementato su varie architetture.

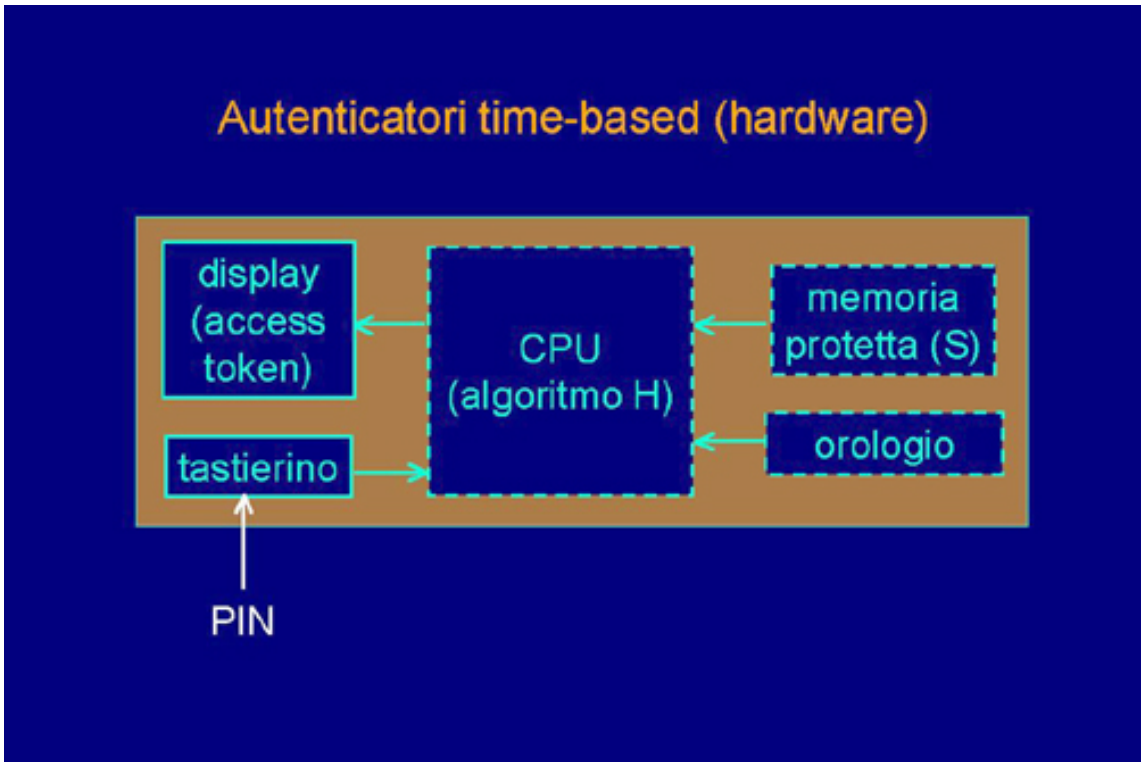
Sistemi di autenticazione time-based



Sistemi di autenticazione time-based

Basandoci sul medesimo principio di avere delle *password* non riutilizzabili più volte, si può scegliere una filosofia diversa: quella di non calcolare una *password* dopo l'altra, ma di calcolare una *password* in funzione dell'istante di tempo in cui essa stessa debba essere utilizzata. Nei sistemi *one-time password* basati sul tempo, le *password* dipendono dal seme iniziale scelto dall'utente, ma anche dall'istante di tempo. Quindi la *password* non è più numerata, ma è la *password* dell'istante di tempo t ed è calcolata applicando una funzione di *hash* al seme e al tempo. In questo caso, per far si che il *server* a cui noi effettuiamo il collegamento possa essere convinto che la *password* che abbiamo fornito sia quella giusta, il *server* e l'utente devono condividere tre cose: il segreto S, il riferimento di tempo e l'algoritmo di calcolo H. In questo caso c'è una richiesta in più rispetto alle *one-time password* precedenti: anche il *server* deve conoscere il segreto S.

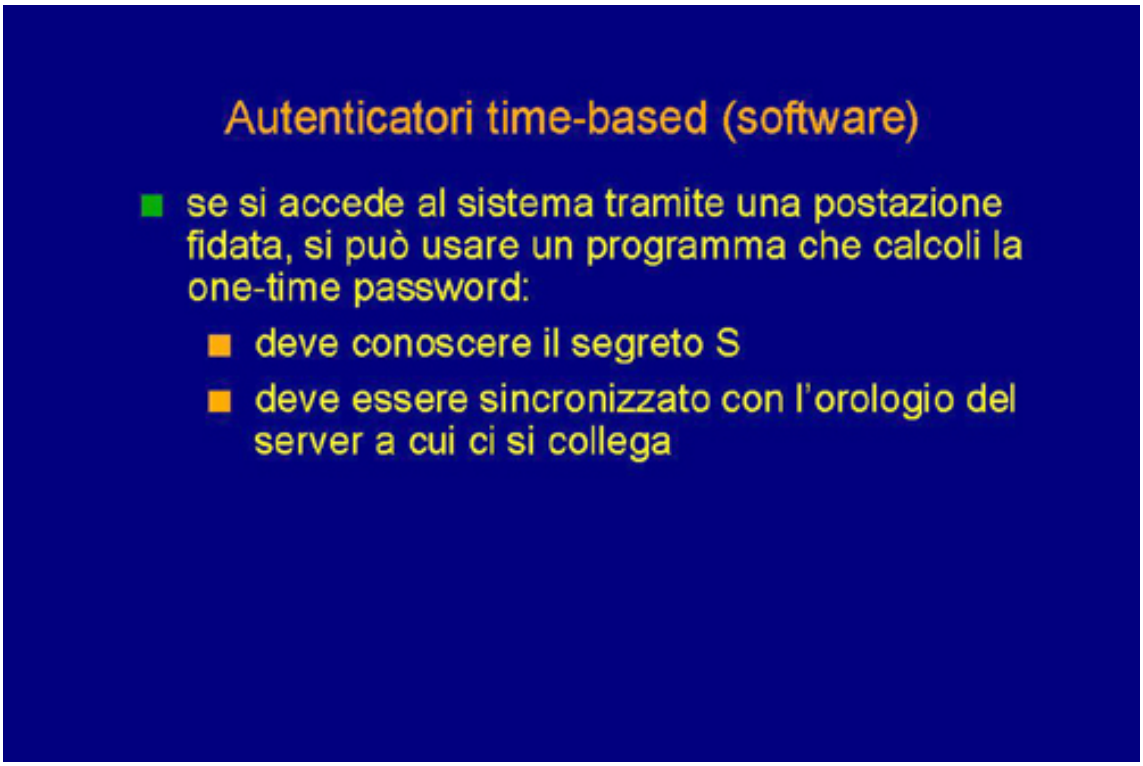
Autenticatori time-based (hardware)



Autenticatori time-based (hardware)

Nel caso in cui vogliamo raggiungere completa indipendenza dalla postazione di lavoro e simultaneamente non avere tra le mani un ingombrante foglietto di carta contenente qualche decina se non centinaia di *password*, alcune ditte commerciali hanno sviluppato i cosiddetti autenticatori *hardware*. Sono delle minicalcolatrici portatili che effettuano per noi il calcolo necessario per il sistema di autenticazione *one-time password*. In questo diagramma vediamo lo schema *hardware* di un autenticatore basato su tempo. Da un punto di vista esterno questo assomiglia a una normalissima carta plastica, formato carta di credito, soltanto leggermente più spessa e dotata tipicamente di un tastierino alfanumerico, o anche solo numerico, su cui l'utente deve battere un *PIN* per abilitare la carta alle sue funzioni. Il *PIN* abilita il calcolo dell'algoritmo H sulla *CPU* presente all'interno di questo autenticatore. La *CPU* prenderà i suoi dati da una memoria protetta, sempre interna alla carta stessa, che contiene il segreto S e da un riferimento temporale interno, quindi un orologio autonomo della carta. Il risultato del calcolo sarà la *one-time password* valida per un certo istante di tempo e sarà visualizzata su un piccolissimo *display* alfanumerico che quindi visualizzerà il cosiddetto *token* di accesso, che ci permetterà di accedere al sistema.

Autenticatori time-based (software)



Autenticatori time-based (software)

Se si accede invece al sistema tramite postazioni fidate, anche per sistemi basati su *one-time password* differenti per i diversi istanti di tempo, è possibile utilizzare una soluzione basata su *software* e non su *hardware*. Le soluzioni su *hardware* di solito sono molto più care di quelle basate su *software*, perché richiedono di avere questi oggetti, che essendo non facilmente manipolabili o forzabili dall'esterno, sono oggetti molto cari. La soluzione *software* si basa sull'utilizzo di un programma che calcoli la *one-time password* sul sistema fidato. Ovviamente a questo programma deve essere fornito il segreto S ed inoltre il PC, la postazione di lavoro su cui il programma esegue, deve avere il suo orologio sincronizzato con quello del *server* a cui ci si collega. Questo è il requisito essenziale affinché entrambi i sistemi calcolino la medesima *password* nel medesimo istante di tempo.

I sistemi a sfida

I sistemi a sfida

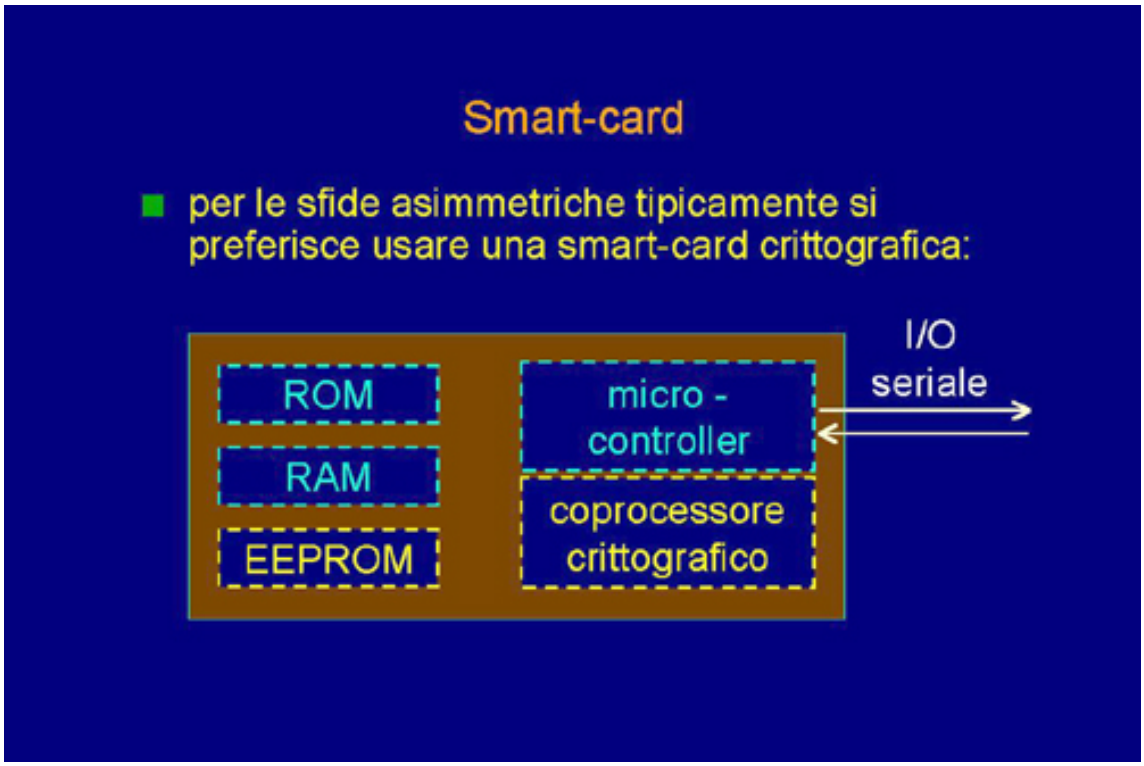
Veniamo ora all'altra categoria di sistemi che possono aiutarci a rimpiazzare le *password* nell'autenticazione nei confronti di un sistema di elaborazione. Questi sono i sistemi a sfida. Nei sistemi a sfida il concetto è che la *password* sia una *password* non riutilizzabile, perché calcolata come risposta a una sfida fornita dal sistema di elaborazione e basata anche su un segreto noto soltanto all'utente e al sistema di elaborazione. Il problema, in questo caso, è come permettere all'utente di conservare, indipendentemente dalla postazione di lavoro, il segreto suo personale e come calcolare la funzione F, quantomeno il risultato della funzione F. Una prima soluzione non ci rende indipendenti dalla postazione di lavoro, ossia dice: il segreto viene memorizzato utilizzando il disco del mio *personal computer*, supposto sicuro, e la funzione F viene calcolata utilizzando la *CPU* del mio sistema di elaborazione. Questa è una soluzione valida, ma lega l'accesso alla postazione di lavoro. Una soluzione migliore e più indipendente è quella che utilizza un dispositivo autonomo di memoria e calcolo. Quindi, qualcosa di simile agli autenticatori *hardware* che abbiamo visto un attimo fa per le *one-time password* basate sul tempo.

Calcolatori per sfide simmetriche

Calcolatori per sfide simmetriche

Normalmente, per i sistemi a sfida si utilizzano quelle che si chiamano le criptocalcolatrici. I sistemi a sfida possono basarsi su algoritmi di crittografia simmetrici o asimmetrici. Per i sistemi basati su sfida simmetrica si usano delle criptocalcolatrici tipicamente basate sull'algoritmo DES. Quindi, in questo caso, il segreto corrisponde con la chiave di cifratura, ad esempio la chiave di cifratura DES, e la funzione F è l'algoritmo di cifratura simmetrico presente a bordo della carta. L'autenticatore *hardware*, in questo caso, contiene al suo interno la chiave di cifratura, implementa l'algoritmo di cifratura stesso, aspetta l'introduzione della sfida fornita dal sistema all'utente e calcola poi la risposta. È compito dell'utente battere la risposta che è stata ottenuta sulla tastiera, per fornirla al sistema che ci ha sfidato.

Smart-card

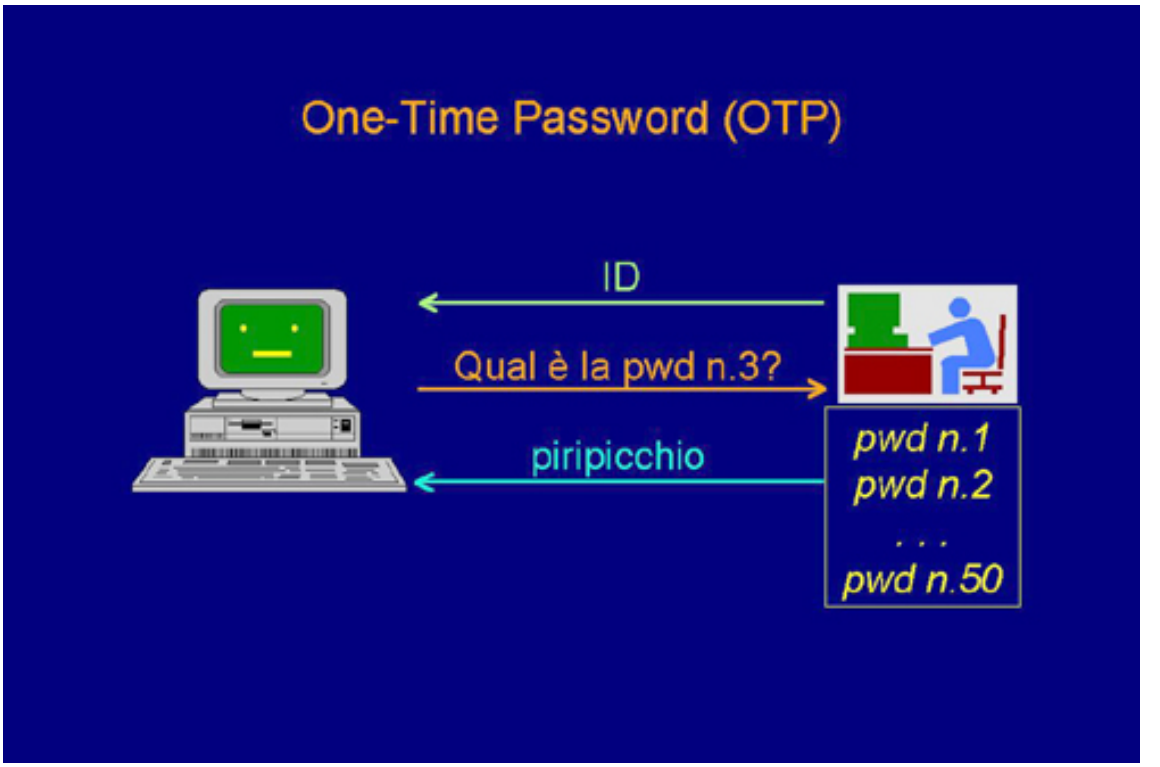


Smart-card

Nel caso di sfide asimmetriche, al posto che utilizzare una criptocalcolatrice vera e propria si utilizzano dei dispositivi particolari chiamati *smart-card* (che significa carta intelligente), che non sono altro che criptocalcolatrici normalmente di tipo asimmetrico. Possono esistere anche atri tipi di *smart-card*: quelle che sono comunemente utilizzate per darci i premi nei concorsi, ad esempio per chi acquista più benzina, o per chi fa più spesa al supermercato, sono delle normalissime carte dotate soltanto di un *chip* di memoria. Nel caso invece di applicazioni alla sfida asimmetrica, la *smart-card* è dotata di un *chip* in grado di effettuare i calcoli crittografici asimmetrici. Questo è lo schema di una *smart-card* utilizzabile per sfide asimmetriche. La *smart-card* è racchiusa in un contenitore plastico, che vedremo tra poco, e contiene al suo interno un *microcontroller*. Un *microcontroller* non è nient'altro che una piccola *CPU* in grado di svolgere varie funzionalità, tra cui anche semplici funzioni di *input/output* di tipo seriale. Normalmente le *smart-card* colloquiano con il calcolatore a cui sono collegate tramite un dispositivo di *input/output* seriale a 9.600 b/s. La *smart-card* contiene anche un po' di ROM, contenente il programma di controllo della *smart-card*, e un po' di RAM, per permettere alla *CPU microcontroller* di effettuare alcuni semplici calcoli. Affinché una *smart-card* sia utilizzabile per risolvere delle sfide asimmetriche deve essere dotata di due cose addizionali: in particolare deve essere dotata di un coprocessore crittografico. Il coprocessore crittografico è del *hardware* aggiuntivo che realizza la funzione di crittografia asimmetrica, tipicamente la funzione RSA, talvolta anche la funzione DSA. Inoltre, è presente all'interno della *smart-card* della memoria aggiuntiva, chiamata EEPROM o E2PROM. Si tratta di una memoria di sola lettura che però è anche scrivibile tramite funzioni elettriche. Questa a tutti gli effetti agisce come un piccolo *hard disk*, come un piccolo disco magnetico, pur trattandosi di memoria puramente elettronica. Ci permette di memorizzare dei dati sulla *smart-card* in modo permanente, ossia dei dati che rimangono memorizzati anche quando la *smart-card* non viene

collegata al *computer* e quindi non è alimentata elettricamente.

Standard per le smart-card



Standard per le smart-card

Ovviamente, per permettere alle *smart-card* di essere utilizzate su tanti sistemi diversi e da tante applicazioni diverse (cosa che oggi è molto richiesta, perché abbiamo visto che le funzioni asimmetriche sono valide non soltanto per l'autenticazione a sfida, ma anche per la firma digitale), anche per la sicurezza delle reti sono stati definiti una serie di standard. In particolare, gli standard definiti dall'ISO, raggruppati sotto il numero 7816, definiscono il formato fisico, elettrico e logico, diciamo di basso livello, delle *smart-card*. Nel caso che però la *smart-card* effettivamente sia in grado di effettuare delle operazioni crittografiche, è desiderabile anche avere un'interfaccia verso queste funzioni. Quest'interfaccia normalmente è chiamata interfaccia di tipo CSP (*Crypto Service Provider* = fornitori di servizi crittografici). I servizi crittografici che fornisce sono due: il calcolo di alcuni algoritmi di crittografia e un *database* sicuro per le chiavi private dell'utente. Non c'è accordo su quale deve essere l'interfaccia CSP standard da utilizzare. Attualmente esistono due standard in competizione fra di loro e con funzionalità simili. Lo standard PKCS-11 è l'interfaccia CSP utilizzata da alcuni sistemi, tipo i *browser Netscape*, o i sistemi di *workflow* forniti da *Lotus*. L'interfaccia CAPI (*Crypto* API) è invece l'interfaccia CSP fornita standard con i sistemi *Microsoft*. Sono interfacce equivalenti ma purtroppo incompatibili. Lo standard PC/SC è invece lo standard de facto per l'interfacciamento dei lettori di *smart-card*, ossia per permettere ad una *smart-card* di essere utilizzata su lettori di tipo diverso.

Smart Card

Smart Card

Questa è una *smart-card* grezza; come vedete si tratta semplicemente di un rettangolo di plastica su cui è stato incastrato a forza un *chip* sottilissimo. Questa è una *smart-card* crittografica in grado di contenere 8 *Kbyte* di memoria e in grado di svolgere funzioni di crittografia asimmetrica di tipo RSA. Queste *smart-card* cominciano ad essere utilizzate in vari progetti, anche a livello applicativo. Ad esempio, ho portato qui con me una delle *smart-card* utilizzate nel progetto *DISTINCT*. È un progetto della Comunità Economica Europea che coinvolge varie città attraverso l'Europa, in particolare partecipa anche la città di Torino, con il comune, gli uffici anagrafici e il politecnico di Torino stesso. Come vedete le carte possono essere personalizzate. Questo è il bozzetto della carta che si vorrebbe dare ai cittadini di Torino per permettere di usufruire delle funzionalità anagrafiche, di trasporti e così via.

Sistemi biometrici

Sistemi biometrici

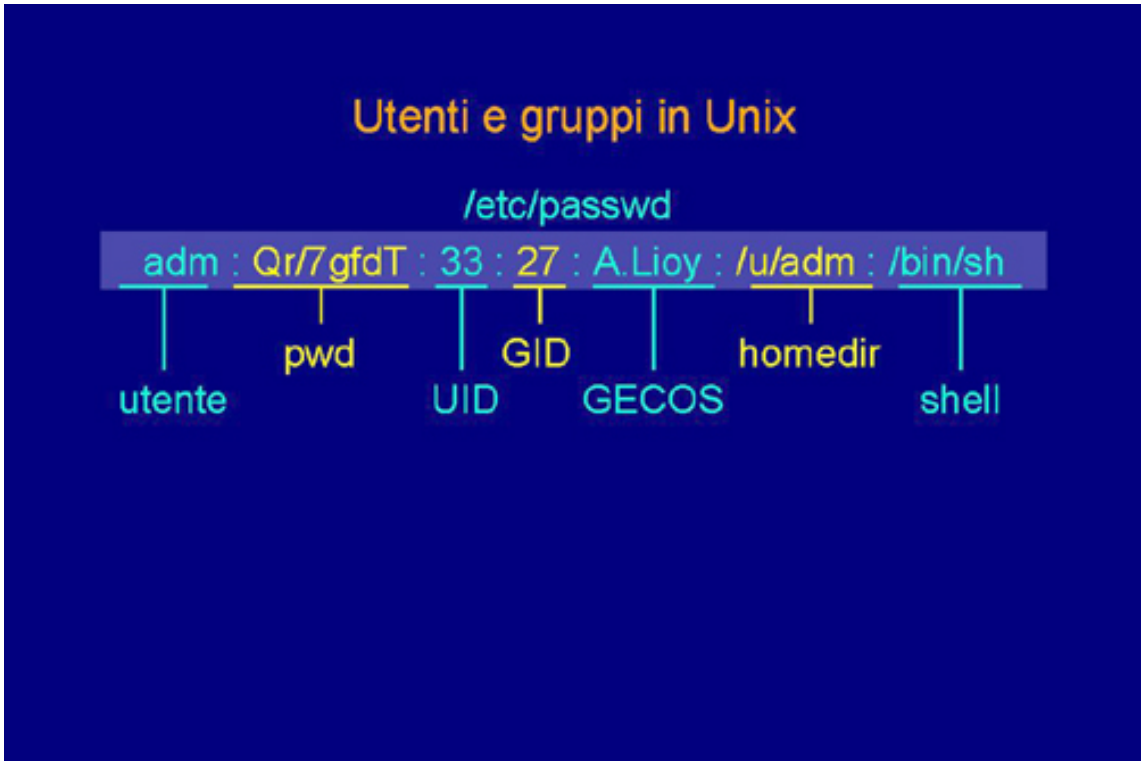
Per evitare il problema che una *smart-card*, o uno di questi autenticatori *hardware*, venga rubato al legittimo titolare e poi utilizzato per accedere ad un sistema, è stato proposto di utilizzare questi dispositivi *hardware* in congiunzione con sistemi di identificazione personale di tipo biometrico. In pratica il sistema biometrico serve a sostituire la *password*, o il *PIN*, che normalmente deve essere utilizzata per abilitare le funzioni della carta. Si utilizzano basilarmente tre tipi di tecniche. Esistono dei dispositivi in grado di abilitare l'utilizzo della carta solo in base al riconoscimento di una determinata impronta digitale, oppure del tono vocale dell'utente o, se dotato di una piccola telecamera oculare, effettuando lo *scanner* dei vasi sanguigni presenti sul fondo della nostra retina. Tutte queste tecniche sono ugualmente valide in teoria, in realtà l'implementazione pratica non ha ancora raggiungo un grado di affidabilità sufficiente in qualunque ambiente. Il problema che ci troviamo ancora a fronteggiare è quello delle cosiddette accettazioni o rifiuti di collegamento falsi. Un falso rifiuto si ha quando l'utente viene rifiutato, cioè non gli viene dato il permesso di accedere, nonostante lui sia veramente chi dice di essere: basta ad esempio aver posto male il dito sul lettore di impronte digitali e questo potrebbe capitare. Più pericoloso ancora il caso contrario, in cui nonostante un utente non autorizzato sia dotato di un impronta digitale diversa da quella autorizzata, mettendo il dito sul vetrino gli viene concesso l'accesso. Ovviamente più il sistema è sofisticato più crescerà il suo prezzo, ma questo ovviamente tenderà anche a diminuirne l'accettazione in vari ambiti applicativi. Sistemi biometrici in unione con dispositivi di accesso *hardware* sono però sicuramente una delle strade maestre che si diffonderanno largamente nell'utilizzo dei sistemi informativi nei prossimi anni.

Sicurezza nei sistemi Unix

Franco Callegati

9.1.1 (Implementare appropriate procedure per garantire la sicurezza di una rete) - 9.1.4 (Assegnare agli utenti i diritti appropriati per accesso a file, applicazioni e risorse) - 9.1.5 (Usare un sistema di account su una rete)

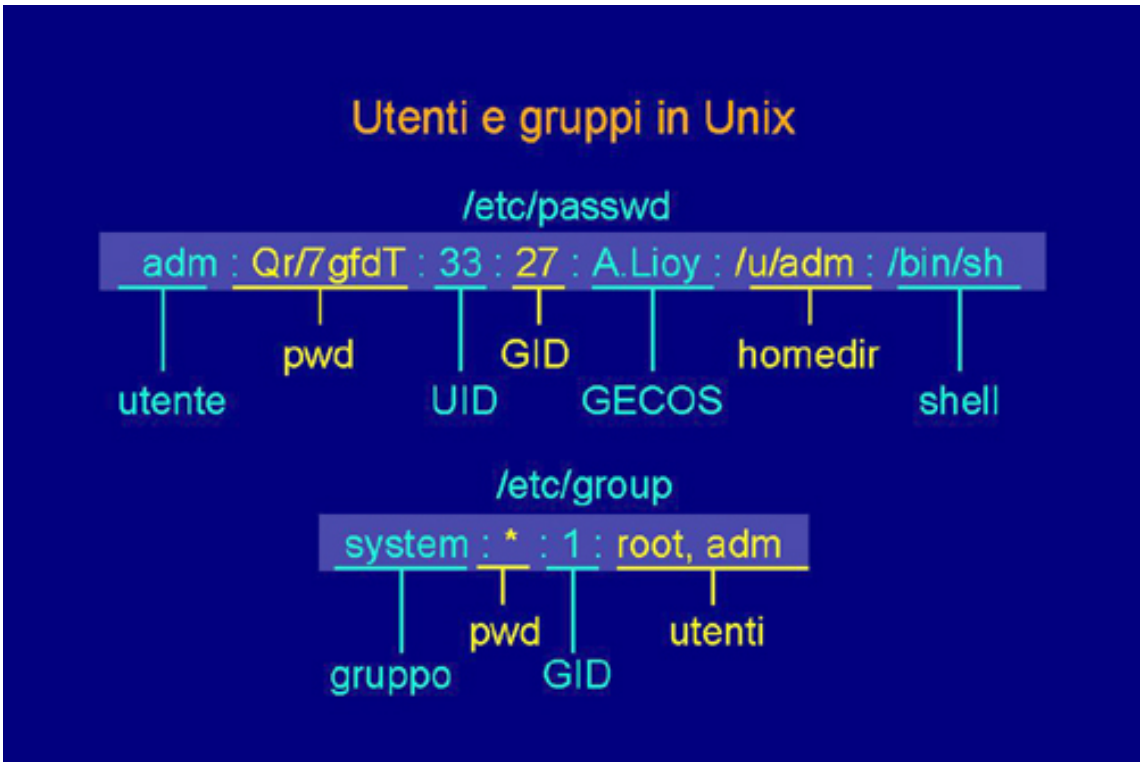
Utenti e gruppi in Unix (1/2)



Utenti e gruppi in Unix

In questa lezione tratteremo di alcuni concetti di sicurezza tipici dei sistemi *Unix*. In particolare, questi sistemi definiscono gli utenti abilitati ad utilizzare il sistema in un *file* chiamato *passwd*, che si trova all'interno del direttorio etc ed è un *file* di testo organizzato a righe. In ogni riga di questo *file* ci sono vari campi separati da :. Il primo campo, quello che qui contiene il valore adm è il nome dell'utente, diciamo il nome mnemonico dell'utente. Il secondo campo, che qui contiene il valore Qr/7gfdT, contiene la *password* di quell'utente; il fatto che ci sia questa scritta non significa che la *password* dell'utente sia realmente Qr/7gfdT, ma vuol dire che la *password* crittografata dell'utente da come risultato quello qui indicato. Il terzo campo indica la UID, ossia l'identificatore univoco, il numero che è stato assegnato a questo utente. In realtà il sistema *Unix* lavora identificando utenti ed oggetti tramite questi numeri, quindi la scritta adm, il nome mnemonico, è solo un aiuto per l'utente per dover evitare di ricordarsi sempre i numeri univoci. Analogamente, il quarto campo contiene un altro numero, che è il cosiddetto identificatore di gruppo: ossia questo utente appartiene primariamente al gruppo numero 27. Vedremo tra breve dove esistono dei nomi mnemonici associati a questi gruppi numerici.

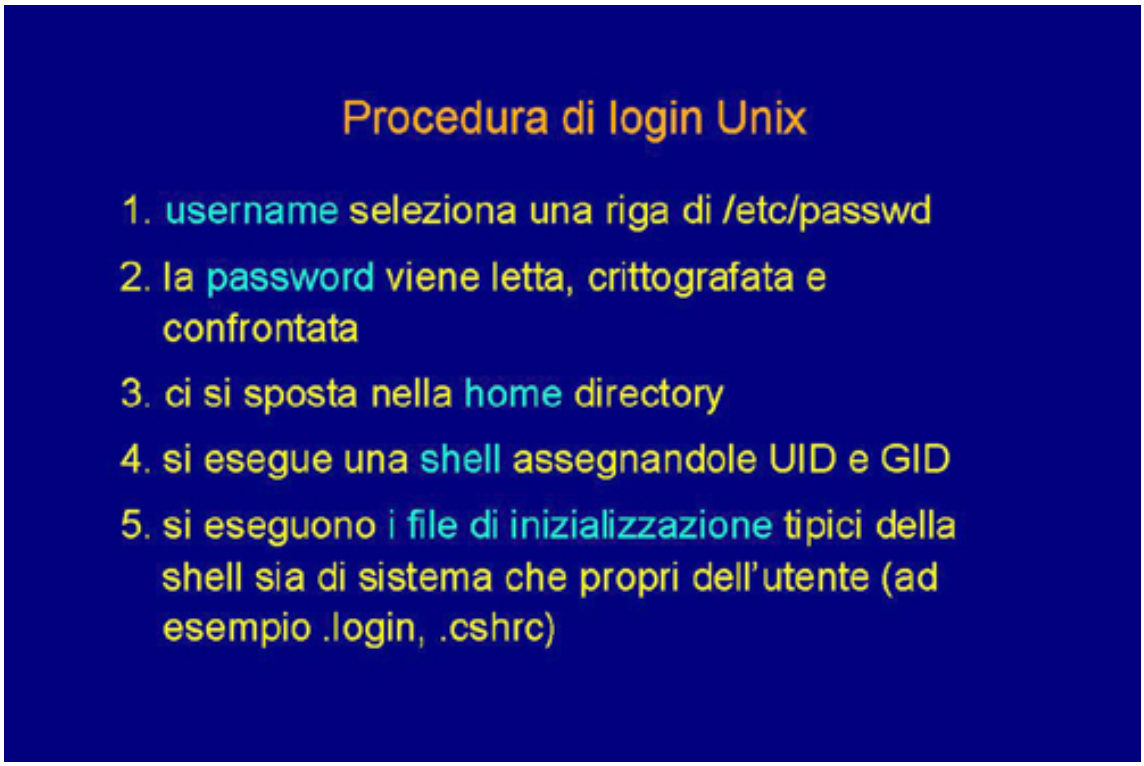
Utenti e gruppi in Unix (2/2)



Utenti e gruppi in Unix

Il quinto campo contiene delle informazioni ausiliarie, è detto campo GECOS e prende il nome da un antico sistema *Unix* in cui questo campo aveva una importanza particolare. Oggigiorno viene normalmente utilizzato per mettere informazioni ausiliarie non trattate automaticamente dal sistema ma di aiuto per il sistemista. Il penultimo campo contiene la cosiddetta *home directory*, ossia il punto del *file system* di *Unix* in cui sono memorizzati i *file* di questo utente. In questo caso particolare si tratta del direttorio adm all'interno del direttorio u. Infine, l'ultimo campo contiene la cosiddetta *shell*, ossia il programma che deve essere automaticamente attivato dopo la procedura di *login* di questo utente. Il sistema *Unix* utilizza anche un altro *file*, chiamato *group*, sempre nel direttorio etc, che serve a fornire la corrispondenza fra identificativi numerici di gruppo e nome mnemonico. Il primo campo contiene, appunto, il nome mnemonico di un gruppo. Il secondo campo contiene una *password*, che però normalmente in tutti gli attuali sistemi *Unix* contiene il campo *, nel senso che le *password* da lungo tempo non vengono più utilizzate per gli accessi ai gruppi. Il terzo campo contiene l'identificativo numerico del gruppo ed il quarto campo contiene l'elenco di tutti gli utenti che appartengono al gruppo in questione. Ovviamente anche questo gruppo è costituito da tante righe, una per ogni gruppo che è stato definito nel sistema. Questi due *file* rappresentano un problema se non gestiti correttamente, perché per il corretto funzionamento di *Unix* questi *file* devono essere leggibili da tutti quanti gli utenti.

Procedura di login Unix



Procedura di login Unix

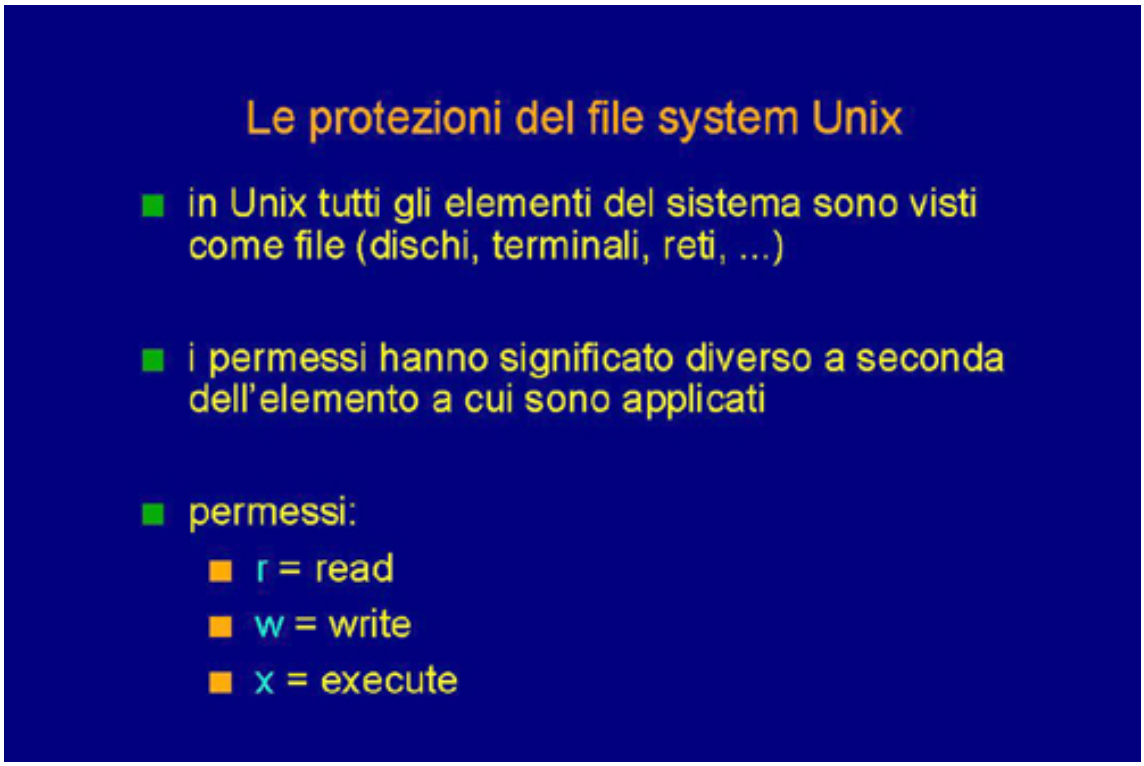
In particolare, questi *file* vengono letti durante la procedura di *login* al sistema. La procedura di *login* segue i seguenti passi: avendo l'utente fornito uno *username* e una *password*, per accedere al sistema tramite lo *username* indicato, la procedura di *login* seleziona una riga del *file* /etc/passwd, quella corrispondente a questo utente. Successivamente la *password* viene letta, crittografata e confrontata con la *password* presente nel secondo campo della riga di /etc/passwd selezionata. Se il confronto della *password* ha avuto successo allora il *login* è permesso e il sistema si sposta nella *home directory*, ossia nel punto dove sono memorizzati i *file* dell'utente. A partire da questa *home directory* viene attivata una *shell*, ossia il programma che deve essere attivato automaticamente al *login* e questa *shell* viene eseguita, non con i permessi di sistema, ma con i permessi che derivano dagli identificativi univoci di gruppo e di utente. Infine, ogni *shell* può avere uno o più *file* di inizializzazione, che sono tipici di questa *shell*. Questi possono essere *file* di inizializzazione di sistema oppure personali dell'utente e servono, in qualche modo, a personalizzare il funzionamento della *shell*.

La shadow password

La shadow password

Se qualcuno è in grado di andare a leggere le *password*, come abbiamo già detto in passato, può provare a montare un attacco di tipo dizionario, ossia può cercare di indovinare la *password*. Per evitare questo problema, nel sistema *Unix* è stato sviluppato il concetto di *shadow password*, o *password* ombra, o *password* nascoste, come preferite. In parole povere si tratta semplicemente di questa cosa: dal *file* /etc/passwd vengono rimosse le *password* crittografate, che vengono invece salvate in un altro *file* accessibile soltanto al sistema. In questo modo si permette agli utenti di leggere /etc/passwd per effettuare le normali corrispondenze, ad esempio fra UID e nome mnemonico dell'utente, ma si toglie il permesso di leggere le *password* crittografate e quindi di montare attacchi di tipo dizionario. Come vedete, in questo caso se utilizziamo le *shadow password* quello che capita è che nel *file* /etc/passwd il secondo campo contiene un asterisco, ossia un indicazione che per questo utente la *password* non si trova all'interno di questo *file*. In realtà la *password* viene memorizzata dentro il *file* /etc/shadow, ossia il *file shadow* nel direttorio etc. Questo *file* è anch'esso organizzato a righe, per ognuna delle quali c'è nome mnemonico dell'utente e la sua *password* crittografata. La differenza è che il *file* /etc/passwd è leggibile da qualunque utente del sistema, mentre il *file* /etc/shadow richiede i cosiddetti privilegi di super-utente, di cui parleremo fra poco.

La protezione del file system Unix



La protezione del file system Unix

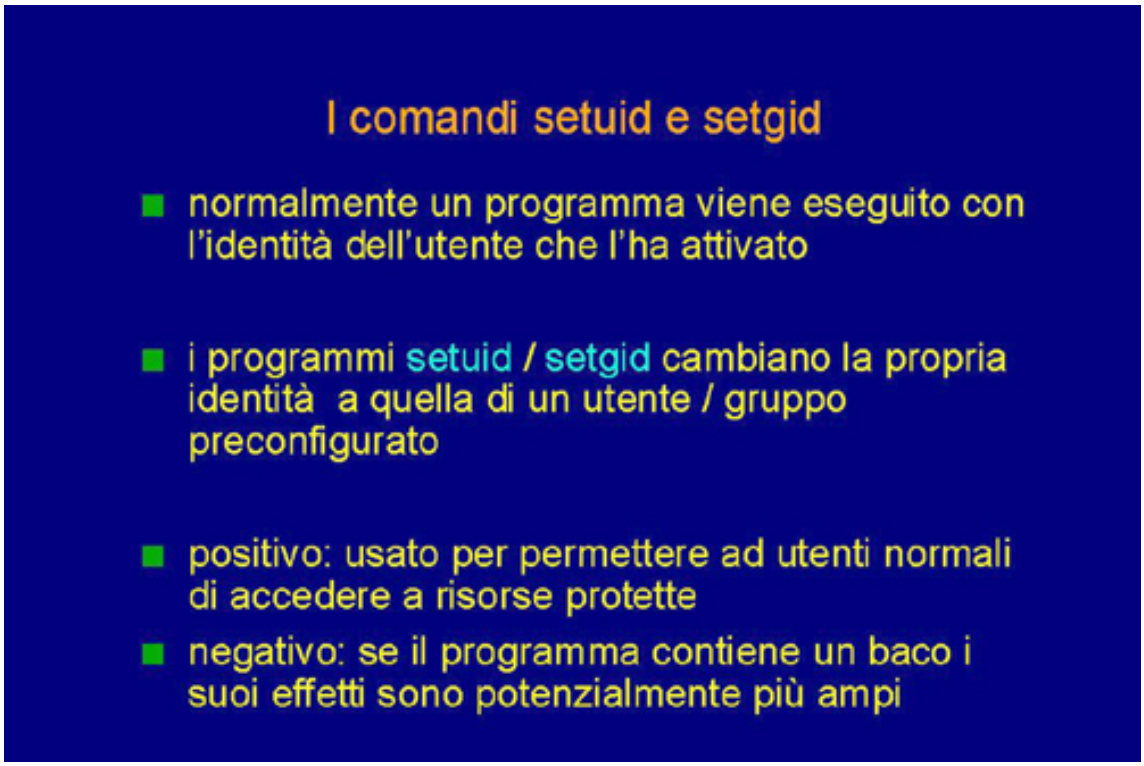
In generale, all'interno del sistema *Unix* è stata fatta una scelta semplificativa, quella di trattare tutti quanti gli oggetti del sistema come *file*. Ossia, sia che stiamo parlando veramente di un disco organizzato a *file*, sia che stiamo parlando di un generico *driver* di un dispositivo (una scheda di rete, un terminale, una linea seriale, una linea di comunicazione), tutti quanti questi oggetti vengono rappresentati come *file*. Questo da una parte semplifica la gestione, dall'altra complica un po' il controllo dei permessi di accesso a questi oggetti. I permessi quindi hanno significato diverso a secondo dell'elemento a cui sono applicati. Esistono tre permessi base in *Unix*: i permessi di lettura, di scrittura e di esecuzione. Questi tre permessi hanno un significato abbastanza chiaro nel caso che stiamo parlando di *file*, ma nel caso in cui stiamo parlando, ad esempio, di un direttorio, ovviamente il permesso di esecuzione non avrebbe di per sé senso, perché un direttorio non è un programma eseguibile. Allora, in questo esempio, avere il permesso di esecuzione su un direttorio significa avere il permesso di entrare o di attraversare questo direttorio durante uno spostamento nel *file system*. Analogamente, non esiste un'operazione di permesso di cancellazione, ma viene ottenuto tramite il permesso di scrittura nel direttorio, perché a tutti gli effetti cancellare un *file* significa rimuovere questo *file* dal direttorio corrente. Dopo un po' ci si fa l'abitudine, ma è chiaro che non è una cosa facile da imparare e da capire. Questo può portare a degli errori di configurazione che possono essere sfruttati, da persone malintenzionate, per manipolare impropriamente *file* del nostro sistema.

I permessi

I permessi

In generale, i permessi sono attribuiti agli oggetti e sono specificati in modo diverso per tre identità che normalmente, colloquialmente, sono chiamati UGO. In realtà UGO è l'acronimo di *User, Group e Others*: indica che i permessi vengono specificati per il proprietario dell'oggetto, per il gruppo a cui l'oggetto appartiene, ossia in pratica il gruppo degli amici del proprietario, e infine sono specificati degli altri permessi anche per *others*, ossia per il resto del mondo, tutti gli altri utenti che non sono né il proprietario né il gruppo. Notate che nel caso ipotetico, difficile da realizzarsi ma teoricamente possibile, in cui l'utente ha meno diritti del resto del mondo, si applicano i permessi più restrittivi, quelli di utente, perché non appena si fa un *match* con una di queste tre *entry* si applicano i permessi relativi. Bisogna notare che in *Unix* esiste un utente che può fare tutto senza rispettare alcun tipo di permesso, questo è il cosiddetto super-utente. Il super-utente è chiunque abbia UID pari a zero. Normalmente UID = 0 viene data esclusivamente all'utente chiamato *root*. Una operazione tipica degli *hacker* che riescono a manipolare il *file* /etc/passwd, è quella di assegnare UID = 0 anche ad altri utenti che apparentemente non hanno permessi speciali, perché non hanno il nome di *root*. Visto che quel che conta non è il nome ma la UID, se questi utenti hanno UID uguale a zero, hanno a tutti gli effetti il permesso di fare qualunque cosa all'interno del nostro sistema. Quindi, prestare particolare attenzione al fatto che nel *file* /etc/passwd esista un'unica riga con UID pari a zero e questa riga deve corrispondere con l'utente *root*.

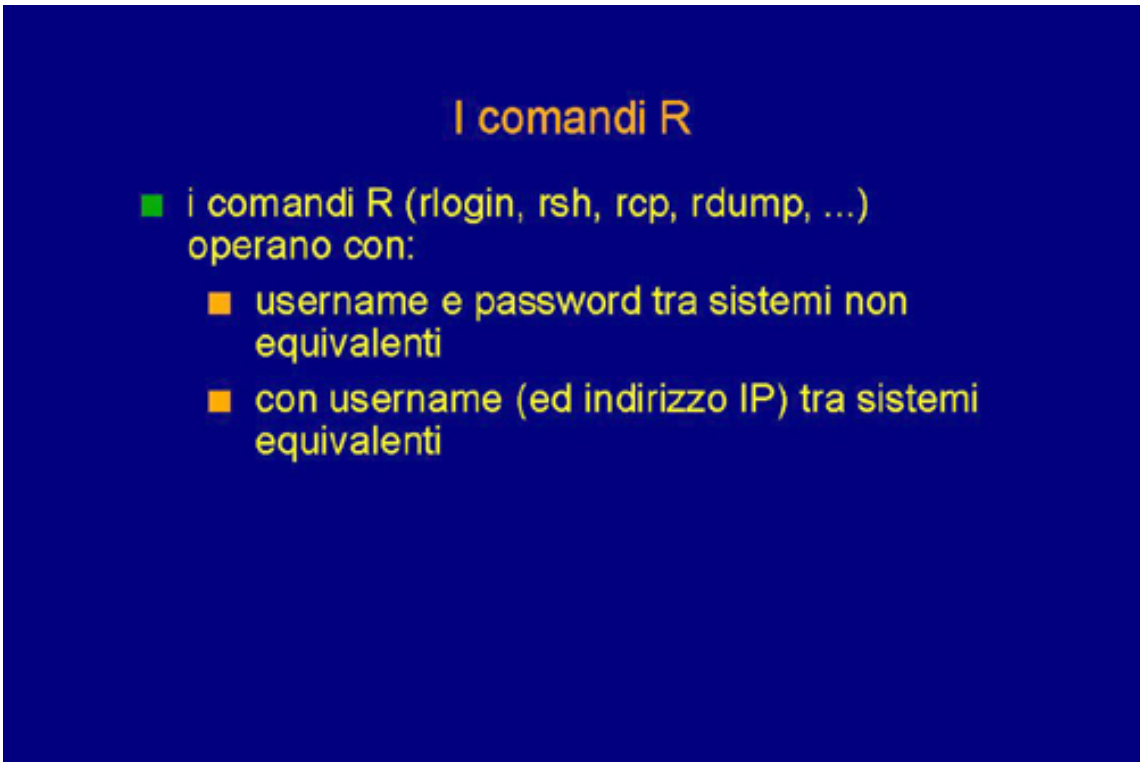
I comandi setuid e setgid



I comandi setuid e setgid

Normalmente, quando un programma viene eseguito da un utente, viene eseguito con i permessi relativi a quel utente. Questo impedirebbe normalmente agli utenti di accedere ed utilizzare delle risorse che richiedono invece i permessi di sistema. Ad esempio, nel caso di un calcolatore multi-utente, i dispositivi di collegamento, come la scheda di rete, devono essere gestiti dal sistema per risolvere i conflitti e ciò vorrebbe dire che nessun utente potrebbe accedere direttamente alla scheda di rete o a dispositivi analoghi, ad esempio pensiamo ad una stampante condivisa. Per risolvere questo problema, *Unix* ha introdotto i comandi *setuid* e *setgid*. Questo significa che al posto di essere eseguiti con i privilegi dell'utente che ha attivato il programma, *setuid* e *setgid* sono in grado di cambiare la propria identità a quella di un utente o di un gruppo, nel caso di programma *setgid* preconfigurato. Tipicamente questo viene usato in modo positivo per permettere ad utenti nomali di accedere a risorse protette. Ad esempio, possiamo dare a più utenti il permesso di controllare lo stato della stampante facendo un comando *setuid root*, ossia un comando che solo per l'esecuzione di quel particolare programma permette di accedere alla stampante, pur non essendo sistemista. Ma i programmi *setuid* o *setgid* possono anche avere un effetto negativo. Ad esempio, se il programma contiene un baco ovviamente i suoi effetti sono potenzialmente molto più ampi se il programma viene eseguito come *root*, quindi con i massimi privilegi di sistema, invece che con i privilegi tipici di un solo utente. Quindi, questa è un'altra cosa a cui bisogna prestare attenzione: bisogna assicurarsi che i programmi *setuid* e *setgid* siano solo quelli definiti dal sistema e che non ne siano stati introdotti degli altri. Inoltre è necessario correggere rapidamente eventuali bachi presenti all'interno di questi programmi.

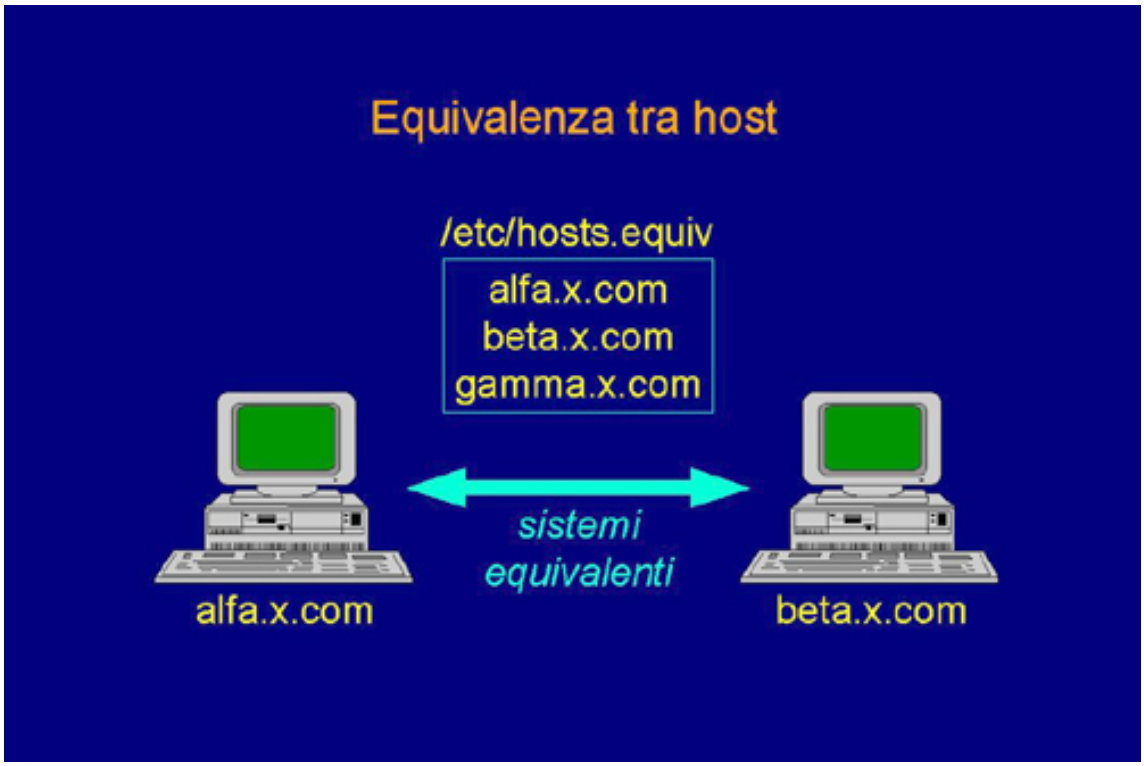
I comandi R



I comandi R

Quando da un sistema *Unix* si desidera svolgere delle operazioni su un altro sistema, esistono vari modi di farlo. Tipicamente si può ricorrere ad un collegamento in emulazione di terminale, come telnet, o un trasferimento di *file*, come FTP. Ma nel caso che siamo fra elaboratori collegati, tutti quanti, alla medesima rete locale, è abbastanza usuale usare i cosiddetti comandi R: un insieme di comandi che permettono l'esecuzione di una procedura remota su altri sistemi della nostra rete. Per esempio, con rlogin facciamo un emulazione di terminale, con rsh eseguiamo un particolare comando su un sistema remoto, rcp mi permette di copiare dei *file*, rdump mi permette di utilizzare un'unità nastro remota per fare un *dump* e così via. I comandi R possono funzionare richiedendo *username* e *password*, per permettere l'operazione sul sistema remoto, nel caso che i due sistemi coinvolti nello scambio di dati non siano equivalenti. Ma può lavorare anche in un modo che evita all'utente di dover ogni volta introdurre la *password*, semplicemente chiedendogli lo *username* e ovviamente l'indirizzo IP, questo fra sistemi equivalenti. Questa può essere la causa di un grosso problema di sicurezza all'interno dei sistemi *Unix*.

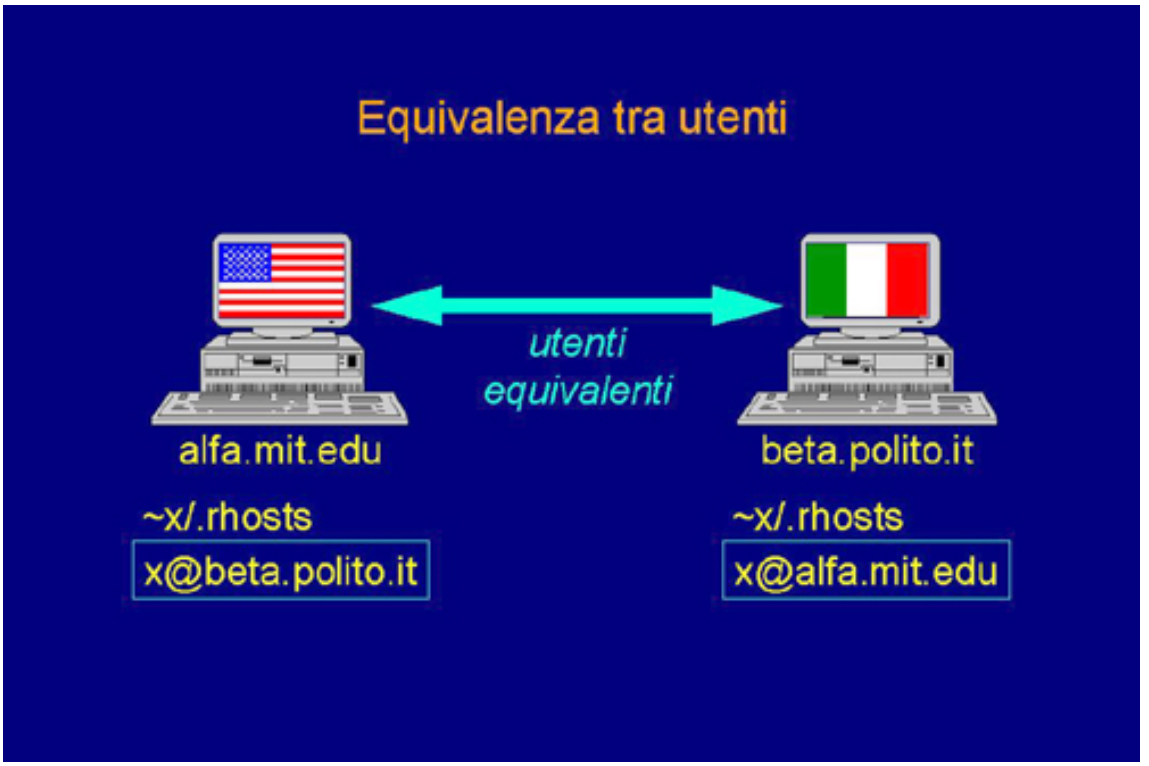
Equivalenza tra host



Equivalenza tra host

Il problema è legato alla definizione di sistemi equivalenti, ossia sistemi che possono permettere l'esecuzione di questi comandi R senza richiedere la *password* all'utente. È possibile definire due sistemi come completamente equivalenti, se su entrambe queste macchine viene introdotto nel direttorio etc il *file* chiamato hosts.equiv. Questo *file* contiene l'elenco di tutte le macchine che sono da considerarsi equivalenti dal punto di vista del *login*. Se un utente ha fatto *login* sulla macchina alfa e quindi si è già autenticato mostrando *username* e *password*, anche i sistemi beta e gamma accetteranno di eseguire per conto suo dei comandi, senza richiedergli ulteriormente la *password*. Poiché questa equivalenza tra *host* è definita in base al nome del calcolatore e quindi per traduzione in base al suo indirizzo IP, se qualcuno è in grado di assumere l'indirizzo IP di uno di questi sistemi, cosa che come abbiamo visto in passato è molto facile da fare, allora può a tutti gli effetti entrare facendo finta di essere un utente equivalente a quelli dei sistemi.

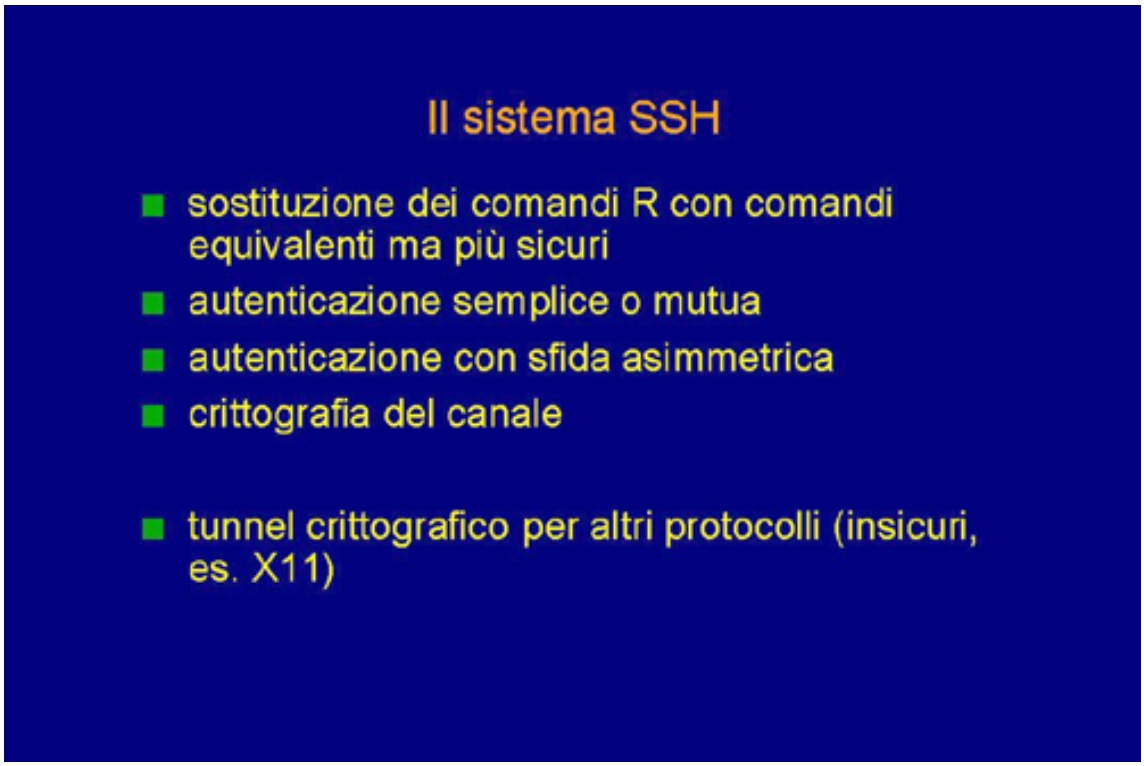
Equivalenza tra utenti



Equivalenza tra utenti

Esiste anche un altro tipo di equivalenza per i comandi R ed è l'equivalenza che non definisce tutti gli utenti di un sistema equivalenti a tutti gli utenti dell'altro sistema, ma definisce soltanto un equivalenza fra utenti specifici. Ad esempio, supponiamo di avere due calcolatori, uno situato in America e uno situato in Italia, ma di avere casualmente un utente che è il medesimo su entrambi i sistemi. Questo utente potrebbe desiderare di eseguire comandi R senza dover fornire la *password* ogni volta. Allora, è possibile definire non entrambi i sistemi come equivalenti, ma soltanto i due utenti come equivalenti. Per far questo occorre che l'utente, che abbiamo supposto avere *login* x, nella sua *home directory* (x indica il direttorio di *login*) abbia scritto un *file* chiamato .rhosts. All'interno di questo *file* deve essere messo il nome dell'utente e del calcolatore da considerare equivalente all'utente locale. Quindi, questo indica che l'utente x sul calcolatore alfa.mit.edu è da considerarsi equivalente all'utente x sul calcolatore beta.polito.it. L'equivalenza può essere monodirezionale o bidirezionale. Per averla bidirezionale occorre che anche sul calcolatore beta.polito.it venga messo un *file* analogo, che dichiari che l'utente x sul sistema americano è equivalente all'utente locale. Se questi *file* .rhosts non sono adeguatamente protetti, una persona malintenzionata può introdurre altre righe. Questo *file* può essere composto da più righe di testo e quindi introducendo altre persone queste possono essere dichiarate equivalenti ad un utente e quindi possono entrare su quel sistema senza fornire la *password*. Bisogna prestare particolare attenzione alle protezioni e al contenuto dei *file* .rhosts.

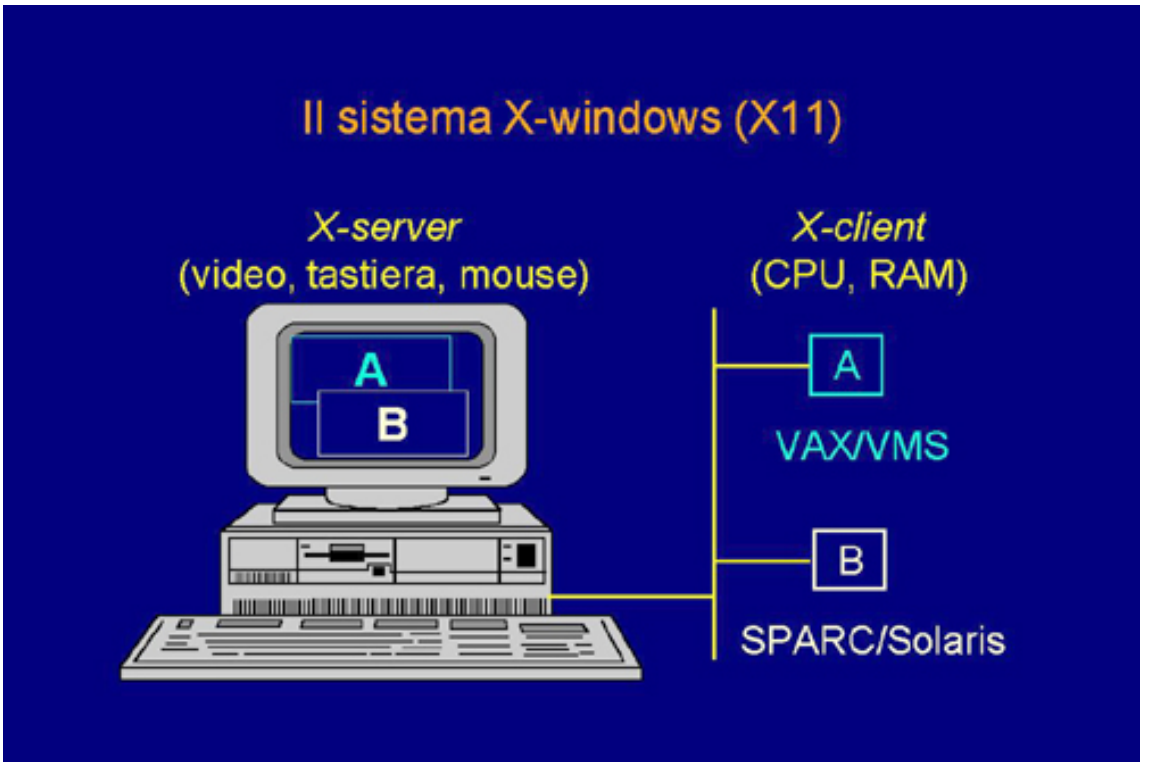
Il sistema SSH



Il sistema SSH

Una soluzione ancora migliore è non utilizzare i comandi R, visto che sono intrinsecamente insicuri. Esiste per *Unix* una sostituzione completa dei comandi R chiamato SSH. SSH è un programma sviluppato in Finlandia, ma diffuso ormai in tutto quanto il mondo, che sostituisce i comandi R con comandi equivalenti ma più sicuri. In pratica, permette di fare autenticazione semplice, quindi solo l'utente, o mutua, ossia anche il calcolatore mostra le proprie credenziali. L'autenticazione può essere fatta tramite una sfida asimmetrica, ossia usando chiavi pubbliche e chiavi private, che vengono assegnate sia all'utente sia ai sistemi a cui vogliamo collegarci. Inoltre, il canale di comunicazione tra l'utente ed il sistema viene crittografato, quindi anche un eventuale attaccante che potesse osservare la rete non può leggere i nostri dati. Una particolarità molto interessante del sistema SSH è l'esistenza di un tunnel crittografico, ossia SSH può veicolare al suo interno altri protocolli. Tipicamente questo viene fatto per protocolli insicuri, quali il sistema X11 che andiamo adesso ad esaminare.

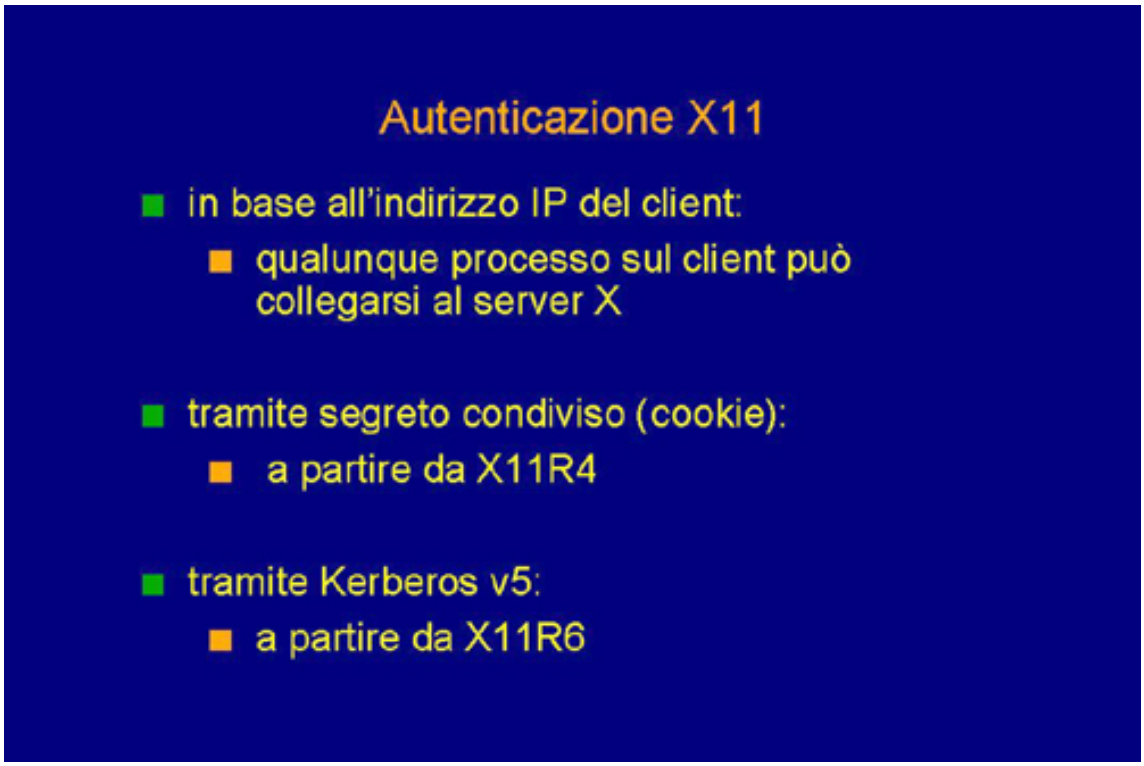
Il sistema X-windows



Il sistema X-windows

Il sistema X11 è un altro dei prodotti del famoso progetto *Athena* del *MIT*, di cui abbiamo già parlato a proposito del sistema *Kerberos*. Il sistema X11 è un sistema distribuito, ossia che funziona su una rete di calcolatori, grafico a finestre. Vedete qui l'illustrazione: nella terminologia di X-*windows* si parla di X-*client* per quei programmi che stanno svolgendo del calcolo, che quindi usano una certa *CPU*, una certa RAM e hanno bisogno del servizio di *input/output* nei confronti dell'utente. Il servizio di *input/output* viene fornito dal X-*server*, che è un programma che gira su una stazione personale dell'utente. Il X-*server*, in pratica, offre il servizio di video, tastiera, *mouse* ai programmi remoti. Trattandosi di un programma distribuito in rete, esso è anche eterogeneo, ossia le architetture dei programmi coinvolti possono essere diverse, come qui illustrato.

Autenticazione X11



Autenticazione X11

L'autenticazione all'interno del sistema X normalmente è molto debole, nel senso che esistono tre metodologie base. Quella utilizzata più di tutte è la prima che vedete indicata, perché è la più antica. È una autenticazione in base all'indirizzo IP del *client*: l'utente che controlla l'X-*server*, decide quali sono i *client* che possono accedere, ma purtroppo la granularità è solo a livello di indirizzo. Questo significa che qualunque processo che sia in esecuzione sul X-*client* può collegarsi al *server* e quindi può leggere quello che noi stiamo battendo sulla tastiera, o può visualizzare l'intero schermo e, a questo punto, può sottrarci qualunque tipo di dato. A partire dalla versione X11R4 è stato introdotto un altro tipo di autenticazione, i cosiddetti *cookie*, biscottini magici, che sono un segreto condiviso tra *client* e *server*. Solo i *client* che mostrano di conoscere questo *cookie* hanno diritto di accesso al *server*. A partire dalla versione X11 release 6, è stata anche introdotta l'autenticazione con *Kerberos* versione 5. Le autenticazioni con *cookie* e con *Kerberos* sono sicuramente più forti della normale autenticazione basata sull'indirizzo IP. Purtroppo, nella maggior parte dei casi, le implementazioni correnti utilizzano solo gli indirizzi e quindi il sistema X11 è potenzialmente fonte di grossi problemi all'interno del nostro sistema *Unix*.

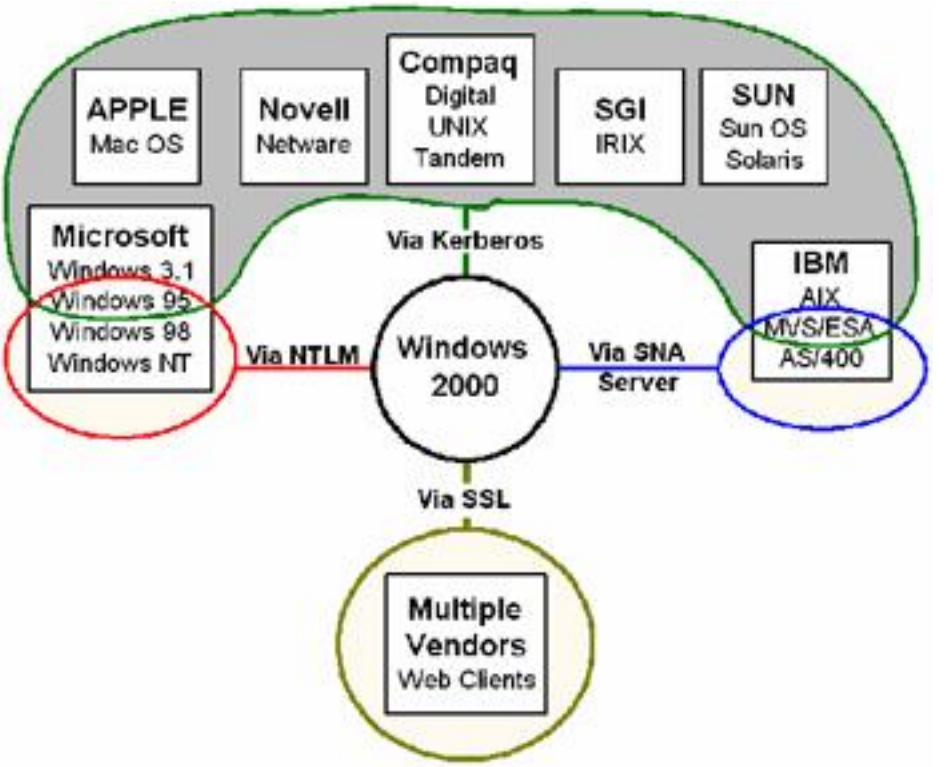
Sicurezza in Windows NT/2000

Franco Callegati
Walter Cerroni

9.1.1 (Implementare appropriate procedure per garantire la sicurezza di una rete), 9.1.5 (Usare un sistema di account sulla rete), 9.1.7 (Discutere gli aspetti connessi con le varie tecniche di autenticazione degli utenti)

Single sign on

L'acronimo SSO (*Single Sign-On*) si riferisce a qualsiasi tecnologia che sia in grado di facilitare il *logon* e l'accesso alle risorse della rete da più piattaforme. La procedura SSO non è necessariamente composta da una sola fase, così come non sempre prevede un *sign-on*. Per esempio, a volte le soluzioni SSO richiedono ulteriori informazioni di autenticazione per accedere a un sistema o a una rete differente. I meccanismi di SSO possono inoltre autenticare un utente che non ha compiuto il *sign-on* sul sistema (una cosa forse non particolarmente utile, ma comunque possibile).



Single Sign-On

Lo schema SSO permette di risolvere molti problemi delle organizzazioni che si affidano a importanti applicazioni che richiedono più di una piattaforma. Per l'autenticazione su più sistemi, gli utenti devono tenere a mente varie *password* ed essere in grado di usare interfacce differenti. In conseguenza, spesso scelgono *password* molto semplici oppure le scrivono su un foglio di carta, provocando in questo modo dei potenziali punti di debolezza nella sicurezza. Gli amministratori devono inoltre navigare attraverso molte interfacce di gestione per aggiungere nuovi utenti, eliminarne altri, oppure cambiare le *password* su vari sistemi diversi come NT, *Unix* e *Novell NetWare*. Questo procedimento è scomodo e può ridurre la sicurezza della rete.

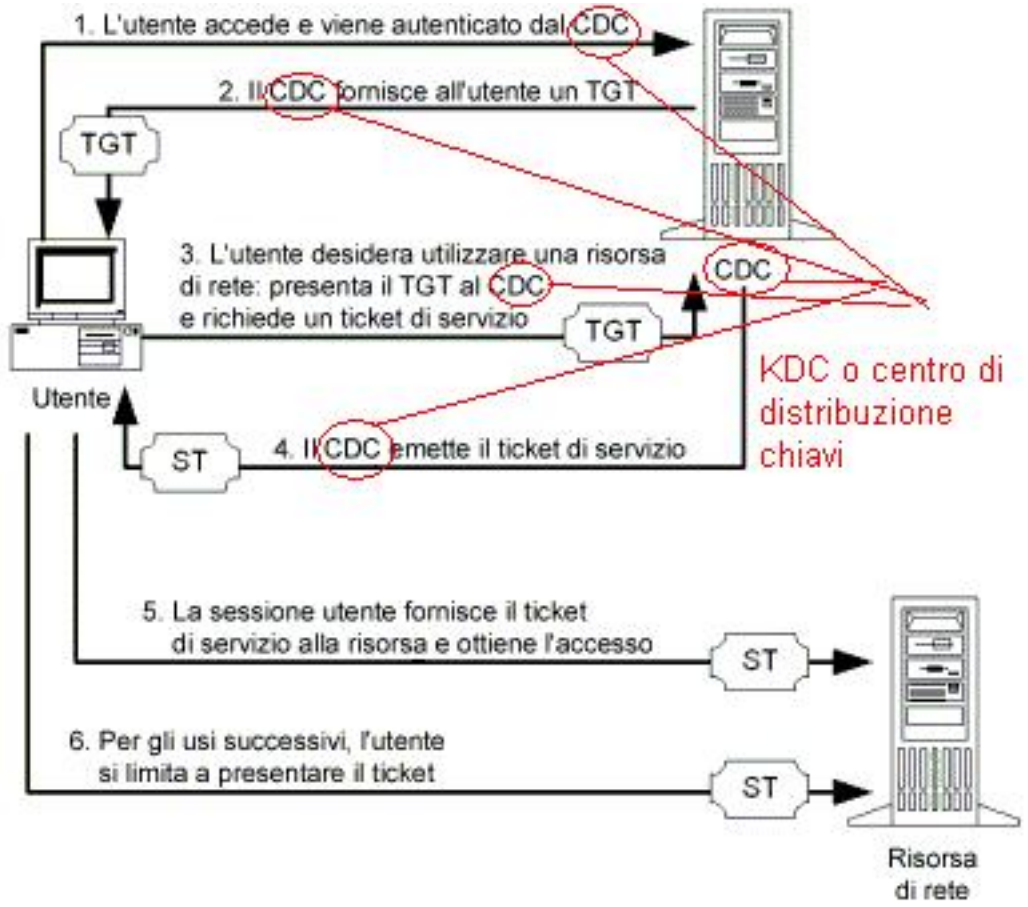
Lo schema SSO consente invece di risolvere questi problemi.

La procedura di SSO è più complessa negli ambienti misti, dal momento che nessuna soluzione universale va bene per qualsiasi ambiente e sistema operativo. In conseguenza, le organizzazioni devono identificare la portata della soluzione SSO e limitarla soltanto ad alcune applicazioni o piattaforme. Sono disponibili varie tecnologie SSO che permettono di adattare le soluzioni a problemi specifici. Queste alternative SSO comprendono la sincronizzazione delle *password*, l'automazione del *logon* e l'autenticazione basata su *token*.

SSO in Windows

La funzionalità SSO è disponibile in modo nativo nei domini *Windows* 2000 mediante il protocollo di autenticazione *Kerberos*, ovvero il protocollo di autenticazione predefinito di *Windows* 2000. L'utilizzo del protocollo *Kerberos* contribuisce in modo significativo al miglioramento delle prestazioni di rete, alla semplificazione della gestione amministrativa nonché alla protezione.

Il protocollo *Kerberos* si basa sul concetto di ticket, ovvero pacchetti di dati crittografati emessi da un'autorità ritenuta affidabile detta Centro distribuzione chiavi (KDC, *Key Distribution Center*). Un ticket comprova l'identità di un utente e contiene anche altre informazioni. Un centro KDC fornisce i ticket per tutti gli utenti della propria area di autorità o area di autenticazione. In *Windows* 2000, ogni *domain controller* funge da centro KDC e l'area di autenticazione di un *domain controller* corrisponde al dominio. Per ulteriori informazioni sul protocollo *Kerberos*, vedere il *white paper* di *Microsoft* dedicato all'anteprima delle funzionalità di protezione della rete mediante la tecnologia dei servizi di protezione distribuita di *Windows* 2000.



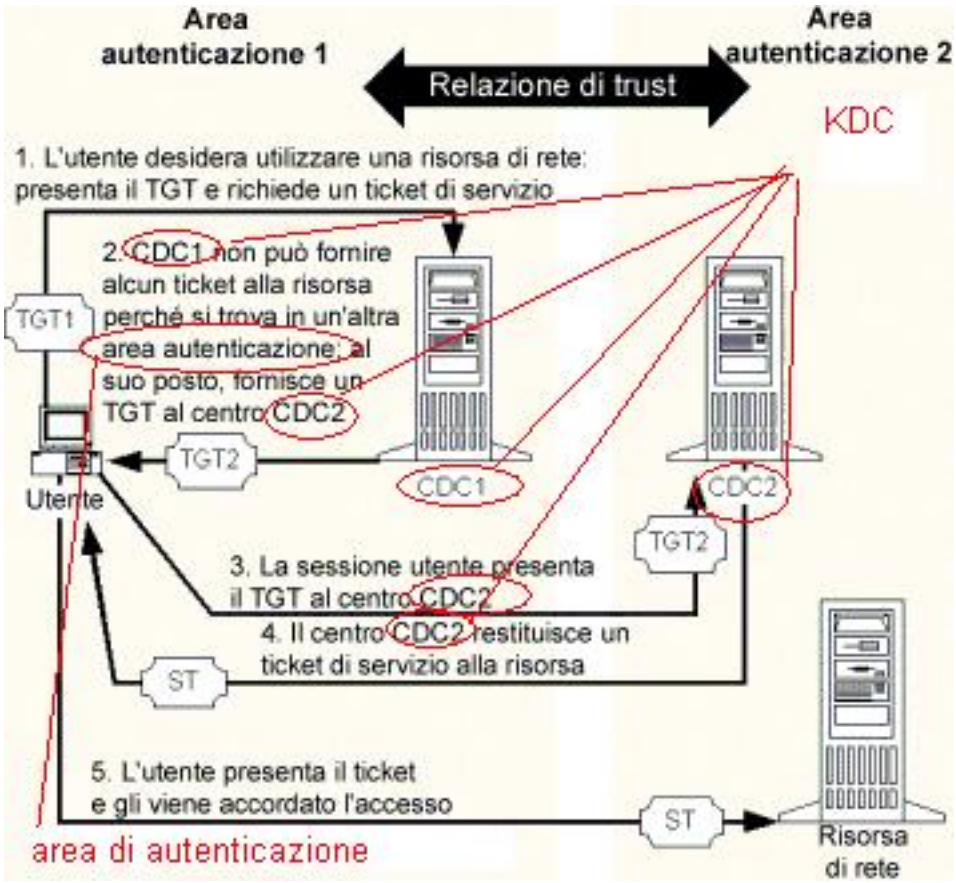
SSO in Windows 1

Al momento dell'accesso, l'utente viene autenticato da un centro KDC, che gli fornisce un ticket iniziale detto ticket di concessione (TGT, *Ticket Granting Ticket*). Quando l'utente deve utilizzare una risorsa di rete, la sessione utente corrispondente presenta il TGT al *domain controller* e richiede un ticket per quella determinata risorsa detto ticket di servizio (ST, *Service Ticket*), il quale viene presentato alla risorsa, che a sua volta concede l'accesso all'utente.

L'integrazione del protocollo *Kerberos* nel modello di protezione e amministrazione di *Windows* 2000 consente di gestire e controllare gli utenti e le risorse di rete in modo semplice ed efficiente. Un inconveniente ricorrente delle precedenti implementazioni del protocollo *Kerberos* è rappresentato dal fatto che esse sono aggiunte al sistema operativo anziché essere integrate. Il *software Kerberos* funziona al di sopra della normale architettura di protezione del sistema operativo piuttosto che come parte di esso, pertanto occorre che le informazioni specifiche della funzionalità SSO vengano memorizzate separatamente dalle altre informazioni di sistema e che l'amministratore apprenda l'utilizzo di ulteriori strumenti amministrativi al solo scopo di gestire l'infrastruttura SSO.

Tuttavia, il protocollo *Kerberos* è completamente integrato in *Windows* 2000 e ne rappresenta il metodo di autenticazione predefinito. Le informazioni relative alla funzionalità SSO vengono memorizzate in *Active Directory* insieme a tutte le altre informazioni concernenti gli oggetti di rete.

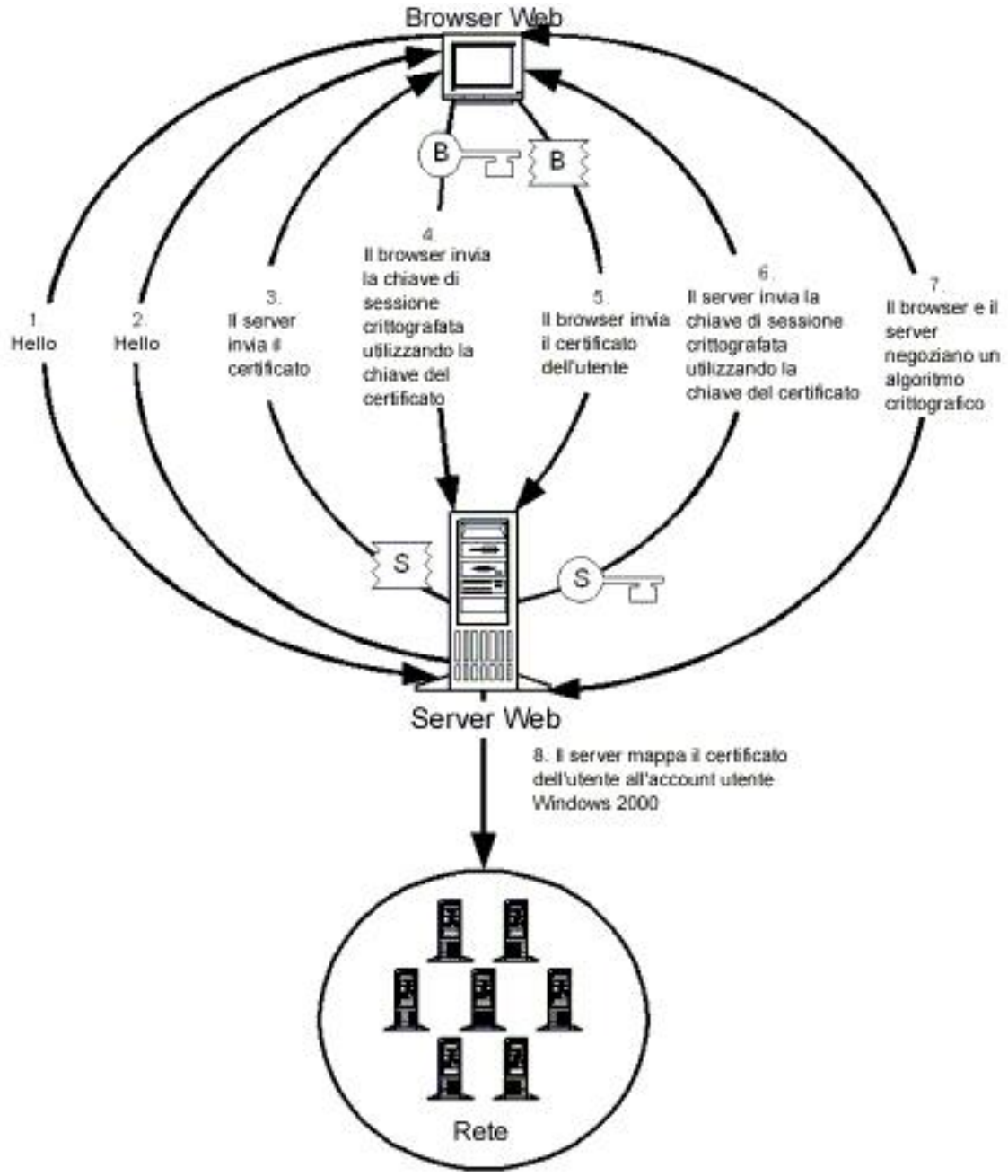
Le relazioni di *trust* tra domini in realtà presentano i *domain controller*, ossia i centri KDC *Kerberos*, ai due domini. Come illustrato di seguito, quando un utente di un dominio deve accedere a una risorsa di un dominio con il quale esiste una relazione di *trust*, il *domain controller* dell'utente provvede, mediante un riferimento tra aree di autenticazione, alla richiesta di un ticket al *domain controller* che possiede la risorsa. Tale *domain controller* considera affidabile il riferimento ed emette il ticket per l'utente. Questa integrazione è uno dei motivi fondamentali per cui le reti basate su *Windows* 2000 possono raggiungere dimensioni notevoli pur garantendo il controllo locale sulle risorse locali. Per impostazione predefinita, esiste una relazione di *trust* reciproca tra tutti i domini di una struttura di domini *Windows* 2000, che accettano pertanto i rispettivi riferimenti. Non esistono relazioni di *trust* predefinite tra le foreste in *Windows* 2000, ma è possibile definirle in modo semplice, se necessario.



SSO in Windows 2

In una rete omogenea basata su *Windows* 2000 l'amministratore non è costretto ad eseguire attività amministrative specifiche della funzionalità SSO, poiché tali funzioni sono integrate nelle funzionalità amministrative. Ad esempio, l'amministratore non ha mai bisogno di stabilire le chiavi crittografiche condivise dall'utente e dal *domain controller*. Quando l'amministratore crea l'*account* utente, le chiavi condivise vengono generate in modo trasparente nell'ambito del processo di creazione dell'*account* e vengono distribuite in modo protetto quando necessario. Analogamente, quando l'amministratore stabilisce le relazioni di *trust* tra i domini, le chiavi necessarie per gestire i riferimenti tra le aree di autenticazione vengono generate in modo trasparente e scambiate in modo sicuro.

Il protocollo SSL utilizza i certificati digitali come base per l'autenticazione. Per certificato digitale si intende un pacchetto di dati a prova di manomissione emesso da un'autorità di certificazione (CA), che fornisce una chiave pubblica e un nome e comprova che la persona o il *server* indicati sono i proprietari di tale chiave pubblica. Nella forma in cui vengono utilizzati in riferimento alla funzionalità SSO, i certificati vengono scambiati tra il *client Web* e il *server*, quindi le chiavi pubbliche in essi incorporate vengono utilizzate per lo scambio di informazioni quali le chiavi di crittografia delle sessioni. In questo modo vengono autenticati entrambi gli utenti, in quanto, se uno dei due non corrisponde all'entità indicata nel certificato presentato, non gli sarà possibile decifrare quanto inviato dall'altro utente. Per una descrizione completa del protocollo SSL, consultare la bozza delle relative specifiche formulate dal comitato IETF disponibile all'indirizzo: http://search.ietf.org/rfc/rfc2246.txt.



SSO in Windows 3

Tutti i componenti necessari a rendere disponibile la funzionalità SSO tramite il protocollo SSL sono inclusi in *Windows* 2000 e completamente integrati nell'architettura di protezione del sistema operativo. La gestione dei certificati digitali è possibile grazie all'infrastruttura a chiave pubblica PKI (*Public Key Infrastructure*) di *Windows* 2000, che è costituita da due componenti: Servizi certificati, che consente agli amministratori di emettere e revocare i certificati; *Active Directory*, che fornisce informazioni sui criteri e sulla posizione delle autorità di certificazione e consente la pubblicazione delle informazioni relative ai certificati revocati. I servizi di *hosting Web* di *Windows* 2000 sono forniti da *Microsoft Internet Information Service* (IIS), il *server Web* incluso nel S.O., che utilizza l'architettura di protezione di *Windows* 2000 per fornire il protocollo

SSL e gli altri servizi di protezione.

L'integrazione del protocollo SSL e dei certificati digitali nella piattaforma *Windows* 2000 si riflette nell'integrazione degli strumenti di gestione del protocollo SSL e dei certificati, realizzata nel gruppo di strumenti amministrativi di *Windows* 2000. Tutte le attività amministrative necessarie per rendere disponibile la funzionalità SSO tramite il protocollo SSL vengono svolte mediante *snap-in* MMC quali *Active Directory Manager*, Gestione certificati e Gestione servizio *Internet*.

Le password

NT utilizza il *manager* SAM (*Security Account Manager*) per archiviare e leggere le credenziali degli utenti come le *password*. Dal momento che questo modulo archivia le proprie informazioni nel *database* SAM, si può ritenere che NT sia sicuro soltanto quando lo sono i suoi dati SAM.

NT mantiene sul disco fisso una copia permanente e funzionante del *database* SAM. È possibile accedere a questo *database* sotto la chiave SAM nell'*hive* del Registro di Configurazione *HKEY_LOCAL_MACHINE*, sia scrivendo un apposito programma che utilizzando un *editor* del Registro di Configurazione come regedt32.exe.

***Hash* delle *password* in memoria**

Per *default*, NT compie il *caching* delle credenziali di *logon* per gli ultimi dieci utenti che si sono collegati in modo interattivo. Lo scopo di questa funzione è quello di consentire all'utente di compiere il *logon* sul sistema anche se questo è scollegato dalla rete, oppure se i *controller* di dominio non sono disponibili. Anche se NT offre una certa protezione per la *cache* delle credenziali di *logon*, se l'ambiente richiede un livello maggiore di sicurezza potrebbe essere necessario disattivare completamente il *caching*, dal momento che qualcuno potrebbe attaccarlo.

È necessario tenere presente che le credenziali della *cache* di *logon* contengono gli *hash* delle *password* di altri *hash*, che rendono i dati difficili da leggere o da usare per compiere un tentativo di *logon* non autorizzato. Attualmente, non si è ancora verificato nessun tentativo noto e coronato da successo di attacco a questa *cache*. Per disattivare il *caching* delle credenziali, è necessario modificare l'elemento *CachedLogonCount* (di tipo REG_DWORD, valore 0) nella chiave:

HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVerion\Winlogon del Registro di Configurazione.

Il *database* SAM sulla rete

NT utilizza il protocollo SMB (*Server Message Block*), che è stato sviluppato congiuntamente da *Microsoft, IBM* e *Intel* e definisce una serie di comandi a livello di programma per ottenere o fornire servizi di *file* remoti in un ambiente di rete. Una sessione SMB prevede la trasmissione in rete di pacchetti contenenti informazioni riservate. Tra le altre informazioni, quando NT trasmette i pacchetti durante la fase di autenticazione di una sessione SMB, questi pacchetti contengono tipicamente una versione crittografata del *challenge* NTLM.

Gli attaccanti che utilizzano un qualsiasi *sniffer* di pacchetti tra quelli disponibili sul

mercato possono facilmente ottenere i dati inviati attraverso la rete. Anche se il lavoro necessario per ottenere i pacchetti giusti ed estrarre da questi ultimi le informazioni delle *password* non è mai stato particolarmente semplice, la situazione è tuttavia cambiata notevolmente dopo il rilascio di SMB *Packet Capture* da parte di *L0pht Heavy Industries*; si tratta di un *tool sniffer* SMB che ora è stato strettamente integrato in *L0phtCrack*.

Chi dispone di questo *software* può ottenere facilmente dalla rete gli *hash* delle *password* basati sul protocollo SMB. L'*utility sniffer* incorporata in *L0phtCrack* legge silenziosamente gli *hash* delle *password* SMB e li archivia per poterli decodificare.

Una volta decodificate, queste *password* permettono un facile accesso a qualsiasi risorsa di rete alla quale può accedere l'*account* dell'utente. Il rischio in questo caso è evidente, ma la prevenzione è semplice. Per proteggersi nei confronti di un attacco di questo tipo è necessario usare NTLMv2, fornito con il *Service Pack* 4 o 5, oppure un *tool* VPN (*Virtual Private Network*) come il protocollo PPTP (*Point To Point Tunneling Protocol*) di *Microsoft*. NTLMv2 dovrebbe essere sufficiente per proteggere i dati nel momento in cui viaggiano nella LAN interna, mentre il protocollo PPTP aiuta a proteggere le informazioni durante il loro viaggio attraverso le reti non affidabili (per esempio *Internet*).

Il processo di logon

La gestione della sicurezza dell'*account*, è a cura del SAM, che contiene informazioni su tutti gli *account* utente e gruppo. SAM viene utilizzato da LSA per autenticare gli utenti durante il processo di collegamento, confrontando il nome utente e la *password* inserita dall'utente nella registrazione dell'utente nel *database* SAM. SAM crea anche gli identificatori di sicurezza (**SID**) che identificano univocamente ogni *account* utente e di gruppo. Un **SID** è un numero simile al seguente:

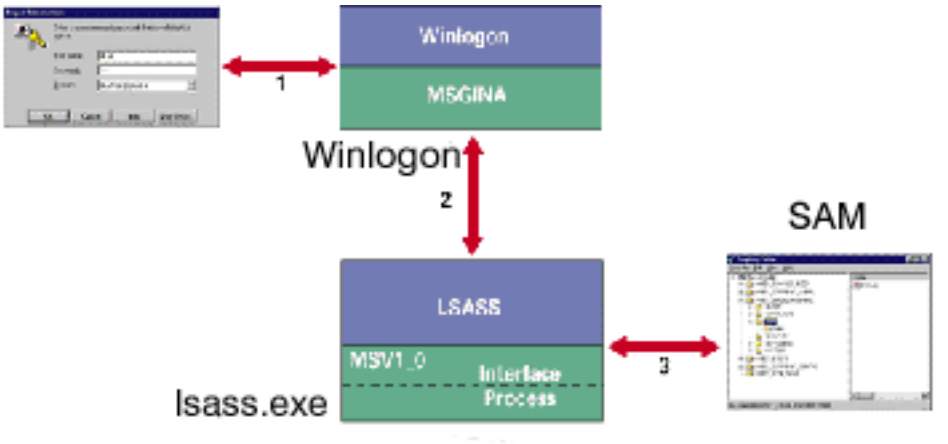
S-1-5-21-2087915065-388913830-1877560073-500

L'utente, o gruppo, è identificato nel sistema dal **SID**, non dal nome che viene fornito per comodità dagli utenti. Quando un *account* utente o di gruppo viene eliminato, il **SID** viene ritirato. È impossibile quindi ricreare lo stesso *account* utente o di gruppo con lo stesso **SID**. Un nuovo *account* potrebbe avere il vecchio nome, ma gli verrebbe assegnato un nuovo **SID**.

I **SID** sono il motivo per cui occorre particolare attenzione quando si elimina un *account* utente o di gruppo, perché non è più possibile ottenere lo stesso *account*. Se si reinstalla *Windows* NT, operazione diversa da quella di aggiornamento, dopo l'installazione il *computer* potrà avere lo stesso nome, ma avrà un nuovo **SID** e sarà essenzialmente un nuovo *computer* sulla rete. I *database* di sicurezza sulla rete conterranno il vecchio **SID** ma non riconosceranno la nuova configurazione del *computer*, a causa del cambiamento del **SID**.

Procedura di *logon*

I collegamenti obbligatori sono la chiave nel sistema di sicurezza di *Windows* NT. Il processo di collegamento di *Windows* NT segue le seguenti fasi:



Il processo di logon

- In *Windows* NT, la combinazione CTRL+ALT+CANC produce l'esecuzione del programma *WinLogOn*, un componente di LSA, che visualizza la finestra di dialogo GINA, che richiede l'immissione di un nome utente un dominio e una *password*. Se il nome utente o la parola chiave non sono corretti, il sistema risponde con il messaggio *User Authentication Failure*, che avvisa del fallimento dell'autenticazione dell'utente. Il sistema non specifica se la causa dell'errore risieda nel nome utente o nella *password*.
- Se il nome utente e la *password* sono validi, LSA passa i dati di *logon* al sottosistema di sicurezza.
- Il sottosistema di sicurezza passa tali dati al SAM.
- SAM consulta il *database* della sicurezza dell'*account* per determinare se il nome utente e la *password* corrispondano ad un *account* del *database*. Queste informazioni vengono restituite al sottosistema di sicurezza insieme alle informazioni sui permessi dell'utente, sulla posizione della *home directory* e sul profilo utente. Se l'utente ha un *file script* di *login*, questo viene eseguito.
- Se è stata trovata una corrispondenza per il nome utente e la *password* e se l'*account* utente ha i permessi sul sistema, il sottosistema di sicurezza genera un *token* d'accesso che rappresenta l'utente e i *server* come una chiave che contiene le credenziali dell'utente sul sistema. Il *token* d'accesso viene passato al sottosistema *Win*32.
- Il sottosistema *Win*32 genera un nuovo processo che è associato ai *token* d'accesso.
- Il processo creato da *Win*32 è utilizzato per avviare explorer.exe di *Win*32 attivando il *desktop*.

Nell'ambiente di domino di *Windows* NT *Server* si verificano due tipi di collegamenti:

- Collegamenti interattivi. Questi si hanno in risposta ad una richiesta di collegamento nella finestra di dialogo specifica. L'utente effettua il *logon* in un dominio o nel *computer* locale a seconda se il campo *Domain* specifichi un dominio o il nome di un *computer* locale.
- Collegamenti remoti. Quando un utente è già collegato ad un *account* utente e cerca di stabilire una connessione di rete ad un altro *computer*, si

verifica un collegamento remoto. Un esempio di collegamento remoto è il tentativo di sostituire un'unità di disco di rete o di accedere ad una stampante condivisa.

Supportando i collegamenti remoti, *Windows* NT *Server* consente agli utenti di accedere alle risorse disponibili su altri *computer* e in altri domini.

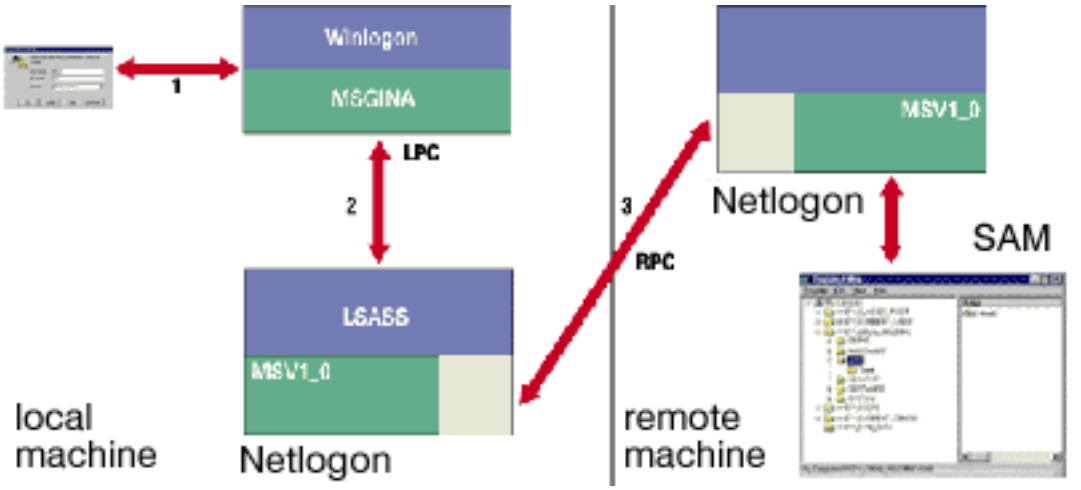
Il processo di net-logon

Il processo di collegamento è controllato dal servizio *NetLogon* che dipende dal servizio *Workstation*. Il servizio *NetLogon* ha tre compiti rispetto al processo di collegamento:

- **Ricerca**. Quando un *computer* con *Windows* NT viene attivato, il servizio *NetLogon* tenta di individuare un *controller* di dominio (un PDC o un BDC) per il dominio che è specificato nel campo *Domain* della finestra di dialogo *LogOn*. Il controllore di dominio scoperto viene quindi contattato per l'autenticazione del collegamento dell'utente.
- Attivazione di un **canale sicuro di comunicazioni**. I *computer* con *Windows* NT che comunicano richieste e risposte su questioni per stabilire l'esistenza di *account* utente validi. Dopo la verifica, viene stabilita una sessione di comunicazione tra i *computer* e vengono trasmessi i dati d'identificazione degli utenti.
- L'**autenticazione *pass-through***. Quando un utente tenta di accedere a una risorsa di rete e l'accesso richiesto non può essere autenticato da un *controller* di dominio del dominio di collegamento dell'utente, la richiesta viene passata a un *controller* di dominio nel dominio di destinazione per l'autenticazione.

L'autenticazione *pass-through* rende la struttura della rete considerevolmente più trasparente all'utente che sta tentando di accedere alle risorse condivise. L'utente viene esplicitamente autenticato durante il collegamento a un dominio. I tentativi di accedere alle risorse in altri domini sono gestiti attraverso l'autenticazione *pass-through*, eliminando la necessità di collegarsi esplicitamente a ulteriori domini. I domini a cui un utente può accedere attraverso l'autenticazione *passthrough* sono determinati da relazioni di fiducia che vengono stabilite tra i domini.

Il servizio *NetLogon* è responsabile anche della duplicazione dei cambiamenti nel *database* SAM sul PDC nei BDC nel dominio. Il processo di recupero della coerenza delle diverse copie del *database* SAM viene chiamato sincronizzazione.



Il processo di net-logon

Controllo d'accesso

Sebbene sia possibile configurare, a livello di sistema ed attraverso il *database* SAM, le aree di controllo relative a politica degli *account*, diritti degli utenti e politica di *audit*, molte altre aree di controllo (1 - **ACL**, 2 - *audit control lists*, 3 - autorità amministrativa e 4 - servizi di sistema) operano invece indipendentemente dal *database* SAM.

- **ACL**. È possibile usare i permessi a livello di oggetto per accedere ad oggetti quali *file* e cartelle. Ciascun oggetto dispone di una lista di permessi, chiamata **ACL**, che definisce gli utenti ed i gruppi che possono accedere all'oggetto e in che modo possa avere luogo questo accesso. Non bisogna confondere i permessi con i diritti; si tratta infatti di elementi completamente diversi in NT. I diritti vengono assegnati a livello di sistema tramite *User Manager*. I permessi vengono invece definiti a livello di oggetto, tramite le liste **ACL**. Per visualizzare la lista **ACL** di un *file*, aprire *Windows Explorer*, fare clic sul *file* con il pulsante destro del *mouse* e selezionare *Properties*. Porsi in corrispondenza della scheda *Security* e fare clic su *Permissions*. NT accorda agli utenti un accesso cumulativo, a seconda dell'appartenenza ai gruppi.
- ***Audit control list***. Ciascun oggetto dispone anche di una *audit control list*, che definisce i tipi di accesso all'oggetto che NT dovrebbe registrare nel *log Security* locale. In corrispondenza di ciascun tipo di accesso (per esempio, *Read*, *Write*, *Delete*), è possibile indicare se NT deve registrare gli eventi riusciti o falliti. Per esempio, per effettuare il *logging* dei tentativi effettuati dagli utenti per leggere un documento confidenziale al quale non hanno accesso, bisogna attivare l'*auditing* per quel documento in modo che NT registri i tentativi *Read* falliti da parte dei membri del gruppo *Everyone*. Per tener traccia dei cambiamenti apportati ad un *file*, attivare l'*auditing* di quel *file* in modo che NT registri i tentativi *Write* riusciti. Per visualizzare la *audit control list* di un *file* o di una cartella, aprire *Windows Explorer*, fare clic sull'oggetto con il pulsante destro del *mouse* e selezionare *Properties*. Porsi in corrispondenza della scheda *Security* e fare clic su *Auditing*. Per

visualizzare la *audit control list* di una chiave del registro di configurazione, aprire regedt32, selezionare la chiave e quindi selezionare *Security*, *Permissions*, oppure *Security*, *Auditing* dalla barra del menu.

- **Autorità amministrativa**. I gruppi *Administrators* e *Power Users*, incorporati in NT, offrono la possibilità di controllare l'autorità amministrativa. I membri del gruppo *Administrators* possono condividere *directory* e stampanti, creare e gestire *account* macchina locali e gruppi macchina locali nel *database* SAM locale del *computer*, assegnare diritti agli utenti ed impostare la politica di *audit*. I membri del gruppo *Power Users* possono condividere *directory* e stampanti, creare e gestire *account* macchina locali e gruppi macchina locali.
- **Servizi di sistema**. I servizi rappresentano le porte di accesso al sistema NT per l'ingresso dalla rete; in conseguenza, è necessario identificare tutti i servizi in esecuzione su ciascun *computer* e stabilire se sia opportuno disattivare qualcuno di questi servizi. Per visualizzare i servizi del sistema locale, aprire la *applet Services* del pannello di controllo. Per visualizzare i servizi di un sistema remoto, aprire *Server Manager* (sotto *Administrative Tools*), selezionare il *computer* desiderato, quindi selezionare *Computer, Services* dalla barra del menu (se sul vostro sistema non riuscite a trovare *Server Manager*, copiate semplicemente srvmgr.exe da un *controller* di dominio).

Sicurezza a livello di dominio

Anche se i sistemi NT possono funzionare in modo indipendente e sicuro anche senza i vantaggi di un dominio, il fatto di basarsi unicamente sulla sicurezza a livello di *host* è accompagnato da alcune limitazioni che si manifestano quando la rete ha molti utenti che devono accedere alle risorse su vari *server*. Quando un *computer* non è membro di un dominio, per effettuare il *logon* gli utenti devono usare un *account* macchina locale nel *database* SAM di quel *computer*. In mancanza di un dominio, ciascun utente ha bisogno di vari *account* utente - uno per ciascun *server* al quale deve accedere. La presenza di molteplici *account* per ogni utente comporta vari problemi.

Gli utenti vengono sovraccaricati dal lavoro legato alla sincronizzazione delle *password* per ogni *account*. Quando i dipendenti abbandonano l'azienda, bisogna individuare e cancellare tutti i loro *account*. È facile dimenticarsi di un *account* e ritrovarsi con un problema di accesso residuo (in altre parole, gli utenti sono sempre in grado di entrare nella rete, anche molto tempo dopo che hanno lasciato l'azienda). Per risolvere con facilità questi problemi, è opportuno creare un dominio ed assegnare a ciascun utente un *account* di dominio che risulti valido per tutti i *server* ai quali deve accedere.

Quando una *workstation* o un *server* membro si unisce ad un dominio, non perde nessuna delle sue impostazioni di sicurezza specifica a livello di *computer*, ma ottiene ulteriori funzionalità a livello di dominio. Quando un *server* o una *workstation* si unisce al dominio, accorda la fiducia ai *controller* di dominio per l'autenticazione degli utenti. In conseguenza, questi ultimi possono effettuare il *logon* usando un *account* macchina locale, oppure un *account* di dominio. Quando un *computer* si unisce ad un dominio, è anche possibile aggiungere gruppi ed utenti del dominio alle liste **ACL** dei *file* e degli altri oggetti sul *computer*. NT aggiunge automaticamente il gruppo *Domain Admins* del dominio al gruppo *Administrators* locale del *computer* (ciò spiega perché i membri di

Domain Admins abbiano l'accesso amministrativo a tutti i *computer* del dominio.

Mentre i *database* SAM delle *workstation* o dei *server* membri sono specifici e limitati ad un *computer*, il *database* SAM di un *controller* di dominio non lo è affatto. Il SAM di un *controller* PDC è il SAM per l'intero dominio, e *User Manager for Domains* lo apre automaticamente. Quando si crea un utente o un gruppo, oppure si cambia una politica in *User Manager for Domains*, l'*utility* registra la modifica nel *database* SAM del *controller* PDC. Qualche minuto dopo, il *controller* PDC replica questa modifica sul *database* SAM di ogni *controller* BDC (*Backup Domain Controller*). In conseguenza - tenendo conto della latenza tra i cicli di replicazione - tutti i *controller* in un dato dominio condividono un *database* SAM duplicato e dispongono degli stessi *account* utente, gruppi e politiche. Per esempio, quando si attiva l'*auditing* in *User Manager for Domains*, viene attivato l'*auditing* per il *controller* PDC. Quando avviene la replicazione, NT attiva l'*auditing* su ciascun *controller* BDC.

Controllo di dominio o locale

Oltre ad ottenere ulteriori funzionalità di dominio, la configurazione della sicurezza NT di un *computer* opera indipendentemente dopo l'aggiunta del *computer* ad un dominio. Le modifiche apportate tramite *User Manager for Domains* non hanno alcun effetto sulle politiche o sui diritti utente delle *workstation* e dei *server* membri. Per esempio, quando si usa *User Manager for Domains* per assegnare il diritto *Logon locally* a *Peter*, si autorizza questo utente ad effettuare il *logon* sulle *consolle* di tutti i *controller* del dominio. Se tuttavia l'utente deve effettuare il *logon* su un *server* membro del dominio, l'amministratore dovrà fare *logon* su quel *server* ed accordare localmente a *Peter* il diritto attraverso *User Manager*.

La politica degli *account* e la politica di *audit* che vengono specificate in corrispondenza del *controller* di dominio non hanno niente a che fare con la politica di *audit* e gli *account* definiti localmente sulle *workstation* ed i *server* membri. Per esempio, si supponga di fare *logon* su un *controller* di dominio, aprire *User Manager for Domains* e configurare la lunghezza minima delle *password*. Questa azione non ha alcun effetto sugli *account* macchina locali definiti nel *database* SAM di una *workstation* o di un *server* membro; in conseguenza, se le politiche degli *account* locali sono piuttosto trascurate, la situazione è critica per la sicurezza. Sarà infatti possibile utilizzare un *account* locale per attaccare un sistema.

In conseguenza, si dovrebbe evitare la creazione di molti *account* macchina locali. È invece opportuno fornire a ciascun utente un *account* di dominio, valido per tutti i *computer* nel dominio. Se ci si trova nella situazione in cui un amministratore ha creato degli *account* macchina locali su una *workstation* o un *server* membro, sarà possibile riconfigurare il *database* SAM locale della *workstation* o del *server* membro attraverso la rete. Copiare semplicemente sulla propria *workstation User Manager for Domains* (usrmgr.exe) da qualsiasi *server* NT. Aprire il programma e selezionare *User, Select Domain*, ma non inserire un nome di dominio. Inserire invece due sbarrette rovesciate ed il nome di un *computer* remoto (per esempio, se si desidera configurare il *database* SAM su un *computer* chiamato kramer, digitare \\kramer). Per verificare che venga effettivamente usato il *database* SAM del sistema remoto e non quello del dominio, osservare la barra del titolo della finestra, che dovrà comprendere due sbarrette rovesciate.

La sicurezza dei domini NT non è centralizzata come potrebbe apparire a prima vista.

Ogni *computer* dispone di una configurazione discreta, con centinaia di opzioni di sicurezza che richiedono attenzione. Anche negli ambienti di dominio, solo gli *account* ed i gruppi degli utenti risultano veramente centralizzati (e soltanto se si evita la creazione di *account* macchina locali sui *server* membri). In conseguenza, una completa conoscenza delle opzioni di sicurezza NT di base è una pietra miliare per avere una rete sicura.

La sicurezza nei sistemi basati su Windows 2000 e XP

Il cambiamento del livello di autenticazione di maggior risalto in *Windows* 2000 è costituito dall'uso di *Kerberos* e dal nuovo protocollo di autenticazione di *default*. Ogni *server* e *workstation Windows* 2000 è dotato di un *provider* di autenticazione *Kerberos client*. *Windows* 2000 non include tuttavia il supporto *Kerberos* per altre piattaforme *Microsoft*; se si desidera che i propri *client* NT 4, *Windows* 95, o *Windows* 98 effettuino l'autenticazione usando *Kerberos*, sarà necessario aggiornare le *workstation* a *Windows* 2000 *Professional*.

Due altri cambiamenti importanti nel sistema operativo che influenzano l'autenticazione di *Windows* 2000 sono costituiti dal supporto per *Active Directory* quale *database* per la convalida delle credenziali (ogni *controller* di dominio *Windows* 2000 dispone di una *Active Directory*) e l'inclusione di un nuovo *provider* SSP (*Security Support Provider*): il SSP *Negotiate*. Questo nuovo *provider* SSP attiva la negoziazione del protocollo di autenticazione tra il *server* e un *client* connesso tramite RPC (*Remote Procedure Call*).

Per il momento, il SSP *Negotiate* può gestire soltanto i protocolli di autenticazione *Kerberos* e NTLM (NT LAN *Manager*). La sua prima opzione è sempre costituita dal protocollo *Kerberos*. Se quest'ultimo non è disponibile, il sistema passa a NTLM.

Kerberos

Kerberos viene specificato nel documento RFC (*Request for Comments*) 1510 della IETF (*Internet Engineering Task Force*). Ciò ne fa uno *standard* aperto, che può essere usato per fornire lo schema SSO (*Single sign-on*) tra *Windows* 2000 e altri sistemi operativi che supportano un'implementazione *Kerberos* basata sul documento RFC 1510. L'uso di *Kerberos* presenta anche altri vantaggi:

- autenticazione più veloce;
- mutua autenticazione;
- supporto per la delega dell'autenticazione;
- relazioni di fiducia transitive;
- *logon* con *smartcard*.

Grazie al suo univoco sistema di *ticketing*, *Kerberos* non ha bisogno dell'autenticazione *pass-through* e può offrire un'autenticazione più veloce. L'autenticazione *pass-through* è un meccanismo che viene usato durante la sequenza di autenticazione NT 4 basata su NTLM.

Mutua autenticazione: ciò significa che il *client* si autentica con il servizio che è responsabile per la risorsa e che quest'ultimo si autentica con il *client*. Questa è un'altra importante differenza rispetto al protocollo NTLM. Lo schema

response di NTLM offre unicamente l'autenticazione del *client*: il *server* interroga il *client*, e quest'ultimo calcola una risposta che viene convalidata dal *server*. Usando NTLM, gli utenti potrebbero fornire le proprie credenziali a un *server* fasullo.

Kerberos supporta delle **relazioni di fiducia transitive** interdominio, che permettono di ridurre il numero delle relazioni di fiducia necessarie per l'autenticazione. Ciò significa che se *europe.compaq.com* e *us.compaq.com* accordano la fiducia a *compaq.com*, allora *europe.compaq.com* accorda implicitamente la fiducia anche a *us.compaq.com*. *Kerberos* di *Windows* 2000 supporta inoltre un meccanismo noto come delega dell'autenticazione o inoltro delle credenziali.

La **delega** può essere considerata come un'ulteriore fase dopo l'impersonalizzazione: grazie a quest'ultima, un servizio può accedere alle risorse locali per conto di un utente; grazie alla delega, un servizio può accedere anche alle risorse *remote* per conto dell'utente. Il vero significato della delega è che l'utente A può assegnare alla macchina intermediaria B i diritti di autenticarsi su un *server* di applicazioni C come se la macchina B fosse l'utente A. Il *server* di applicazioni C basa le proprie decisioni di controllo dell'accesso sull'identità dell'utente A invece che sull'*account* della macchina B. Nella terminologia *Kerberos*, ciò significa che i ticket dell'utente possono essere inoltrati da una macchina all'altra.

Attraverso l'estensione *Kerberos* **PKINIT** (*Public Key cryptography for INITial authentication*), *Windows* 2000 include il supporto per il *logon* con *smartcard*. Quest'ultimo offre un'autenticazione molto più forte rispetto al *logon* con *password*, dal momento che è basato su un'autenticazione a due fattori: per effettuare il *logon*, l'utente ha bisogno di una *smartcard* e del relativo codice *PIN* (*Personal Identification Number*). In generale, il *logon* tramite *smartcard* offre anche una sicurezza più forte: permette infatti di bloccare gli attacchi dei cavalli di Troia che cercano di ottenere la *password* dell'utente leggendola dalla memoria del sistema.

Il protocollo *Kerberos* di base, così come viene definito nel documento RFC 1510, gestisce unicamente l'autenticazione. L'implementazione di questo protocollo compiuta da *Microsoft* permette di gestire anche i dati di controllo dell'accesso. I ticket *Kerberos* di *Windows* 2000 contengono le informazioni relative ai privilegi e all'appartenenza ai gruppi dell'utente, in un campo chiamato PAC (*Privilege Attribute Certificate*). *Kerberos* di *Windows* 2000 può inoltre fornire una chiave segreta, che può essere usata per l'autenticazione dei pacchetti, l'integrità e i servizi di riservatezza per i pacchetti scambiati dopo la sequenza iniziale di autenticazione.

Controllo dell'accesso

Windows 2000 include alcune nuove funzionalità per il controllo dell'accesso, ma il modello del controllo dell'accesso è fondamentalmente lo stesso che viene usato in NT 4. Questo modello si basa ancora sui seguenti concetti chiave:

- *access token*;
- *access mask*;
- *security descriptor*;
- *impersonation*.

Il controllo dell'accesso di *Windows* 2000 comprende alcune importanti modifiche relativamente alle liste **ACL** (*Access Control List*), agli elementi **ACE** (*Access Control*

Entry) associati, al modo in cui viene amministrato, all'ereditarietà, alle regole di valutazione delle **ACL** e all'**ACL** *editor*.

Il nuovo ACL *editor*

La caratteristica più importante del nuovo **ACL** *editor* di *Windows* 2000 è costituita dall'indipendenza dagli oggetti (lo stesso *editor* viene usato per impostare il controllo dell'accesso su diversi tipi di oggetti che possono essere resi sicuri) e dal supporto per elementi **ACE** *deny* e le nuove regole di valutazione **ACL**. Mentre NT 4 supporta elementi **ACE** negativi, l'**ACL** *editor* non è invece in grado di gestirli. Quando in NT 4 si impostano gli elementi **ACE** *deny* a livello di programma, usando l'**ACL** *editor* di NT 4 veniva visualizzato un messaggio di errore. Le nuove regole di valutazione **ACL** verranno spiegate più avanti. Il nuovo **ACL** *editor* dispone di una **GUI** (*Graphical User Interface*) completamente nuova, che consiste in una vista di base e una vista avanzata.

Per accedere in maniera semplice e immediata all'**ACL** *editor*, è sufficiente selezionare una cartella del *file system* e fare clic con il tasto destro del *mouse* e attivare il tab Protezione.

ACE basati sul tipo di oggetto

Windows 2000 supporta gli elementi **ACE** *object-type*. Si tratta di una funzionalità della versione 4 del modello per il controllo dell'accesso di *Windows*. *Microsoft* ha implementato questa nuova versione unicamente per gli oggetti *Active Directory*. Il motivo principale per cui *Microsoft* ha incorporato questa nuova versione **ACL** è stato quello di attivare in un modo più granulare la definizione del controllo dell'accesso sugli oggetti *Active Directory*.

Queste modifiche attivano anche una delega amministrativa di grana fine sugli oggetti *Active Directory*, un'altra funzionalità chiave di *Windows* 2000. Usando gli **ACE** *object-type*, l'amministratore può creare delle impostazioni per il controllo dell'accesso a grana fine per gli oggetti *Active Directory*. Per esempio, questi elementi si possono usare per definire a quali tipi di oggetti (utenti, gruppi, o *computer*) verrà applicato un elemento **ACE** impostato su un'unità OU (*Organizational Unit*).

Gli **ACE** *object-type* attivano altri **ACE** basati su proprietà, insieme di proprietà e diritti estesi. Gli **ACE** basati sulle proprietà e insiemi di proprietà si possono usare per impostare il controllo dell'accesso su una proprietà o un insieme di proprietà di un oggetto. Alcuni esempi di proprietà dell'oggetto sono *first name*, *Home Directory*, *City* e *Manager's name* di un utente. Alcuni insiemi di proprietà esemplificativi di un oggetto utente sono *Phone and Mail options*, *Account restrictions* o *Personal Information*, *Public Information* e *Home Address*. Un esempio di insieme di proprietà è costituito dal *Public Information*: copre gli attributi *E-mail addresses*, *Manager Common Name* di un utente. Nel *Windows* 2000 *platform* SDK (*Software Development Kit*) viene spiegato come creare degli insiemi di proprietà personalizzati per la propria organizzazione.

I diritti estesi (*Extended Rights*) sono delle speciali azioni o attività correlate agli oggetti *Active Directory*, che non sono coperte da nessuno dei diritti di accesso *Windows* 2000 *standard* (*read*, *write*, *execute*, *delete*). Ciò che li rende così speciali è il fatto che non possono essere collegati alle proprietà degli oggetti. Alcuni validi esempi sono costituiti dai diritti estesi *send as* e *receive as* specifici delle caselle di posta. Anche se i diritti

estesi non sono collegati alle proprietà degli oggetti, vengono visualizzati insieme ai permessi *standard* sull'oggetto nell'**ACL** *editor*. Per ottenere una panoramica degli *Extended Rights*, aprire il container *Extended-rights* del contesto di assegnazione dei nomi di configurazione di *Windows* 2000 *Active Directory* (è possibile esaminare i contenuti di *Active Directory* usando *tool* ADSIEDIT).

Analogamente agli insiemi di proprietà, l'organizzazione può creare ulteriori diritti estesi; come per gli insiemi di proprietà, il modo per configurarli è dettagliato nella documentazione di sistema (*Windows* 2000 *platform* SDK).

Nuove regole di valutazione delle ACL

Tutte le modifiche al controllo dell'accesso che sono state elencate in precedenza hanno forzato *Microsoft* a rivedere le regole e l'ordine di valutazione delle *Discretionary* **ACL**. Nel gergo del sistema, l'ordine di tali regole viene detto ordine canonico e viene visualizzato mediante la vista avanzata dell'**ACL** *editor*.

Questo ordine di valutazione contiene tre regole fondamentali:

- Le impostazioni per il controllo dell'accesso definite in modo esplicito hanno sempre la precedenza sulle impostazioni ereditate. Come è stato indicato in precedenza, questa è una conseguenza diretta del modello di controllo dell'accesso discrezionale usato in *Windows* NT e in *Windows* 2000.
- Le impostazioni per il controllo dell'accesso padre *tier* 1 hanno la precedenza sulle impostazioni per il controllo dell'accesso padre *tier* 2.
- I permessi *deny* hanno la precedenza sui permessi *allow*. In caso contrario, un utente con diritto di accesso *deny* potrebbe ancora essere in grado di accedere a una risorsa sulla base, per esempio, di un **ACE** *allow* per uno dei suoi gruppi.

Auditing

Per quanto alle funzionalità di *auditing* ci si riferisce all'*event viewer* (modificato rispetto alla precedente versione) e l'inclusione del SCTS (*Security Configuration Tool Set*).

L'*event viewer* di *Windows* 2000

Rispetto al suo predecessore in NT 4, l'*Event Viewer* è stato esteso ed include una serie di nuove cartelle per raccogliere le informazioni di *auditing* relative ad alcuni servizi centrali del sistema operativo (come il *Directory Service*, *DNS Server* e il *File Replication Service*). Anche la porzione per la descrizione degli eventi è stata estesa, facilitando la risoluzione dei problemi. Alcuni eventi comprendono inoltre un puntatore HTTP al sito di supporto *on-line* di *Microsoft*. Infine, ora è possibile accedere ai *log* degli eventi usando l'interfaccia di gestione WMI (*Windows Management Instrumentation*).

Due funzionalità che ancora mancano dall'*Event Viewer* di *Windows* 2000 e dal suo sistema per il *log* degli eventi, sono la capacità di impostare il *logging* centralizzato e di archiviare le impostazioni del *log* degli eventi in modo professionale. In questo contesto, il termine *logging* centralizzato indica la raccolta in tempo reale e in un singolo *database* centralizzato di tutti gli eventi di *log* di ogni *computer*. Le

organizzazioni che desiderano tenere traccia con l'andare del tempo dei contenuti dei propri *log* degli eventi devono implementare alcune speciali procedure di archiviazione. Inoltre, per attivare il *logging* centralizzato, sono necessari speciali *script* o specifici prodotti *software* di terze parti.

Il *Security Configuration Tool Set*

Usando l'SCTS, l'amministratore può controllare le impostazioni di sicurezza correnti di un *computer* confrontandole con i valori definiti in un modello di sicurezza. I modelli di sicurezza si possono definire usando lo *snap-in* MMC (*Microsoft Management Console*) *Security Templates*. Questi modelli contengono ogni tipo di parametro legato alla sicurezza di *Windows* 2000: **ACL** su oggetti del registro di configurazione e *File system*, impostazioni dell'*Event Log*, gruppi *Restricted* (per impostare l'appartenenza ai gruppi), impostazioni dei servizi di sistema e impostazioni delle politiche sugli *account* di sistema.

Microsoft mette a disposizione due categorie di base di modelli per la configurazione della sicurezza:

- *default*;
- *incremental*.

I modelli di sicurezza *default* contengono le impostazioni di sicurezza *Windows* 2000 *default*, così come vengono applicate al sistema *Windows* 2000 durante la normale installazione. È possibile usare i modelli di *default* per configurare una macchina aggiornata da NT 4 a *Windows* 2000 al livello delle impostazioni di sicurezza *Windows* 2000 *default*.

I modelli di sicurezza *incremental* definiscono invece dei livelli di sicurezza *higher* o *lower*; è possibile usare questi modelli per portare una macchina dal livello di sicurezza di *default* a un livello di sicurezza più alto o più basso.

Per esempio, il modello compatws.inf riduce la sicurezza su una macchina *Windows* 2000 *Professional*, in modo da consentire alle applicazioni di scrivere su un maggior numero di chiavi del registro di configurazione.

Il modello hisecdc.inf rafforza invece la sicurezza di un *controller* di dominio *Windows* 2000. Un modello *incremental* non dovrebbe essere mai applicato senza aver prima applicato un modello di *default*. *Microsoft* definisce tre livelli:

- *compatible*;
- *secure*;
- *high*.

Usando lo *snap-in* MMC SCTS, è possibile caricare in SCTS i modelli appropriati a seconda delle proprie esigenze di sicurezza. SCTS usa uno speciale *database* (secedit.sdb) per archiviare le informazioni relative al modello di sicurezza che è stato caricato. È possibile eseguire il motore SCTS dallo *snap-in* MMC, oppure dal *prompt* dei comandi (usando l'eseguibile secedit). Lo stesso motore viene usato per la porzione di sicurezza delle impostazioni GPO (*Group Policy Object*) *Windows* 2000. SCTS si usa per definire le impostazioni di sicurezza locali. La porzione di gestione della sicurezza GPO può configurare le impostazioni di sicurezza non locali che vengono definite a livello di dominio *Active Directory*, di unità OU, oppure di sito.

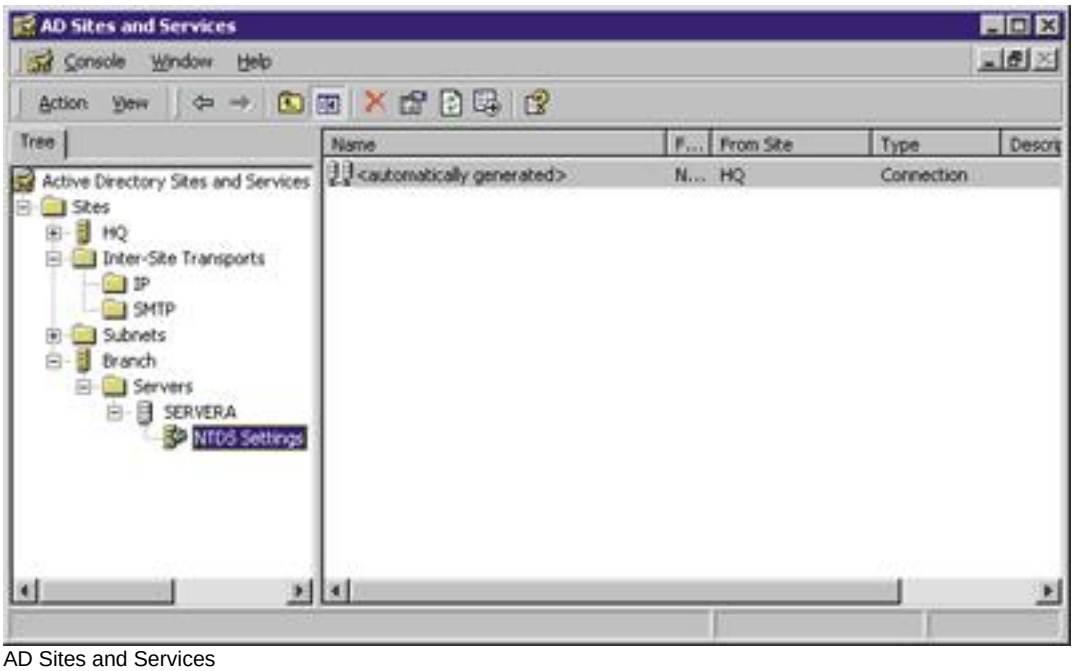
Funzionalità di active directory

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*.

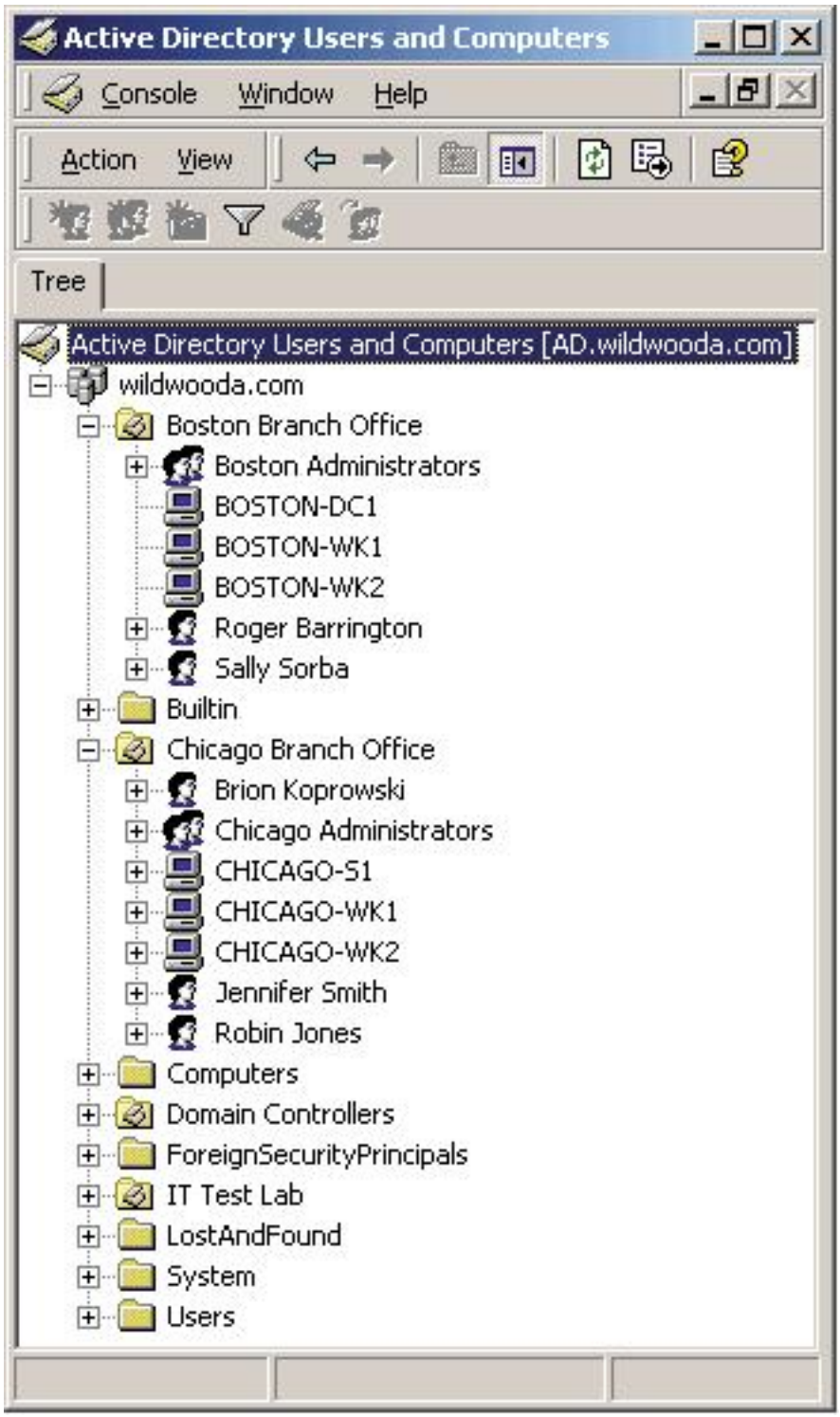
Per *default*, il primo *controller* del primo dominio nel primo albero diventa il *server Global Catalog*. Per specificare manualmente altri *controller* di dominio come *server Global Catalog*, si può usare lo *snap-in* MMC (*Microsoft Management Console*) *Active Directory Sites and Services*.



AD 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 clic 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*.

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.

Struttura dei *controller* di dominio. In NT 4.0 il *controller* PDC è un singolo punto di modifica per il *database* SAM, oltre che un singolo punto di guasti. Come accennato in precedenza, *Windows* 2000 non richiede il *controller* PDC per le modifiche al *database* SAM, dal momento che il sistema operativo supporta la replicazione *multimaster Active Directory*. In ogni caso, il ruolo di *controller* PDC esiste ancora. Le foreste *Windows* 2000 richiedono i seguenti cinque ruoli *Operations Master* sui *controller* di dominio: PDC, **RID** *Pool, Infrastructure, Domain Naming* e Schema (in precedenza *Microsoft* faceva riferimento ai ruoli *Operations Master* usando l'acronimo FSMO - *Flexible Single - Master Operations*).

I ruoli *Operations Master* PDC, **RID** *Pool* e *Infrastructure* devono risiedere su almeno un *controller* in ogni dominio. Se il *server* che contiene un particolare ruolo si guasta, occorre elevare manualmente a questo ruolo un altro *controller* di dominio. Il ruolo PDC è piuttosto intuitivo: se si dispone di *client* e *controller* BDC NT 4.0 *downlevel*, il *controller Windows* 2000 che ottiene il ruolo PDC è il *controller* PDC del dominio. Il ruolo **RID** *Pool* si riferisce al valore **RID** (*Relative IDentifier*) nell'identificatore **SID** (*Security IDentifier*) di un utente. Dal momento che *Windows* 2000 permette a qualsiasi *controller* di dominio di apportare modifiche alla *directory*, ha bisogno di un metodo per coordinare l'assegnazione degli identificatori **RID** ai nuovi oggetti. L'*Operations Master* **RID** *Pool* riveste esattamente questo ruolo. Il ruolo *Infrastructure* è un processo che mantiene la coerenza interdominio tra gli oggetti replicati attraverso i confini del dominio (per esempio, connessioni di replicazione, configurazione del sito, *Global Catalog*).

I ruoli *Operations Master Domain Naming* e Schema risiedono almeno su un *controller* di dominio della foresta. Il ruolo *Domain Naming* garantisce l'univocità dei nomi di dominio all'interno della foresta quando vengono aggiunti nuovi domini. Il ruolo Schema definisce invece quale *controller* di dominio può apportare modifiche allo schema di *directory*, dal momento che il fatto di permettere a più *controller* di compiere modifiche allo schema della *directory* può generare dei problemi.

Migrazione verso *Active Directory*

Il metodo più semplice di migrazione verso *Active Directory* è quello di aggiornare i domini NT 4.0 in loco, ovvero aggiornare il *controller* PDC nel primo dominio *master*. Dopo l'aggiornamento del *controller* PDC, il primo dominio funziona in modalità mista: i *controller* di dominio *Windows* 2000 ospitano *Active Directory* ma, per i rimanenti dispositivi NT 4.0, hanno lo stesso aspetto dei *controller* di dominio NT 4.0. Non appena il primo *controller* PDC viene aggiornato a *Windows* 2000, tutte le *workstation* e i *server Windows* 2000 possono avvantaggiarsi di alcune funzionalità di *Active Directory* (come le unità OU e le *Group Policies*), senza influenzare i rimanenti dispositivi NT 4.0. Ogni successivo dominio NT 4.0 che viene aggiornato entra a far parte della foresta che era stata creata in corrispondenza dell'aggiornamento del primo dominio. Se si usano dei domini delle risorse, sarà necessario aggiornare anche questi; successivamente, si possono spostare le risorse in unità OU all'interno di un altro dominio già esistente - riducendo in questo modo il numero totale di domini (con le unità OU non sono più necessari i domini delle risorse). Dopo la migrazione verso

Windows 2000 di tutti domini, si possono usare dei *tool* di terze parti o le *utility* del *Microsoft Windows* NT *Server* 4.0 *Resource Kit* per consolidarli a seconda delle proprie esigenze amministrative e organizzative.

Un altro metodo per compiere la migrazione verso *Active Directory* è quello di creare da zero l'infrastruttura *Windows* 2000 e usare dei *tool* di terze parti (come DM/*Manager* di *FastLane Technologies*, *DirectMigrate* 2000 di *Entevo*, *OnePoint Domain Administrator* di *Mission Critical Software*), oppure le *utility* del *Resource Kit* (per esempio gli *script* di *history cloning* **SID**) per far migrare uno alla volta i gruppi di utenti. Questo approccio conservatore offre un metodo organizzato per aggiornare a *Windows* 2000 utenti e *computer*, senza necessità di ricorrere a un approccio di tipo tutto-o-niente. La possibilità di iniziare da zero permette di evitare il problema della gestione dell'eredità dell'infrastruttura NT già esistente, quando si cerca di muoversi in avanti. Questa strategia alternativa offre inoltre un'opzione di *back-out*, dal momento che i *tool* di terze parti e le *utility* del *Resource Kit* permettono di ricreare o clonare gli oggetti NT 4.0 nella foresta *Windows* 2000 lasciando intatti gli oggetti NT 4.0 già esistenti.

Componenti dell'infrastruttura PKI

La base dell'infrastruttura PKI di *Microsoft* è costituita dalla **API** di crittografia: la CryptoAPI 2.0. Questa **API** mette disposizione un servizio di crittografia e di gestione dei certificati per la sicurezza a chiave pubblica. Il servizio crittografico compie alcune funzioni, come quelle relative alla generazione delle chiavi, all'*hashing* dei messaggi, alle firme digitali e alla codifica.

Il servizio di gestione dei certificati mette invece a disposizione la gestione e l'archiviazione dei certificati digitali *X*.509v3. L'infrastruttura PKI offerta da *Windows* 2000 è formata da vari componenti:

- i *provider* CSP (*Cryptographic Service Provider*);
- il *Certificate Server*;
- il servizio *smartcard*;
- un canale sicuro;
- il *file system* EFS (*Encrypting File System*);
- il *Microsoft Exchange Server* KM (*Key Management*) *Server*;
- alcune applicazioni che utilizzano l'infrastruttura a chiave pubblica.

Componenti dell'infrastruttura PKI

Il nuovo sistema operativo di *Microsoft* dispone di un'architettura modulare, che consente agli amministratori di aggiornare, integrare, estendere e sviluppare con facilità l'infrastruttura PKI dell'azienda, senza modificare i sottostanti *kernel* di sistema operativo. Per esempio, *Exchange Server* 5.5 utilizza unicamente il suo *server* KM per creare e gestire i certificati *client Exchange Server*. Con il *Service Pack* 1 in *Exchange Server* 5.5, quest'ultimo utilizza *Certificate Server* invece di *Exchange Server* KM *Server* per creare e gestire i propri certificati *client*.

Gli sviluppatori possono creare applicazioni compatibili con l'infrastruttura PKI, che siano basate sui componenti PKI e sulla CryptoAPI forniti da *Microsoft*. Per esempio, è possibile impiegare la CryptoAPI e i certificati digitali per crittografare e autenticare i messaggi delle applicazioni MSMQ (*Microsoft Message Queue Server*). È anche possibile utilizzare selettivamente i vari componenti *Microsoft* PKI, in funzione delle effettive necessità aziendali. Per esempio, se l'azienda ha bisogno di disporre di un sito *Web* sicuro, si può utilizzare *Certificate Server* e la funzione a canale sicuro incorporata in *Internet Information Server* e in *Internet Explorer*.

Esaminiamo ora più da vicino alcuni componenti PKI di *Windows* 2000:

- i *provider* CSP;
- i *Certificate Server*, con i suoi *Certificate Manager*, ed il *tool Certificate Server Manager* e le politiche di certificato da esso generate;
- il servizio ***smartcard*** (di abbiamo disposto apposita sezione);
- la funzione a canale sicuro;
- lo schema ***Authenticode***;
- il *file system* EFS.

Provider CSP

Lo schema CSP (*Cryptographic Service Provider*) implementa alcuni algoritmi e chiavi specifiche, in funzione dei requisiti di sicurezza richiesti (per esempio, la generazione di chiavi pubbliche RSA a 1.024 *bit*, la crittografia RC2 e RC4 a 128 *bit*, lo *standard* DES -

Data Encryption Standard - a 56 *bit*). Le applicazioni possono essere configurate con vari *provider* CSP, a seconda delle esigenze aziendali e delle limitazioni legali nei confronti dell'esportazione al di fuori degli USA degli algoritmi di crittografia. Per esempio, si potrebbe utilizzare il *Microsoft Enhanced Cryptographic Provider* (che supporta le chiavi pubbliche RSA a 1.024 o più *bit*, oltre agli schemi RC2/RC4 a 128 *bit*, DES e 3DES) all'interno del Nord America, e il *Microsoft Base Cryptographic Provider* (che supporta le chiavi pubbliche RSA a 512 *bit* e lo schema RC2/RC4 a 40 *bit*) al di fuori degli USA, in modo da rispettare le leggi sull'esportazione degli algoritmi. Quando è necessario specificare un *provider* CSP all'interno di un'applicazione, è sufficiente scegliere quello desiderato dalla lista dei *provider* CSP installati in NT.

Oltre al *provider* CSP di base e a quello migliorato che sono stati inclusi da *Microsoft* in NT 4.0, *Windows* 2000 ne comprende anche altri, in grado di adattarsi meglio alle varie esigenze di sicurezza - per esempio, gli schemi RSA, DSS (*Digital Signature Standard*), *Diffie-Hellmann* e vari *provider* CSP di base per *smartcard*. Gli sviluppatori di terze parti possono scrivere nuovi *provider* CSP proprietari, che possono essere installati dopo avere ottenuto l'approvazione da parte di *Microsoft* per il loro utilizzo in NT. Per esempio, *Datakey* (una società che offre prodotti *smartcard*) ha reso disponibile un *provider* CSP approvato da *Microsoft* e chiamato *SignaSURE Cryptographic Provider*, che consente di eseguire tutte le principali funzioni di crittografia proprie delle *smartcard*.

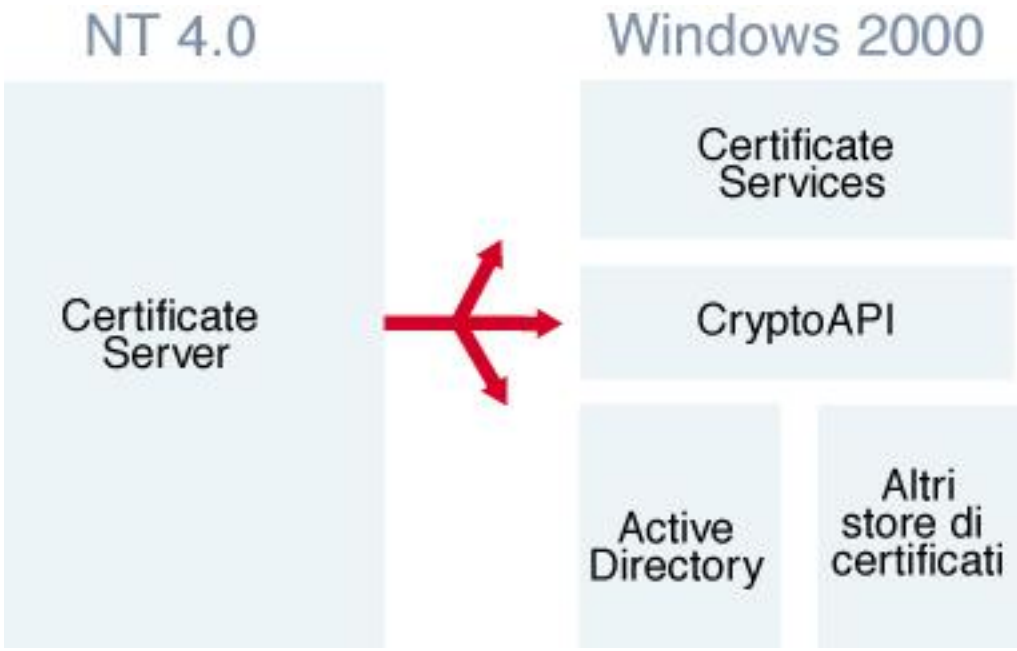
Fisicamente, un *provider* CSP è costituito da un *file* .dll. Anche se la maggior parte è basata sul *software*, è tuttavia possibile implementarli anche nell'*hardware*. Per esempio, un produttore potrebbe implementare il suo *provider* CSP proprietario per *smartcard* e l'adattatore per la lettura di queste ultime, in modo da contribuire a migliorare prestazioni ed efficienza del sistema.

Certificate server

Anche *Certificate Server* è disponibile a partire da NT 4.0; ha sempre fornito le funzionalità di base di CA per richiedere, rilasciare, pubblicare e gestire i certificati. *Certificate Server* offriva l'integrazione S/*MIME* (*Secure Multipurpose Internet Mail Extensions*) per *Exchange Server*, ma successivamente *Microsoft* lo ha orientato soprattutto all'autenticazione *client* basata su chiave pubblica per IIS (*Internet Information Server*). Per controllare la configurazione di *Certificate Server*, si richiedeva agli amministratori delle modifiche dirette su alcuni *file* di testo. *Certificate Server* era privo di alcune funzionalità di gestione particolarmente importanti per l'uso *enterprise* dell'infrastruttura PKI (come i *tool* per personalizzare i tipi di certificato e le impostazioni delle politiche) e offriva il supporto unicamente per le gerarchie CA a due livelli (che risultava inadeguato per l'implementazione PKI su larga scala).

Directory principale console - autorita' di certificazione

In *Windows* 2000, il nome *Certificate Server* è stato leggermente modificato in *Certificate Services*. Questi ultimi sono più potenti e meglio integrati con tutto il resto del sistema operativo. Gli *snap-in* della *consolle* MMC (*Microsoft Management Console*) offrono dei *tool* **GUI** per il lato *client* e per quello *server*. Anche se i *Certificate Services* possono mantenere un proprio *data store* indipendente, per ottenere la completa funzionalità *enterprise* usano *Active Directory* per archiviare e pubblicare i certificati. In questo modo si possono mappare con facilità i certificati sugli utenti e sfruttare le funzionalità di gestione dell'*editor* GPE (*group Policy Editor*) per controllare a chi, per conto di chi e a quale scopo i *Certificate Services* rilasciano i certificati. Infine, i *Certificate Services* ora supportano le gerarchie multilivello.

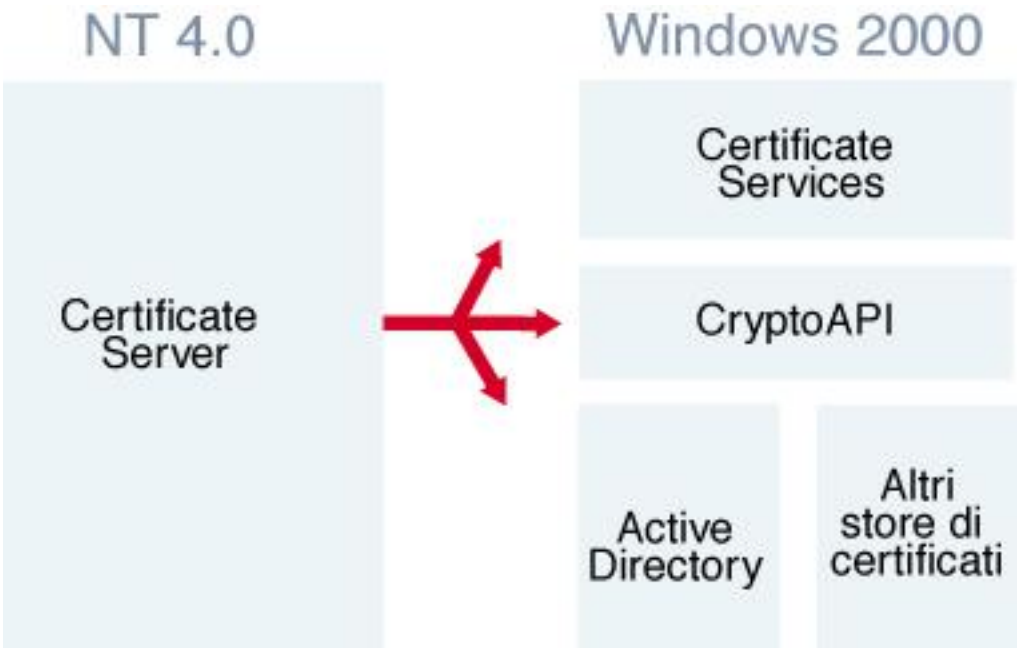


Certificate server

I *Certificate Services* sono diventati molto più leggeri in *Windows* 2000, dal momento che alcune porzioni del loro predecessore per NT 4.0 sono passate ad altri componenti. Per esempio, *Microsoft* ha aggiunto alla CryptoAPI le funzioni per la gestione dei certificati; ha inoltre spostato in *Active Directory* il *data store* dei certificati di *default*. Dal momento che i *Certificate Services* accedono al proprio *store* dei certificati attraverso la CryptoAPI, possono pubblicare i certificati in altre *directory* di terze parti.

Canale sicuro

Windows 2000 incorpora un canale sicuro che supporta il protocollo SSL, il quale viene utilizzato da molte aziende per rendere sicure le comunicazioni *Web* su *Internet*. L'ente IETF (*Internet Engineering Task Force*) ha ratificato lo schema SSL quale *standard Internet*, assegnandogli il nuovo nome di TLS (*Transport Layer Security*).



Canale sicuro

Il canale sicuro messo disposizione da *Microsoft* comprende anche la crittografia SGC (*Server Gated Cryptography*): si tratta di un'estensione dello schema SSL 3.0, che utilizza la codifica a 128 *bit* per rendere sicure le sessioni bancarie *on-line*. *Internet Explorer* 3.0 e le versioni precedenti, oltre a *Internet Information Server* 3.0, supportano lo schema SSL/TLS; *Internet Explorer* 4.0, *Internet Information Server* 4.0 e le versioni successive, oltre a *Money* 98, supportano invece lo schema SGC.

Rispetto a quest'ultimo, lo schema SSL/TLS è costituito da un protocollo generico, che può essere utilizzato per qualsiasi sito *Web*. A causa delle limitazioni insite nelle esportazioni al di fuori degli USA degli algoritmi di crittografia più potenti, lo schema SSL/TLS implementa due versioni di *Internet Explorer*: la prima dispone della crittografia a 128 *bit* per il Nord America, mentre la seconda utilizza la codifica unicamente a 40 *bit* per l'uso internazionale. Le banche e le istituzioni finanziarie hanno tuttavia bisogno di utilizzare una crittografia particolarmente sofisticata per proteggere i clienti che utilizzano l'accesso *on-line* tramite il *Web*; per questo motivo, il governo degli Stati Uniti consente alle organizzazioni finanziarie internazionali di utilizzare la crittografia SGC tra gli USA e praticamente qualsiasi altro paese del mondo. *Microsoft* ha incorporato questo protocollo nelle versioni 4.0 e 5.0 del *browser Internet Explorer*, a 40 e 128 *bit*. Il *browser* può utilizzare la funzione SGC per la crittografia a 128 *bit* soltanto se il *server Web* con cui è in comunicazione è a sua volta dotato di un certificato *server* SGC. Ricordiamo l'interfaccia SSPI (*Security Support Provider Interface*) e la CryptoAPI. Le **API** di *Windows* 2000 permettono agli sviluppatori di riutilizzare i servizi (per esempio quelli crittografici) forniti dal sistema operativo. Il fatto di essere in grado di astrarre le applicazioni dal *provider* permette inoltre di proteggerle dall'obsolescenza. È possibile aggiornare e migliorare i componenti *provider* di pari passo con l'evoluzione della tecnologia, senza per questo influenzare l'applicazione. Per esempio, un'applicazione che usa il *provider* CSP può sfruttare rapidamente l'algoritmo di crittografia di un nuovo *standard*.

Esaminiamo ora il modo in cui gli schemi SSL/TLS e SGC utilizzano i certificati e le

chiavi pubbliche al fine di instaurare un canale sicuro tra *client* e *server*. Nello *standard* SSL/TLS, il *client* invia per prima cosa un messaggio di *hallo* al *server*, mentre quest'ultimo saluta il *client* inviandogli il proprio certificato *server*. Il *client* autentica il *server* utilizzando la chiave pubblica dell'autorità CA (che viene estratta dal certificato CA archiviato nel *client*), per controllare la firma di quest'ultima nel certificato del *server*. Il *server* può a sua volta autenticare opzionalmente il *client*, chiedendogli e verificando il suo certificato (in *Internet Information Server* 3.0 e nelle versioni successive, gli amministratori possono fare sì che il *server* autentichi il *client*). Quando il *client* ha autenticato il *server*, genera una chiave di sessione a 40 o 128 *bit*, a seconda delle limitazioni per la sicurezza. Il *client* crittografa quindi la chiave di sessione usando la chiave pubblica del *server* (estratta dal certificato di quest'ultimo), quindi gliela invia. Il *server* la decodifica quindi utilizzando la propria chiave privata. A questo punto, il *server* e il *client* possono scambiarsi i dati crittografati con questa chiave di sessione.

Con lo schema SSL/TLS, è possibile usare i certificati rilasciati da *Certificate Server* di *Windows* 2000 oppure da altre autorità CA. Per utilizzare lo *standard* SGC, è invece necessario richiedere un certificato *server* SGC a un'autorità CA autorizzata (per esempio a *VeriSign*), la quale esaminerà i requisiti del richiedente. Il protocollo SGC utilizza una sequenza di *handshake* simile a quella adottata dallo schema SSL/TLS per impostare una sessione sicura, ma che differisce tuttavia da quest'ultima sotto due aspetti. Per prima cosa, con lo *standard* SGC il *client* resetta e fa ripartire la sequenza di *handshake* dopo il primo scambio di *hello*, quando rileva che il certificato del *server* è di tipo SGC. Secondariamente, il *client* genera sempre una chiave di sessione a 128 *bit* dopo aver resettato la sequenza di *handshake* e l'autenticazione del *server*.

Authenticode

L'infrastruttura PKI di *Microsoft* mette a disposizione una tecnologia chiamata *Authenticode* per la firma del codice, che assicura l'integrità e l'origine del *software* commerciale e di quello gratuito distribuito su *Internet*. Lo schema *Authenticode* è basato sulla tecnologia a firma digitale; aggiunge una firma digitale, un certificato di firma del codice e un contrassegno dell'ora e della data al codice che caratterizza il *software* come quello di *applet Java*, controlli *ActiveX*, *file* .dll, *file* eseguibili, *file cabinet* e *file catalog*. Questo schema consente anche di verificare il codice scaricato prima di eseguirlo sul proprio sistema. Per firmare e verificare il codice, il sistema *Authenticode* utilizza due tecniche differenti: la firma del codice e la sua verifica.

Prima di poter firmare il codice per garantirne l'integrità e l'origine, è necessario ottenere da un'autorità CA un certificato per la firma del codice o per la pubblicazione del *software*. Dopo questa operazione, per firmare il codice si potranno usare le funzioni di firma *Authenticode* offerte dal kit SDK *ActiveX*. Il codice che si desidera autenticare viene passato attraverso un algoritmo di *hashing*, mentre la propria chiave privata viene usata per firmare l'*hash* ottenendo una firma digitale. A questo punto viene creato il blocco della firma, che contiene la firma digitale e il certificato per la firma del codice. Lo schema *Authenticode* consente di contrassegnare il blocco della firma con l'ora e la data, in base ai dati che vengono forniti da un *provider* di servizi di data e ora, come per esempio *VeriSign*. Infine, il blocco della firma viene legato al *software* originale; a questo punto è possibile pubblicare sul proprio sito *Web* il *software* firmato, rendendolo disponibile per il *download*.

Il kit SDK *ActiveX* mette disposizione anche una funzione dedicata alla verifica del codice, che consente di verificare il *software* scaricato prima di eseguirlo. *Microsoft* ha incorporato questa funzione in *Internet Explorer* 3.0 e nelle versioni successive. Prima che *Internet Explorer* esegua il codice firmato, viene chiamata questa funzione per la verifica del codice che controlla la firma, il certificato dell'editore e la data e l'ora dell'autenticazione. Dopo questa verifica, la funzione visualizza il nome del codice, quello dell'organizzazione o della persona che lo ha pubblicato, la data in cui l'editore lo ha autenticato e il nome dell'autorità CA che ha fornito il certificato per la firma del codice. L'utente a questo punto può decidere di accordare la fiducia all'editore.

Usando *Internet Explorer*, è possibile impostare una politica di sicurezza per *Authenticode*, caratterizzata da quattro diversi livelli di protezione: alto, medio, basso e personalizzato. Il livello più elevato fa sì che non venga eseguito il codice danneggiato; quello medio avverte l'utente prima di eseguire del codice potenzialmente danneggiato; il livello basso esegue in ogni caso qualsiasi tipo di codice, mentre quello personalizzato consente di scegliere le impostazioni di sicurezza: per esempio attivare il codice *ActiveX*, oppure disattivare le *applet Java*. È anche possibile definire diversi livelli relativi a zone di sicurezza differenti: per esempio *Internet*, intranet, siti affidabili e siti ad accesso limitato di *Internet Explorer*.

prodotti per la gestione di rete

Franco Callegati
Walter Cerroni
9.1.6 (Effettuare procedure di backup e recovery e controllo)

Introduzione

Il mercato oggi propone numerosi prodotti di *System & Network Management*. Tra le famiglie più complete troviamo:

- *OpenView* di *Hewlett Packard*;
- *Tivoli* di *IBM*;
- SMS di *Microsoft*.

Tutti questi prodotti sono disponibili sulle principali piattaforme *Unix* e su *Windows* NT.

Altri produttori non presentano soluzioni complete come quelle già citate, ma si specializzano su particolari funzioni o piattaforme. Tra questi citiamo *EcoSystem* di *CompuWare*, *BMC Patrol* di *BMC Software, Microsoft Management Console*, *AppManager* di *NetIQ*, *MainView* e *SpaceView* di *Boole & Babbage*, *TransView* di *Siemens Nixdorf*.

La scelta non è ovviamente facile, poiché i criteri sono dipendenti dalle reali necessità dell'azienda. In genere, se c'è necessità di una gestione trasversale che interessi diverse aree, gli utenti più maturi scelgono una *suite* integrata piuttosto di un insieme di prodotti specializzati in diverse nicchie. Altre funzioni determinanti nella scelta del prodotto sono la capacità di gestire più piattaforme, la disponibilità di una interfaccia grafica versatile e, possibilmente, accessibile in remoto via *Web*.

Lo strumento di *System & Network Management* deve inoltre essere poco invasivo e, quindi, generare poco traffico di rete, ma deve essere in grado di lavorare anche su reti geografiche.

System Management Server 2.0 (1/2)

Fin dal suo debutto nel 1994, *System Management Server* (SMS) ha cercato di implementare il ZAW (*Zero Administration for Windows*), ossia un *framework* di applicazioni e tecnologie progettate con l'intenzione di ridurre i costi di gestione del parco macchine aziendale. Col passare degli anni il prodotto ha fatto strada includendo funzioni di *hardware* e *software inventory*, *software metering* e *deployment*, nonché strumenti per il *troubleshooting* di postazioni remote.

Indubbiamente il prodotto è di un buon livello, ma sono solo un paio le aree in cui eccelle davvero. Per esempio l'*Installer* di SMS è di gran lunga il componente più flessibile rispetto a tutti moduli di installazione dei concorrenti. Oltre a presentare le opzioni con un'interfaccia utente gradevole, è davvero apprezzabile la possibilità di provare le installazioni per controllare tutte le modifiche che lo *script* di installazione apporta al sistema. Inoltre, se qualcosa durante il processo di installazione non dovesse andare a buon fine, è sempre possibile effettuare un *rollback*, ossia un ripristino del *client* che lo riporta esattamente nella configurazione (librerie, chiavi di registro, impostazioni) appena precedente l'installazione.

Le aree in cui il prodotto *Microsoft* davvero eccelle sono quelle della gestione delle informazioni di sistema e la reportistica. Grazie alla stretta integrazione con *SQL server* e *Crystal Report* (fornito insieme a SMS), i *tool* messi a disposizione dal *software* permettono all'utente di avere il pieno controllo sulle informazioni relative ai *client*: è infatti possibile creare delle interrogazioni personalizzate presentandole con aspetto professionale grazie alla flessibilità di *Crystal Report*. Inoltre con il motore SQL è possibile creare dei gruppi (si tratta di *query* sul *database* mantenuto dalle informazioni del modulo di *inventory*) di macchine e visualizzarli nella vista ad albero della console di gestione. Questa caratteristica aumenta notevolmente la flessibilità del sistema, in quanto permette di accedere ai *desktop* non solo in base a come il sistema scopre i *client* e li visualizza nell'elenco delle *workstation* di rete, ma anche in maniera logica e organica, secondo criteri che possono essere di carattere organizzativo o di omogeneità di *hardware*.

Al pari di *Intel LANDesk*, anche SMS dispone di un ottimo strumento per la diagnosi in remoto dei PC e per verificarne lo stato di salute: mentre un *desktop* sta lavorando è possibile interrogarlo sulle impostazioni del BIOS, sulla quantità di memoria disponibile, sul numero di programmi attivi in quel momento e su tutto ciò che può servire durante un'attività di supporto tecnico.

All'interno delle *utility* per la gestione e il *monitoring* dei *server* figura anche *Network Trace*, un utilissimo strumento in grado di supportare gli amministratori di rete in quanto è in grado di produrre uno schema grafico della rete includendo *workstation*, stampanti, *router* e tutti gli oggetti connessi.

Indubbiamente l'arrivo di *Windows* 2000 ha completato e potenziato le caratteristiche di SMS, dotandolo di un sistema di *directory* potente e professionale, ma alcune peculiarità, come l'installazione dei *driver* di stampa o la protezione antivirus con un modulo *software* dedicato, permetterebbero a *Microsoft* di avere un prodotto davvero da primo della classe.

System Management Server 2.0 (2/2)

La gestione dei PC *client* viene garantita da alcuni moduli che raccolgono via rete le informazioni sulla configurazione *hardware* e la dotazione *software* delle singole postazioni, eseguendo anche alcuni controlli di compatibilità dell'installato con, per esempio, l'adeguamento all'euro o con *Windows* 2000. Le applicazioni presenti sui PC vengono monitorate per stabilirne il profilo di utilizzo (chi le usa, per quanto tempo e in quali momenti della giornata); sulla base di queste informazioni l'amministratore di rete può pianificare eventuali aggiornamenti o l'acquisto di nuove licenze.

La funzione *Network Discovery* rileva inoltre la topologia della rete e il tipo di *client* e sistemi operativi che la popolano. SMS è inoltre dotato di molti report predefiniti in base alle esigenze più comuni. Uno dei punti di forza del *tool Microsoft* è rappresentato dalle funzioni di installazione remota del *software* a partire da un *server* centrale: sfruttando le informazioni memorizzate nell'inventario, per esempio, SMS può stabilire automaticamente su quali macchine non è possibile installare una determinata applicazione.

L'installazione e la rimozione di applicazioni dai PC possono seguire regole e procedure definite dall'amministratore di rete e agire dinamicamente, seguendo cioè i cambiamenti di stato del singolo utente; questi, per esempio, può a un certo punto non

appartenere più al gruppo che utilizza una particolare applicazione, la quale va di conseguenza rimossa dal PC. Le installazioni possono essere programmate in anticipo ed essere effettuate automaticamente nei momenti di minor traffico per la rete, in maniera trasparente per l'utente.

Dal punto di vista della diagnostica, SMS è in grado di rilevare alcuni problemi della rete (come ad esempio la duplicazione di indirizzi *IP* o la presenza di protocolli non desiderati) e può semplificare la risoluzione dei problemi al *desktop* perché l'amministratore di rete può impadronirsi completamente del controllo di un PC remoto. SMS raccoglie e visualizza anche i parametri fondamentali di funzionamento dei *server* più importanti e dei processi di *Windows* NT *Server* e di *Microsoft BackOffice*.

Introduzione OpenView

OpenView è un insieme di strumenti specializzati e in grado di interoperare e di coprire le seguenti aree di *management*: applicazioni e sistemi, rete, sicurezza, ambiente NT, *desktop* e *software*, *storage* e infine IT *Service Level*.

Il punto focale di *OpenView* è certamente il *business*: infatti il prodotto di punta della *suite*, che può fungere da cappello per tutti gli altri *tool*, è stato progettato per indirizzare le persone, i processi e la tecnologia richiesti per un'interazione di successo fra l'organizzazione dell'IT e il *business*. La finalità è di consentire ai membri dell'*Information Technology* delle imprese di fornire servizi di supporto agli utenti, seguendo regole di qualità e di costi.

Il nuovo approccio coniato da *HP* si definisce a *building block*. La sua finalità è riuscire a realizzare in tempi brevi la soluzione per il cliente, tagliata su misura, senza richiedere enormi investimenti iniziali. Con l'architettura *building block* è possibile avviare velocemente la struttura di gestione, partendo ad esempio dal modulo di *network management*, per poi espanderla gradualmente, aggiungendo in successione i moduli d'interesse per il *business* dell'azienda. Gli strumenti di *OpenView* spaziano in tutte le aree di supporto al *business*, soddisfacendo le necessità delle imprese con l'ausilio delle tecnologie DCOM, ActiveX, ITIL (*Information Technology Infrastructure Library*) e *Java*.

Nella piattaforma *OpenView* possono coesistere i *tool* di *network management* (*Network Node Manager*, NNM), *system management* (IT/*Operations*) e molti altri ancora, come ad esempio:

- *Desktop Administrator* - per controllare l'assetto dei pc e ridurre i costi e le attività associate alle funzioni di amministrazione;
- IT *Service Manager* - per controllare la qualità dei servizi *mission-critical* con l'automatizzazione dei processi di gestione;
- IT/*Administration* - per fornire un'accurata visione e un pieno controllo dei sistemi gestiti attraverso l'inventario, la distribuzione del *software* e le configurazioni;
- *ManageX* - per assicurare la disponibilità e prestazioni ottimali dei sistemi e delle applicazioni NT.

Inoltre sono disponibili moduli che risolvono problematiche specifiche delle rete, come *NetMetrix* per *performance reporting*, *monitoring* e analisi; oppure sistemi come gli *SMART plug-in* per la gestione dei *database* o delle applicazioni *Baan*, *Oracle* e SAP

R/3.

HP openview

HP OpenView punta molto sulla crescente diffusione di *Windows* NT *Server*, perciò ci sono nuovi prodotti e servizi di *marketing*, supporto e istruzione rivolti ai partner per sfruttare le opportunità del mercato. Le novità in quest'area riguardano *ManageX* 4.0, il prodotto per la gestione delle applicazioni e dei *server* per ambienti NT, *Back-Office* e *NetWare*. Il *software* è stato potenziato per consentire una migliore gestione proattiva dell'ambiente. Vi sono ulteriori miglioramenti nell'utilizzo dell'architettura WBEM/ WMI (*Web-based Enterprise Management*) e della *Microsoft Management Console*, che rendono possibile l'integrazione con *Microsoft* SMS 2.0 e la coesistenza di entrambi i *software* in una singola stazione.

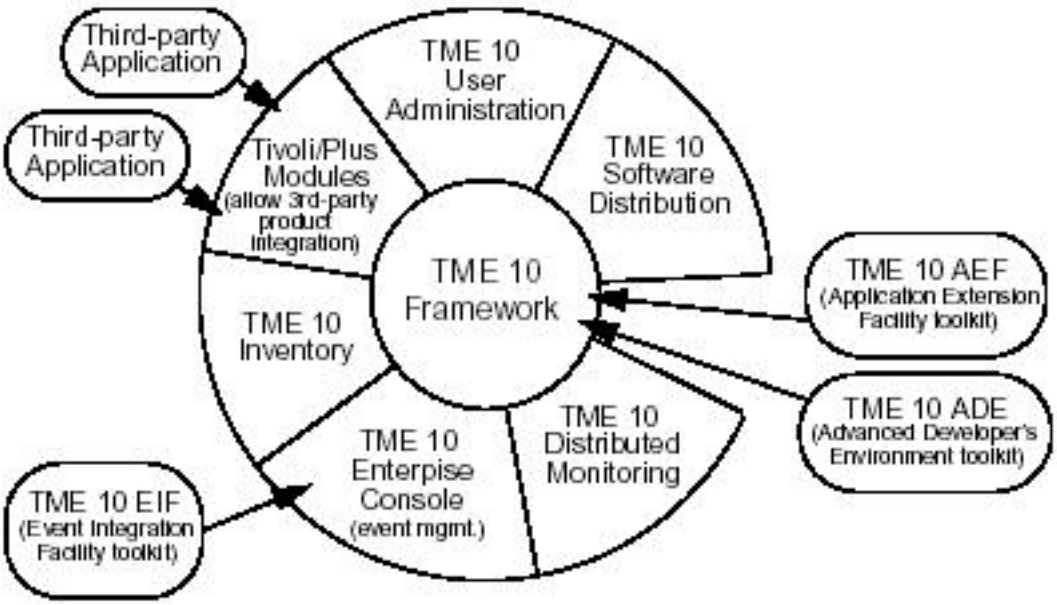
Sono stati anche annunciati prodotti per il *network management*, per la pianificazione della rete e per il supporto degli ambienti eterogenei: questi prodotti andranno a integrare le funzionalità di *NetMetrix* e *NetWork Node Manager*. Infine citiamo *HP OpenView Application Quality of Service*, la nuova strategia che consente agli IT *manager* di avere maggior controllo ed efficienza dell'ambiente per soddisfare le richieste degli utenti, garantendo la continuità dei servizi che sostengono il *business* dell'azienda.

OpenView è una soluzione concepita per supportare il settore dell'IT in ogni processo di *business*, con un occhio di riguardo per gli ambienti *mission-critical*. L'approccio *building block* ha la funzione di dare alta scalabilità e offre la possibilità alle imprese di partire con un minimo impegno economico, verificando in breve tempo il ritorno sugli investimenti. Utilizza *standard* tecnologici affermati che garantiscono un continuo aggiornamento degli strumenti e consentono di interoperare con altre piattaforme, oltre che di offrire ai partner la possibilità di sviluppare soluzioni ad hoc.

Introduzione Tivoli Enterprise (1/3)

La *suite Tivoli Enterprise* è costituita da applicazioni specializzate che, estendendosi dall'S/390 fino al *laptop*, permettono di gestire i sistemi eterogenei e distribuiti come un'unica architettura integrata.

Le soluzioni *Tivoli Enterprise* si basano su un'architettura comune, *Tivoli Management Framework* (TMF), composta da oggetti conformi allo *standard* OMG/CORBA (*Object Management Group/ Common Object Requester Broker Architecture*) che svolgono le funzioni di *manager*, *broker* e agenti, abilitando i servizi di base utilizzati dalle applicazioni di *systems management*. Le **API** sviluppate da *Tivoli* sono state tra l'altro adottate da due importanti organismi di settore, *Open Group* e DMTF, come basi per un possibile *standard* di *network* e *system management*.



Lo spettro di azione dei prodotti di *Tivoli* è particolarmente ampio e abbraccia le seguenti aree:

- Amministrazione - permette la gestione del *software* (inventario, distribuzione) e delle configurazioni degli elementi della rete da una singola locazione centrale attraverso *Tivoli Inventory* e *Tivoli Software Distribution*.
- *Availability* - assicura per gli utenti la piena disponibilità delle risorse di rete e delle applicazioni *mission-critical*. La disponibilità è garantita dall'interazione di alcuni prodotti fra cui *Tivoli NetView* per esplorare le reti *TCP/IP*, visualizzarne le topologie ed effettuare un *monitoring* proattivo delle risorse; *Tivoli NetView Performance Monitor* per effettuare il *monitoring* delle prestazioni e individuare gli eventuali *fault*; *Tivoli Enterprise Console* per raccogliere e correlare le informazioni al fine di diagnosticare i problemi e automatizzare conseguenti operazioni; *Tivoli Reporter* per realizzare report personalizzati.
- Sicurezza - fornisce la protezione di applicazioni *mission-critical*, con il completo controllo degli *account* utente e dei servizi di sicurezza dei diversi sistemi. I prodotti *Tivoli Security Management* e *Tivoli User Management*, fra gli altri disponibili, forniscono la gestione della sicurezza *cross-platform* oltre alla gestione degli utenti e dei gruppi.
- Servizio - i *tool Tivoli Service* Desk, *Tivoli* Decision *Support* e *Tivoli Information Management* garantiscono la continuità del servizio, effettuando dalla console centrale il controllo dei processi, la pianificazione delle attività e il supporto alle decisioni.
- *Operation* - grazie ai moduli ad hoc, è possibile automatizzare le funzioni amministrative di *routine* come ad esempio la programmazione dei *job*, i salvataggi e la supervisione dei sistemi remoti.
- *Application* - la gestione dei *database*, del *server Lotus Notes*, di *Microsoft Exchange*, di *IBM MQ Series*, di SAP R/3 e di altri *software* viene realizzata

attraverso prodotti sviluppati da *Tivoli* e da numerosi suoi partner, sempre basandosi sul *framework* di *Tivoli Enterprise*.

Introduzione Tivoli Enterprise (2/3)

Mentre *Tivoli Enterprise* si rivolge alle grandi organizzazioni, le aziende di medie dimensioni necessitano di strumenti in grado di garantire procedure snelle e rapidi ritorni degli investimenti effettuati in soluzioni di *system management*. Per queste necessità le sei differenti *Tivoli Management Suite* offrono a queste imprese un'ampia gamma di strumenti per il controllo operativo, per l'*asset management* e per la gestione degli applicativi. Le *Tivoli Management Suite* insieme alle risorse dei partner *Team Tivoli* consentono di raggiungere un compromesso tra l'evoluzione delle necessità di *business* e la realtà di una realizzazione IT sostenibile.

Infine l'offerta *Tivoli* non tralascia la piccola impresa, per la quale è stato realizzato *Tivoli IT Director*, un pacchetto che consente di gestire gli ambienti IT di dimensioni più contenute.

Sul fronte dello *storage management*, la *Information Integrity Initiative* di *Tivoli* permette di stabilire *policy* di *storage* e metodologie gestionali unificate, offrendo alle aziende un supporto di altissimo livello per affrontare le nuove problematiche di *storage management* emerse con la *networked economy*.

L'integrazione del sistema *Tivoli Storage Management* all'interno di *Tivoli Enterprise* offre alle aziende l'opportunità di integrare le metodologie di gestione con le politiche di *storage management* tramite una soluzione unificata per i sistemi, l'archiviazione, le reti, le *Storage Area Network* (SAN), i dati e le applicazioni.

I *service provider* sono protagonisti della *e-conomy* e in questo senso *Net Generation* è un'iniziativa supportata da clienti e partner di *Tivoli* che si concretizza in una serie di soluzioni e servizi orientati alle specifiche esigenze di chi offre servizi IT. Al centro dell'iniziativa *Net Generation* si trova la famiglia di soluzioni *Tivoli Service Provider*, che permette di sviluppare rapidamente i nuovi servizi ad alto valore aggiunto, riducendo i costi di realizzazione e gestione. I nuovi prodotti *Tivoli* per la gestione della sicurezza, indispensabile quando le reti aziendali si aprono all'esterno, si basano sulla tecnologia *Tivoli SecureWay Policy Director* che permette di definire filtri per l'accesso alle applicazioni *e-business*, con l'integrazione delle soluzioni *Tivoli* per la gestione di sistemi complessi. La visione sulle problematiche dell'impresa e i processi di *business* è ampia, e sono presente *tool* proprietari e di terze parti.

Tivoli è concepito per rispondere alle esigenze delle diverse fasce di utenza, dalla grande organizzazione alla piccola e *media* impresa, con soluzioni specifiche e complete.

NetGeneration

Una soluzione dedicata ad una specifica area di *business* come la sicurezza dei sistemi, la gestione dei livelli di servizio, l'operatività quotidiana e la disponibilità dei sistemi viene fornita da *Tivoli* con l'iniziativa *NetGeneration*, dedicata ai *service provider*, con la quale *Tivoli* intende offrire numerosi servizi aggiuntivi da affiancare a quelli già disponibili per i clienti dell'area *large account* (per intenderci, da 5 mila postazioni in su).

Le nuove soluzioni rappresentano un naturale ampliamento dei prodotti di fascia *enterprise* e aiutano a massimizzare la qualità dei servizi offerti alla clientela minimizzandone i costi, oltre a introdurre elementi di differenziazione rispetto alla concorrenza. Il *provider* può con questi strumenti amministrare direttamente qualsiasi risorsa del cliente, purché sia *Tivoli-Ready*. Attualmente esistono molti dispositivi con agente *Tivoli* incorporato e il loro numero è in continuo aumento, grazie alle *partnership* strette da *Tivoli* negli ultimi tempi. Si tratta di accordi di integrazione e sviluppo stretti con le più note società di informatica e telecomunicazioni tra cui *Cisco*, 3Com, *Intel*, Telcordia, *Newbridge Networks* e *Nortel Networks*. Con la filosofia *NetGeneration* è stata inoltre presentata la famiglia di soluzioni *Tivoli Service Provider*, basata su un'architettura *standard* flessibile e conforme agli *standard* **SNMP**, TMN e CORBA, alla quale appartengono attualmente i seguenti prodotti:

- *Tivoli for Managed Network Services*, che assicura la gestione della rete *IP* (per eventi, configurazione e attività) e permette l'attivazione dei nuovi servizi o la gestione dei livelli di servizio, dell'inventario e della sicurezza.
- *Tivoli for Managed Computer Devices*, per gestire direttamente anche centinaia di migliaia di dispositivi (quali *desktop, server* o *router*) installati presso i clienti.
- *Tivoli for Managed Application Services*, per l'amministrazione dei servizi applicativi attraverso i *Service Level Agreement*, la visione globale della topologia della rete e dei *database*, la distribuzione del *software*, il *service desk*, la gestione dello *storage* e dell'*e-business*.
- *Tivoli for Hosted Business Services*, per un'efficace gestione dell'accesso ai dati, alle applicazioni, ai sistemi e alle reti in un'architettura orientata al *Web*; con questo strumento si facilitano i *provider* nella creazione di portali e di reti *Extranet* collaborative.
- *Tivoli Subscription Manager*, che consente di registrare, autenticare e servire gli utenti indipendentemente dal tipo di dispositivi che utilizzano per connettersi al servizio, come le apparecchiature cosiddette pervasive: cellulari, videotelefoni e *palm-top*.

L'iniziativa *NetGeneration* orientata ai clienti *enterprise* non deve comunque far pensare che *Tivoli* trascuri l'area *small-medium business*: Maurizio Carli ha infatti evidenziato che gli investimenti sono costanti per le soluzioni dedicate a entrambe le aree. *IT Director*, in particolare, sta dando a *Tivoli* notevoli soddisfazioni come soluzione per le imprese con meno di mille postazioni di lavoro, le quali dispongono di un *budget* contenuto e desiderano risultati immediati.

Autenticazione con SMART CARD in WINDOWS

Franco Callegati
Walter Cerroni

9.1.5 (Usare un sistema di account su una rete) - 9.1.7 (Discutere gli aspetti connessi con le varie tecniche di autenticazione degli utenti)

Smart Card

smart card

Una *smartcard* è un dispositivo che ha le dimensioni di una carta di credito, è protetto da una *password*, ed integra un processore a 8 *bit*, un coprocessore crittografico e un sistema per l'archiviazione locale. La *card* utilizza un sistema operativo dedicato che risiede all'interno di un *chip* ROM con capacità compresa tra 6 e 24 *Kbyte*. Un *chip* EPROM (*Erasable Programmable Read Only Memory*) con capacità compresa tra 1 e 16 *Kbyte* consente invece di memorizzare una quantità limitata di dati dell'utente, per esempio i certificati e le chiavi private. Tali *chip* EPROM dispongono di una limitata quantità di RAM per i dati di *runtime*, compresa tra 128 e 512 *byte*.

È possibile archiviare in una *smartcard* un certificato rilasciato da *Certificate Server*. Successivamente, se al *computer* viene collegato un lettore di *smartcard* si potrà accedere al certificato. La portabilità delle *smartcard* rende possibile usare il certificato su qualsiasi *computer* che sia dotato di un apposito lettore.

La diffusione di un modello *standard* che determina il modo in cui i lettori e le *smart card* si interfacciano con un *computer* promuove l'interoperabilità tra *smart card* e lettori di diversi produttori. Nel passato, la mancanza di interoperabilità è stata una delle ragioni principali della lenta diffusione delle *smart card* al di fuori dell'Europa.

Lo *standard* principale nel settore dell'interoperabilità tra lettori e *smart card* è lo *standard* ISO 7816 per le *smart card* a circuito integrato con contatto. Queste specifiche hanno come obiettivo l'interoperabilità a livello fisico, elettrico e di protocollo per il collegamento dati. Questi *standard* sono stati incorporati nelle seguenti iniziative:

- *Europay*, MasterCard e VISA (EMV). Nel 1996, l'iniziativa EMV ha definito una specifica per *smart card* basata sullo *standard* ISO 7816, destinata soprattutto al settore dei servizi finanziari.
- Sistema GSM (*Global System for Mobile Communications*). Il settore europeo delle telecomunicazioni ha adottato gli *standard* ISO 7816 per le

specifiche delle *smart card* per abilitare l'identificazione e l'autenticazione degli utenti di telefonia mobile.

Sebbene queste specifiche abbiano rappresentato un passo nella giusta direzione, ognuna di esse rappresentava un livello eccessivamente basso o era troppo specifica rispetto alle applicazioni per poter ottenere un supporto più ampio e diffuso e pertanto non è stato possibile risolvere le problematiche legate all'interoperabilità delle applicazioni. Nel 1996 è stato costituito il gruppo di lavoro PC/SC, formato da aziende del settore informatico e delle *smart card*, tra cui *Microsoft*, *Hewlett-Packard*, *Schlumberger* e *Gemplus*, proprio al fine di sviluppare specifiche volte alla soluzione di tali problematiche.

La versione 1.0 della specifica è stata pubblicata nel dicembre del 1997 e ha ottenuto un ampio supporto nel settore. *Microsoft* ha partecipato rendendo disponibili i componenti di base per *smart card*, scaricabili gratuitamente dal *Web* e validi per i sistemi operativi *Microsoft Windows* 95, *Microsoft Windows* 98, *Microsoft Windows Millennium Edition* (Me) e *Microsoft Windows* NT 4.0. Tali componenti sono inclusi in *Windows* 2000.

Componenti base 1

Componenti *software*

Le interfacce **API** indipendenti dalle periferiche consentono agli sviluppatori di applicazioni di non occuparsi delle differenze tra le realizzazioni attuali e quelle future. Dal punto di vista di uno sviluppatore di applicazioni, esistono tre metodi di programmazione per le *smart card*:

- **API** *Microsoft Win32*.
- Crypto**API**.
- SCard COM.

Il metodo scelto dipende dal tipo di applicazione e dalle capacità di una *smart card* specifica.

Win32

Le interfacce **API** *Win32* sono quelle di livello base per l'accesso alle *smart card*. Per un efficace utilizzo di tali interfacce **API**, è necessaria un'approfondita conoscenza del sistema operativo *Windows* e delle *smart card*. Tali interfacce offrono all'applicazione una grande flessibilità per il controllo di lettori, *smart card* e altri componenti correlati. Per gli sviluppatori che intendono avere il massimo controllo sul modo in cui un'applicazione utilizza le *smart card*, l'estensione all'**API** *Win32* di base rende disponibili le interfacce necessarie per la gestione dell'interazione con le periferiche per *smart card*.

Crypto**API**

Crypto**API** è l'interfaccia **API** crittografica di *Microsoft*. È concepita in modo da potersi astrarre dai dettagli della funzione di crittografia, ad esempio dagli algoritmi di crittografia, in modo che le applicazioni possano utilizzare una crittografia collegabile. Per ottenere ciò, le **API** vengono sovrapposte a moduli crittografici sostituibili detti *provider* CSP (*Cryptographic Service Provider*). I *provider* CSP possono essere prodotti basati solo sul *software*, oppure far parte di una soluzione *hardware* nella

quale il motore crittografico risiede in una *smart card* o in un altro elemento *hardware* collegato al *computer*.

Nel modello *Microsoft* per l'accesso alle *smart card*, il *provider* CSP per *smart card* è associato a un tipo specifico di *smart card*, rendendo così disponibile l'associazione tra le funzioni crittografiche esposte tramite l'interfaccia Crypto**API** e i comandi di basso livello accessibili tramite le interfacce **API** *Win32* per *smart card*. Pertanto, il *provider* CSP è in grado di indicare alla *smart card* come portare a termine specifiche operazioni di crittografia. In *Windows* 2000, *Microsoft* ha integrato due *provider* CSP che supportano diverse *smart card* prodotte da *Gemplus* e da *Schlumberger*. Altri fornitori hanno sviluppato *provider* CSP specifici per le proprie *smart card*.

SCard COM

SCard COM è un'interfaccia non crittografica resa disponibile da *Microsoft* per accedere a servizi generici basati su *smart card* utilizzando applicazioni scritte in linguaggi diversi, ad esempio C, *Microsoft Visual* C++, *Java* e *Microsoft Visual Basic*.

L'interfaccia SCard COM rende disponibili a un'applicazione i servizi non crittografici di una *smart card*, mediante *provider* di servizi che supportano interfacce specifiche. Un'interfaccia per *smart card* include un insieme predefinito di servizi, i protocolli necessari per richiamare tali servizi e quanto presupposto dal contesto di questi servizi. È qualcosa di simile all'identificatore di applicazioni ISO 7816-5, ma con un ambito diverso.

Una *smart card* è in grado di registrare il supporto per un'interfaccia tramite l'associazione con il **GUI**D (Identificatore univoco globale, *Globally Unique IDentifier*) dell'interfaccia. Questo collegamento tra *smart card* e interfaccia avviene quando la *smart card* viene inserita per la prima volta nel sistema, in genere al momento dell'installazione del *provider* di servizi. Dopo aver inserito la *smart card* nel sistema, le applicazioni possono eseguire la ricerca di *smart card* in base a un'interfaccia o a un **GUI**D specifici. Ad esempio, è possibile rendere disponibile alle applicazioni *Windows* una *smart card* per prelievo di denaro, registrando le interfacce affinché accedano al relativo schema di spesa.

Integrandoli nella versione 1.0 di *Smart Card Base Components*, *Microsoft* ha reso disponibili vari *provider* di servizi di livello base in grado di eseguire operazioni generiche, ad esempio l'individuazione delle *smart card*, la gestione APDU (*Application Protocol Data Unit*) di comandi e risposte nonché l'accesso al *file system* delle *smart card*. I *provider* di servizi resi disponibili da *Microsoft* vengono installati come oggetti interfaccia COM per consentire agli sviluppatori di *software* e ai *provider* di *smart card* di sviluppare *provider* di servizi e applicazioni di livello superiore.

Gli sviluppatori di *software* possono utilizzare strumenti di sviluppo *standard* quali *Visual* C++ e *Visual Basic* per sviluppare applicazioni e *provider* di servizi in grado di utilizzare le *smart card*.

Componenti base 2

Il sottosistema per *smart card* include i componenti seguenti:

- *Provider* di servizi: *provider* CSP per *smart card* e *provider* di servizi SCard

COM.
- Gestore di risorse.
- *Driver* di periferica.
- Libreria di *driver* per lettori.

Provider di servizi

Affinché le applicazioni *Windows* siano in grado di accedere ai servizi basati su *smart card*, tutte le *smart card* devono disporre di almeno un *provider* di servizi. A seconda del tipo di *smart card* e del relativo distributore, possono esistere più *provider* di servizi. In generale, esistono due categorie di *provider* di servizi: crittografici e non crittografici.

Gestore di risorse

Il gestore di risorse delle *smart card* viene eseguito come servizio attendibile (*trusted*) in un unico processo. Tutte le richieste di accesso con *smart card* vengono indirizzate mediante il gestore di risorse al lettore che contiene la *smart card*. Pertanto, il gestore di risorse è responsabile della gestione e del controllo dell'accesso di tutte le applicazioni a qualsiasi *smart card* inserita in qualsiasi lettore collegato a un *computer* che esegue un sistema operativo *Windows*. Il gestore di risorse rende disponibile a una determinata applicazione una connessione virtuale diretta alla *smart card* richiesta.

Durante la gestione dell'accesso a più lettori e *smart card*, il gestore di risorse esegue tre attività fondamentali. Innanzitutto identifica e registra le risorse. In secondo luogo, controlla l'allocazione dei lettori e delle risorse tra più applicazioni. Infine, supporta le transazioni essenziali per l'accesso ai servizi disponibili in una *smart card* specifica. Si tratta di un aspetto importante, poiché le *smart card* attuali sono periferiche a *thread* singolo che spesso richiedono l'esecuzione di più comandi per portare a termine una singola funzione. Il controllo delle transazioni consente l'esecuzione senza interruzione di più comandi, garantendo così che le informazioni sullo stato intermedio non vengano danneggiate.

Driver di periferica

Un *driver* di periferica per un lettore specifico associa la funzionalità del lettore ai servizi originali resi disponibili dal sistema operativo *Windows* e dall'infrastruttura della *smart card*. Il *driver* del lettore comunica l'inserimento e la rimozione della *smart card* al gestore di risorse e rende disponibili le funzioni di comunicazione delle informazioni da e verso la *smart card*.

Libreria di *driver* per lettori

Smart Card Base Components 1.0 include una libreria di *driver* per lettori utilizzabile dagli sviluppatori per semplificare lo sviluppo di *driver* di periferiche. Questa libreria condivisa supporta lo *standard* ISO 7816 e le comuni funzioni di sistema necessarie alla comunicazione di dati tra una *smart card* e un lettore. Rappresenta un miglioramento significativo rispetto alle modalità con cui i *driver* per lettori di *smart card* venivano sviluppati in passato, poiché esistono oggi interfacce *standard* su cui gli sviluppatori possono basarsi. Queste interfacce comuni consentono uno sviluppo omogeneo di *driver* per lettori di *smart card* e la conseguente accessibilità da parte di tutte le applicazioni *Windows*, e non solo di un selezionato numero in grado di comunicare solo con un lettore specifico.

Soluzioni avanzate

Migliorando le soluzioni basate solo su *software*, quali l'autenticazione dei *client* e la messaggistica protetta, le *smart card* disponibili in *Windows* 2000 consentono alle applicazioni di sfruttare le opportunità offerte dall'emergente economia digitale globale. Le *smart card* offrono agli sviluppatori di applicazioni un metodo sicuro per migliorare le soluzioni per l'azienda e il consumatore.

Autenticazione *client*

L'autenticazione *client* implica l'identificazione e la convalida di un *client* a un *server* per stabilire un canale di comunicazione protetto. In genere vengono utilizzati protocolli protetti quali SSL (*Secure Sockets Layer*) o TLS (*Transport Layer Security*), di solito insieme con un certificato con chiave pubblica attendibile che viene fornito dal *client*. Questo certificato consente al *server* di identificare il *client*. Ad esempio, il *client* potrebbe essere *Microsoft Internet Explorer* in esecuzione in un sistema operativo *Windows* e il *server* potrebbe essere *Internet Information Server* o un altro *server Web* che supporta i protocolli SSL/TLS.

La sessione protetta viene stabilita utilizzando l'autenticazione con chiave pubblica con scambio di chiavi, per ricavare una chiave di sessione univoca che può essere quindi utilizzata per garantire l'integrità e la riservatezza dei dati durante la sessione. È inoltre possibile ottenere un'autenticazione aggiuntiva associando il certificato a un *account* utente o di gruppo che disponga di privilegi per il controllo dell'accesso precedentemente stabiliti. La *smart card* migliora il processo di autenticazione con chiave pubblica, poiché viene utilizzata come archivio protetto della chiave privata e come motore crittografico per l'esecuzione di una firma digitale o di uno scambio di chiavi.

Posta elettronica protetta

La posta elettronica protetta è una delle più interessanti applicazioni abilitate all'utilizzo della chiave pubblica, perché consente agli utenti di condividere informazioni in completa riservatezza e garantisce che l'integrità delle informazioni venga mantenuta durante il transito in *Internet*. Utilizzando *Microsoft Outlook Express* o *Microsoft Outlook*, è possibile selezionare un certificato con chiave pubblica emesso da un'autorità di certificazione attendibile, da utilizzare per firmare digitalmente e decifrare messaggi protetti. Pubblicando il certificato dell'utente in una *directory* pubblica dell'azienda o in *Internet*, altri utenti dell'azienda o di *Internet* possono inviare messaggi di posta elettronica crittografati all'utente e viceversa.

Le *smart card* aggiungono un ulteriore livello di integrità alle applicazioni di posta elettronica protetta poiché consentono di archiviare internamente la chiave privata e di proteggerla con un codice *PIN*. Per manomettere la chiave privata e inviare messaggi firmati assumendo l'identità di un'altra persona, è necessario appropriarsi della *smart card* e del relativo *PIN*.

Programmabilità delle *smart card*

Le *smart card* consentono di ospitare sistemi operativi come *Microsoft Windows for Smart Cards*, e in molti casi un tipo di *file system* nel quale è possibile archiviare dati. Per funzionalità quali l'accesso con *smart card* di *Windows* 2000, è necessario che la

smart card sia programmabile, così da consentire le operazioni seguenti:

- Archiviazione di una coppia di chiavi utente.
- Archiviazione di un certificato con chiave pubblica associato.
- Recupero di un certificato con chiave pubblica.
- Operazioni con chiave privata complete per conto dell'utente.

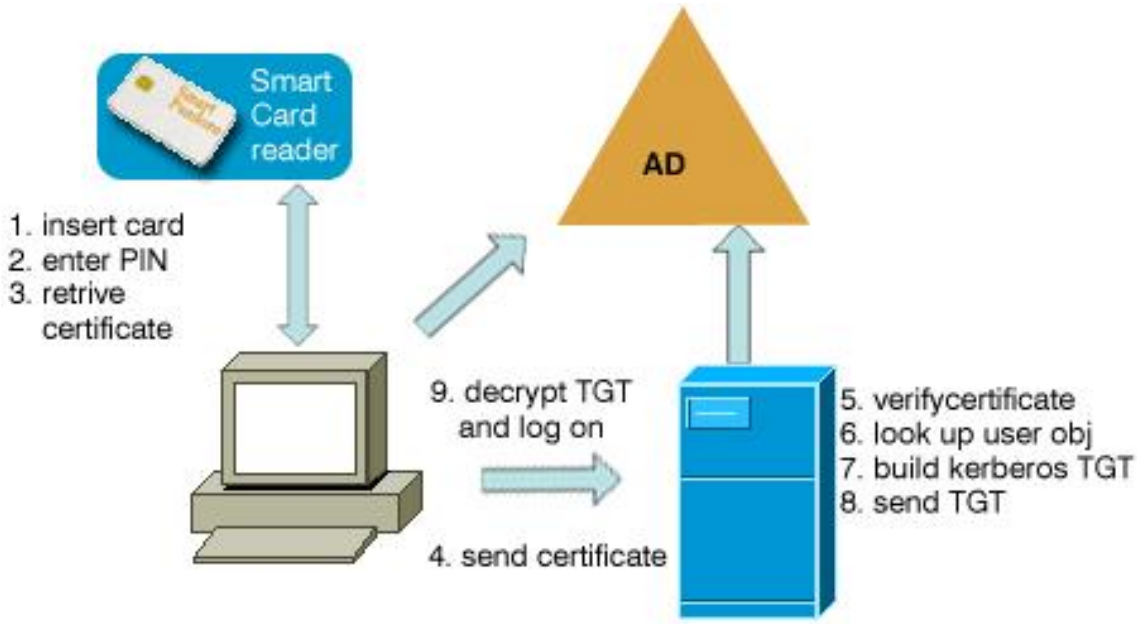
Accesso interattivo con chiave pubblica

Un tipo di utilizzo delle *smartcard* particolarmente importante in *Windows* 2000 è quello relativo al *logon* a chiave pubblica. È possibile utilizzare il certificato contenuto nella *smartcard* per compiere il *logon* su un dominio *Windows* 2000, invece di utilizzare il normale procedimento di *logon* di NT.

Nel passato, l'accesso interattivo indicava la capacità di autenticare un utente in una rete utilizzando una forma di credenziale condivisa, ad esempio una *password* con *hash*. *Windows* 2000 supporta l'accesso interattivo con chiave pubblica utilizzando la versione 3 del certificato X.509 archiviato in una *smart card* con la chiave privata. Invece della *password*, l'utente digita un codice *PIN* per l'autenticazione GINA (*Graphical IdentificatioN and Authentication*). Tale codice autentica l'utente per la *smart card*.

Il certificato con chiave pubblica dell'utente viene recuperato dalla *smart card* mediante un processo protetto, ne viene verificata la validità e la provenienza da una fonte attendibile. Durante il processo di autenticazione, viene inviato alla *smart card* un messaggio di verifica (*challenge*) basato sulla chiave pubblica contenuta nel certificato. Questo messaggio verifica chi è il proprietario della *smart card* e la sua autorizzazione a utilizzare la chiave privata corrispondente.

Dopo la verifica della coppia di chiavi pubblica e privata, l'identità dell'utente contenuta nel certificato viene utilizzata per far riferimento all'oggetto utente archiviato in *Active Directory*, per elaborare un *token* e restituire al *client* il *ticket* TGT (*Ticket-Granting Ticket*). *Microsoft* ha integrato l'accesso con chiave pubblica nella versione 5 di *Kerberos*, che è compatibile con l'estensione per chiave pubblica specificata nella bozza RFC-1510 dell'organizzazione IETF.



schema di autenticazione tramite Smart Card

Quando la si inserisce nel lettore (1) collegato al *computer, Windows* 2000 chiede di inserire il codice *PIN* (*Personal Identification Number*) (2) nell'apposita finestra. Se il sistema operativo identifica l'utente quale proprietario della *card*, l'autorità LSA (*Local Security Authority*) del *computer* recupera il certificato dalla *smartcard* (3) e lo invia al centro KDC (4) (*Key Distribution Center*) *Kerberos*, il quale è un *controller* di dominio *Windows* 2000.

Dopo che quest'ultimo ha verificato la validità del certificato (5) e di chi lo ha rilasciato (per esempio, dell'autorità CA), il nome del soggetto presente del certificato viene utilizzato quale riferimento per compiere la ricerca di un *user object* (6) in AD. Quando il centro KCD trova l'oggetto, viene creato un *ticket* TGT (7) (*Ticket-Granting Ticket*) *Kerberos* che contiene l'*account* dell'utente e le informazioni per il controllo dell'accesso. Il centro KDC crittografa il *ticket* TGT utilizzando una chiave di sessione (che viene successivamente crittografata con la chiave pubblica dell'utente) e lo restituisce al *computer* (8).

(9) La *smartcard* decodifica quindi la chiave di sessione utilizzando quella privata che è contenuta dalla *smartcard* stessa, e utilizza la prima per decodificare il *ticket* TGT. Il centro KDC consente infine all'autorità LSA di autorizzare il *logon* dell'utente sul dominio *Windows* 2000.

Oltre al *logon* tramite *smartcard*, è possibile utilizzare questo servizio di *Windows* 2000 anche per altre operazioni PKI, come l'autenticazione *client* e *server* nel protocollo SSL (*Secure Sockets Layer*). Per sfruttare completamente le possibilità offerte dalla *smartcard*, si possono integrare anche altre funzioni di sicurezza oltre al certificato digitale, per esempio le informazioni identificative dell'azienda e l'autorizzazione Bancomat.

Configurare le Smart Card

Prima di poter utilizzare le *Smart Card*, occorre l'infrastruttura PKI fornita da *Windows* 2000 con *Certificate Server* e *Active Directory*. È disponibile un gran numero di opzioni relative all'impostazione dell'infrastruttura PKI, a seconda che si ospiti una propria *authority* CA (*Certification Authority*), la si assegni in *outsourcing* a un *provider* come *VeriSign*, oppure si faccia uso di uno schema intermedio.

Questo scenario prevede l'uso di un *computer* che funge da *controller* di dominio *Active Directory*, di *Certificate Server* installato su un'*authority* CA radice dell'azienda e di un lettore di *Smart Card*. L'*authority* CA radice (CA *Root*) è la più alta in assoluto nella gerarchia della fiducia. La presenza di un'*authority* CA dell'azienda implica che il *server* di certificati userà *Active Directory* quale proprio *data store* e in conseguenza metterà a disposizione una serie di funzioni per la gestione dei certificati. Le *authority* CA indipendenti, che utilizzano un *database* locale invece di *Active Directory*, vengono usate di solito dai *server Web* pubblici per fornire i certificati agli utenti di *Internet*.

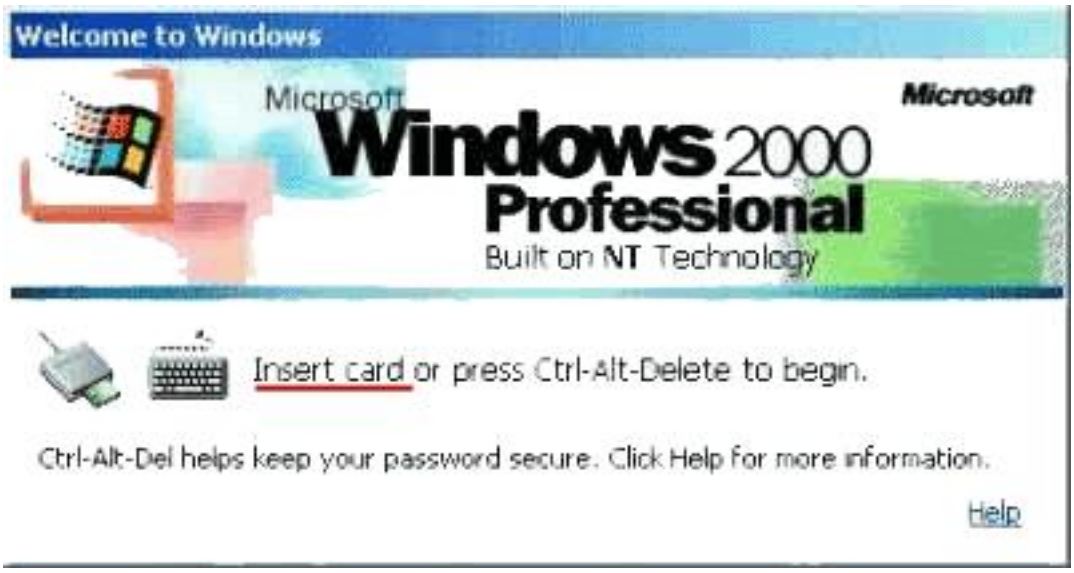
Dopo avere installato *Windows* 2000 quale *controller* di dominio che usa lo schema DNS (*Domain Name System*) e il *server Microsoft* IIS (*Internet Information Server*), occorre installare anche *Certificate Server*.

Porsi in corrispondenza della *applet Add/Remove Windows Components* nel Pannello di Controllo, quindi selezionare *Certificate Services*. Installare *Certificate Server* quale *authority* CA aziendale radice. Dopo avere specificato il nome, si possono usare i valori di *default* in tutte le successive finestre di dialogo. Si dovrà quindi definire quali tipi di certificati possono essere emessi dalla nuova *authority* CA. Si possono usare i certificati per vari scopi, tra cui quello di convalidare i *server Web*, crittografare i *file* con il sistema EFS (*Encrypting File System*) e rendere sicura la posta elettronica. I modelli dei certificati, che sono una funzionalità di gestione delle *authority* CA aziendali, permettono di definire i possibili usi di un particolare certificato.

Tra i modelli è necessario senz'altro il modello *Smartcard Logon* (si accede a questi modelli tramite lo *snap-in Certificate Authority* della *consolle* MMCche risulta disponibile dopo la configurazione di *Certificate Server*). Come suggerisce il nome, i certificati *Smartcard Logon* sono limitati al *logon*; i certificati *Smartcard User* permettono invece di usare una *Smart Card* per compiere il *logon* e per firmare la posta elettronica. Occorre inoltre aggiungere il modello *Enrollment Agent*, che si rende necessario per creare nuovi certificati. A questo punto, si ha a disposizione un'infrastruttura PKI in grado di autenticare gli utenti attraverso le *Smart Card*.

Configurare l'hardware

La fase successiva è quella di installare, ad esempio, il lettore di *Smart Card* GCR410; questa installazione è semplice grazie al *Plug and Play*. Spegnere il sistema, quindi collegare al *computer* i due cavi del lettore. Il cavo PS/2 utilizza un connettore a cuneo, che si inserisce tra la porta PS/2 e il cavo della tastiera o del *mouse*. Il lettore viene alimentato dalla porta PS/2. Il cavo seriale, che deve essere collegato a una porta seriale di riserva, permette di instaurare le comunicazioni tra il sistema e il lettore. Quando si avvia il *computer, Windows* 2000 rileva automaticamente la presenza del nuovo lettore.



schermata di login di Windows 2000

Dal momento che l'implementazione delle *Smart Card* è una funzione amministrativa particolarmente delicata, proprio come l'assegnazione dei valori di *ID* utente e *password*, questa capacità dovrebbe essere limitata a utenti e *workstation* specifiche. In conseguenza, la fase successiva è quella di trasformare un *computer* in una *workstation* per il rilascio dei certificati *Smart Card*. Per prima cosa, richiedere un certificato *Enrollment Agent* per l'amministratore che dovrà rilasciare le *Smart Card* agli utenti. Quando l'amministratore richiede i certificati *Smart Card* per i nuovi utenti, *Windows* 2000 firma la richiesta con questo certificato *Enrollment Agent*. Successivamente aprire lo *snap-in* MMC *Certificates*, che permette di gestire i certificati associati al proprio *account* utente e di richiederne di nuovi. Richiedere infine un nuovo certificato *Enrollment Agent*, fornire un nome amichevole e fare clic con il *mouse* sul pulsante *Install*.

Le fasi precedenti sono necessarie soltanto per la configurazione iniziale. Quelle successive dovranno invece essere ripetute per ciascun utente che deve utilizzare una *Smart Card* (in questo esempio che prevede soltanto un *computer*, l'utente avrà bisogno del diritto di compiere il *logon* localmente su questo *server*, essendo per esempio un membro del gruppo *Server Operators*).

È ora possibile richiedere i certificati *Smart Card* per conto di altri utenti. Selezionare un *account* utente già esistente, oppure crearne uno nuovo. Aprire *Microsoft Internet Explorer* e digitare il nome del proprio *server* di *authority* CA (per esempio, b1) seguito da /CertSrv, ad esempio http://b1/CertSrv.

Accesso interattivo con chiave pubblica

Richiedere un certificato, quindi specificare una richiesta avanzata nella schermata successiva. In corrispondenza di quest'ultima, selezionare *Request a certificate for a smart card on behalf of another user using the Smart Card Enrollment Station* (Richiedere un certificato per una *Smart Card* per conto di un altro utente, usando la *Smart Card Enrollment Station*).

Selezionare il *Certificate Template* (per esempio, *Smartcard User o Smartcard Logon*); è inoltre possibile specificare quale *authority* CA utilizzare, anche se il valore di *default* per questo campo è relativo all'*authority* CA che è stata installata. Si può specificare il *provider* CSP (*Cryptographic Service Provider*) che deve essere utilizzato con questo certificato. Il *provider* CSP fornisce al sistema operativo e alle applicazioni alcuni servizi crittografici di base; quelli specifici alle *Smart Card* utilizzano la scheda per soddisfare le richieste relative ai servizi che prevedono l'uso della chiave privata.

È inoltre possibile specificare il certificato *Enrollment Agent* che dovrà firmare questa richiesta. Utilizzare il certificato *Enrollment Agent* creato in precedenza. È infine necessario specificare l'utente per conto del quale si sta richiedendo il certificato, quindi fare clic con il *mouse* in corrispondenza del pulsante *Enroll*. Il sistema chiederà di inserire la *Smart Card* nel lettore e di digitare il relativo codice *PIN*. A questo punto, *Certificate Server* visualizza una pagina nel *browser*, informando che l'operazione è stata compiuta con successo. È ora possibile visualizzare il certificato sulla *Smart Card*, oppure passare a un altro utente.

A questo punto, si è pronti per compiere il *logon* usando la *Smart Card*. Per prima cosa, scollegarsi dal sistema e re-inserire la scheda. Il lettore notifica al sistema che è stata inserita la scheda, quindi viene chiesto di digitare il codice *PIN*. Dopo averlo digitato, questa parte del *logon* è completa. *Windows* 2000 invia il codice *PIN* alla *Smart Card* per l'autenticazione.

Successivamente, il codice viene usato per compiere alcune funzioni crittografiche richieste da *Windows* 2000 (per esempio, il sistema operativo potrebbe chiedere alla *Smart Card* di firmare una richiesta di *logon*).

Microsoft utilizza una proposta della IETF (*Internet Engineering Task Force*) chiamata PKINIT, per estendere il protocollo di autenticazione *Kerberos* versione 5 in modo da supportare anche l'autenticazione a chiave privata e pubblica, oltre al metodo a chiave simmetrica e segreto condiviso già usato da questo protocollo.

In conseguenza, la *workstation* invia la richiesta di *logon* a un *controller* di dominio, tramite una richiesta AS (*Authentication Services*) *Kerberos*. Il centro KDC (*Key Distribution Center*) *Kerberos* sul *controller* di dominio verifica la richiesta usando la chiave pubblica del certificato, che viene pubblicata dall'*authority* CA. Se i dati sono corretti, *Kerberos* assegna un *ticket* e permette di compiere il *logon*. Se l'utente perde la *Smart Card*, è sufficiente aprire lo *snap-in* MMC *Certificate Services*, selezionare il certificato sotto *Issued Certificates*, revocarlo, pubblicare la lista CRL (*Certificate Revocation List*) aggiornata e fornire all'utente una nuova *Smart Card*. Se qualcuno dovesse cercare di usare la vecchia *Smart Card* ormai non più valida, *Windows* 2000 rifiuterebbe il *logon*.

Installazione del servizio di Active Directory

Franco Callegati
Walter Cerroni
9.1.3 (Progettare, installare e mantenere una struttura di directory)

Attività preliminari

Windows 2000 fornisce un *wizard* per l'installazione di *Active Directory* e dunque di un controllore di dominio. Tale *wizard* è costituito dall'eseguibile 'DCPROMO.EXE'. Tramite lo stesso eseguibile è possibile eseguire la disinstallazione di *Active Directory* e quindi il declassamento del controllore di dominio a *Member Server*.

L'esecuzione di tale *Wizard* consente di realizzare una delle seguenti funzioni:

- aggiunta di un controllore di dominio ad un dominio esistente.
- Creazione di un nuovo dominio figlio di un dominio padre in un albero esistente.
- Creazione di un nuovo albero in una foresta esistente.
- Creazione di una nuova foresta.

Prima di avviare l'installazione di *Active Directory* bisogna verificare che siano soddisfatti i seguenti prerequisiti:

- Un *computer* che esegua *Windows* 2000 *Server* o *Windows* 2000 *Advanced Server* o *Windows* 2000 *Datacenter Server*.
- Una partizione o un volume NTFS.
- Spazio disco adeguato per le informazioni di *directory* (raccomandato 1 *gigabyte* [GB]).
- *TCP*/*IP* installato e configurato per utilizzare DNS.
- Un *server* DNS che supporti i *record* SRV (*service resource records*) ed opzionalmente l'aggiornamento dinamico ed il trasferimento incrementale. Il *wizard* di installazione di *Active Directory* offre la possibilità di installare il DNS se si sta installando il primo controllore di dominio e non esiste un DNS primario per tale dominio oppure esiste ma non supporta gli aggiornamenti dinamici.
- Le credenziali opportune per poter aggiungere un controllore di dominio ad un dominio esistente, oppure un dominio ad un albero esistente oppure un albero ad una foresta esistente: *logon name, password* e dominio.

Durante l'installazione di *Active Directory* viene generato un *file* di *log* salvato nella cartella systemroot\Debug.

Creare il Root Domain

Quando installiamo *Active Directory* per la prima volta sulla nostra rete, stiamo creando il primo controllore di dominio della foresta e dunque il dominio radice della foresta. Tale dominio contiene le informazioni relative allo Schema ed alla Configurazione della Foresta.

Dopo aver eseguito DCPROMO.EXE usare le seguenti informazioni per completare le varie pagine proposte dal *wizard*:

- *Domain Controller Type*: *Domain controller for a new domain*.
- *Create Tree or Child Domain*: *Create a new domain tree*.
- *Create or Join Forest*: *Create a new forest of domain trees*.
- *New Domain Name*: Specificare il nome DNS del nuovo dominio.
- *Domain **NetBIOS** Name*: Specificare il nome *NETBIOS* del dominio. Permetterà a macchine *Windows* precedenti a *Windows* 2000 di identificare il nuovo dominio.
- *Database and Log Locations*: Specificare la locazione del *database* di *Active Directory* (NTDS.DIT) e dei relativi *file* di *log*. Per entrambi il *default* è systemroot\Ntds. Se possibile, per migliorare le *performance* è bene memorizzare il *database* e i *files* di *log* su dischi distinti.
- *Shared System Volume*: Specificare la locazione della condivisione di sistema SYSVOL. Tale struttura risiede su tutti i controllori di dominio, serve ad ospitare *files* ed informazioni relative alle *Group Policies* che vengono replicate tra tutti i controllori di dominio e la partizione che la ospita deve essere NTFS.
- *Permissions*: Specificare se i permessi impostati di *default* su utenti e gruppi sono compatibili solo con *server Windows* 2000 o anche con precedenti versioni di *Windows*.
- *Directory Services Restore Mode Administrator Password*: Specificare la *password* utilizzata dall'amministratore per eseguire l'opzione di *startup Directory Services Restore Mode*.

Una volta che sono state specificate le opzioni di cui sopra, il *wizard*:

- Installa *Active Directory*.
- Converte il *server* a controllore di dominio.
- Aggiunge al gruppo di programmi *Administrative Tools* i seguenti *tools*:
  - *Active Directory Domains and Trusts*. Per la gestione delle relazioni di fiducia.
  - *Active Directory Sites and Services*. Per la gestione dei siti e della replica tra controllori di dominio.
  - *Active Directory Users and Computers*. Per la gestione di oggetti di *Active Directory* (utenti, gruppi, *computer*) e per il passaggio del dominio da Modalità Mista a Modalità Nativa.

Aggiungere un Controllore di Dominio

Una volta creato un dominio installando il primo controllore di dominio, solitamente si aggiunge, per garantire disponibilità e *performance*, almeno un secondo controllore di dominio.

Dopo aver avviato l'esecuzione di DCPROMO.EXE usare le seguenti informazioni per completare le varie pagine proposte dal *wizard*:

- *Domain Controller Type*: *Additional domain controller for an existing domain*.
- *Network Credentials*: Specificare il nome utente, la *password* ed il dominio dell'*account* che ha i privilegi per aggiungere un controllore di dominio al dominio.

- *Additional Domain Controller*: Specifcare il nome DNS del dominio in cui aggiungiamo un controllore di dominio.
- *Database and Log Locations*: Specificare la locazione del *database* di *Active Directory* (NTDS.DIT) e dei relativi *files* di *log*. Per entrambi il *default* è systemroot\Ntds. Se possibile, per migliorare le *performance* è bene memorizzare il *database* e i *files* di *log* su dischi distinti.
- *Shared System Volume*: Specificare la locazione della condivisione di sistema SYSVOL. Tale struttura risiede su tutti i controllori di dominio, serve ad ospitare *files* ed informazioni relative alle *Group Policies* che vengono replicate tra tutti i controllori di dominio e la partizione che la ospita deve essere NTFS.
- *Directory Services Restore Mode Administrator Password*: Specificare la *password* utilizzata dall'amministratore per eseguire l'opzione di *startup Directory Services Restore Mode*.

Una volta che sono state specificate le opzioni di cui sopra, il *wizard*:

- Installa *Active Directory*.
- Converte il *server* a controllore di dominio.
- Replica le informazioni di *directory* da un controllore di dominio esistente.
- Aggiunge al gruppo di programmi *Administrative Tools* i seguenti *tools*:
  - *Active Directory Domains and Trusts*. Per la gestione delle relazioni di fiducia.
  - *Active Directory Sites and Services*. Per la gestione dei siti e della replica tra controllori di dominio.
  - *Active Directory Users and Computers*. Per la gestione di oggetti di *Active Directory* (utenti, gruppi, *computer*) e per il passaggio del dominio da Modalità Mista a Modalità Nativa.

Aggiungere un Dominio Figlio

Credenziali

Una volta creato il dominio radice dell'albero è possibile aggiungere all'albero ulteriori domini figli del dominio radice o figli di un altro dominio figlio.

Dopo aver lanciato DCPROMO.EXE usare le seguenti informazioni per completare le varie pagine proposte dal *wizard*:

- *Domain Controller Type*: *Domain controller for a new domain*.
- *Create Tree or Child Domain*: *Create a new child domain in an existing domain tree*.
- *Network Credentials*: Specificare il nome utente, la *password* ed il dominio dell'*account* che appartiene al gruppo *Enterprise Admin* nella foresta esistente.
- *Database and Log Locations*: Specificare la locazione del *database* di *Active Directory* (NTDS.DIT) e dei relativi *files* di *log*. Per entrambi il *default* è systemroot\Ntds. Se possibile, per migliorare le *performance* è bene memorizzare il *database* e i *files* di *log* su dischi distinti.
- *Shared System Volume*: Specificare la locazione della condivisione di

sistema SYSVOL. Tale struttura risiede su tutti i controllori di dominio, serve ad ospitare *files* ed informazioni relative alle *Group Policies* che vengono replicate tra tutti i controllori di dominio e la partizione che la ospita deve essere NTFS.
- *Permissions*: Specificare se i permessi impostati di *default* su utenti e gruppi sono compatibili solo con *server Windows* 2000 o anche con precedenti versioni di *Windows*.
- *Directory Services Restore Mode Administrator Password*: Specificare la *password* utilizzata dall'amministratore per eseguire l'opzione di *startup Directory Services Restore Mode*.

Una volta che sono state specificate le opzioni di cui sopra, il *wizard*:

- Installa *Active Directory*.
- Converte il *server* a controllore di dominio.
- Aggiunge al gruppo di programmi *Administrative Tools* i seguenti *tools*:
  - *Active Directory Domains and Trusts*. Per la gestione delle relazioni di fiducia.
  - *Active Directory Sites and Services*. Per la gestione dei siti e della replica tra controllori di dominio.
  - *Active Directory Users and Computers*. Per la gestione di oggetti di *Active Directory* (utenti, gruppi, *computer*) e per il passaggio del dominio da Modalità Mista a Modalità Nativa.

Aggiungere un Albero ad una Foresta

Una volta creato il dominio radice, dunque almeno un albero, è possibile aggiungere alla foresta un albero ulteriore.

Dopo aver lanciato DCPROMO.EXE usare le seguenti informazioni per completare le varie pagine proposte dal *wizard*:

- *Domain Controller Type*: *Domain controller for a new domain*.
- *Create Tree or Child Domain*: *Create a new domain tree*.
- *Create or Join Forest*: *Place this new domain tree in an existing forest*.
- *Network Credentials*: Specificare il nome utente, la *password* ed il dominio dell'*account* che appartiene al gruppo *Enterprise Admin* nella foresta esistente.
- *New Domain Tree*: Specificare il nome DNS del nuovo albero.
- *Domain **NetBIOS** Name*: Specificare il nome *NETBIOS* del dominio. Permetterà a macchine *Windows* precedenti a *Windows* 2000 di identificare il nuovo dominio.
- *Database and Log Locations*: Specificare la locazione del *database* di *Active Directory* (NTDS.DIT) e dei relativi *files* di *log*. Per entrambi il *default* è systemroot\Ntds. Se possibile, per migliorare le *performance* è bene memorizzare il *database* e i *files* di *log* su dischi distinti.
- *Shared System Volume*: Specificare la locazione della condivisione di sistema SYSVOL. Tale struttura risiede su tutti i controllori di dominio, serve ad ospitare *files* ed informazioni relative alle *Group Policies* che vengono replicate tra tutti i controllori di dominio e la partizione che la

ospita deve essere NTFS.

- *Permissions*: Specificare se i permessi impostati di *default* su utenti e gruppi sono compatibili solo con *server Windows* 2000 o anche con precedenti versioni di *Windows*.
- *Directory Services Restore Mode Administrator Password*: Specificare la *password* utilizzata dall'amministratore per eseguire l'opzione di *startup Directory Services Restore Mode*.

Una volta che sono state specificate le opzioni di cui sopra, il *wizard*:

- Installa *Active Directory*.
- Converte il *server* a controllore di dominio.
- Aggiunge al gruppo di programmi *Administrative Tools* i seguenti *tools*:
  - *Active Directory Domains and Trusts*. Per la gestione delle relazioni di fiducia.
  - *Active Directory Sites and Services*. Per la gestione dei siti e della replica tra controllori di dominio.
  - *Active Directory Users and Computers*. Per la gestione di oggetti di *Active Directory* (utenti, gruppi, *computer*) e per il passaggio del dominio da Modalità Mista a Modalità Nativa.

Deleghe di amministrazione in sistemi Linux/Unix

Walter Cerroni

9.1.2 (Gestire gli account degli utenti incluso script di login) - 9.1.4 (Assegnare agli utenti i diritti appropriati per accesso a file, applicazioni e risorse)

Introduzione

Nell'ambito dell'amministrazione di una rete locale di grandi dimensioni può essere utile avere la possibilità di delegare ad altri utenti non privilegiati alcuni compiti specifici dell'amministratore, riducendo così il carico di lavoro dell'amministratore stesso senza per questo far conoscere a troppe persone la *password* di superutente. Ad esempio, alcuni compiti tipici che potrebbero essere delegati ad altri utenti sono la gestione degli *account* di utente, delle stampanti, delle operazioni di *backup*, o di alcuni servizi di rete come il *server Web*.

Bit Set User ID

Tradizionalmente, nei sistemi *Linux/Unix* la modalità di delegare ad utenti non privilegiati l'utilizzo di alcuni comandi di amministrazione si è basata tipicamente sull'impiego del cosiddetto *bit Set User ID*. Si è già visto in **Gestione dei permessi** che ogni *file* o *directory* nel *file system* è caratterizzato da nove *bit* che determinano i permessi di lettura, scrittura ed esecuzione (r, w, x) per il proprietario, il gruppo del proprietario e per gli altri utenti del sistema (u, g, o). Oltre a questi nove *bit*, ad ogni *file* o *directory* sono associati tre *bit* ulteriori che fanno riferimento a speciali modalità di accesso:

- SUID (*Set User ID*) - si accede al *file* utilizzando l'identificativo del proprietario (UID).
- SGID (*Set Group ID*) - si accede al *file* utilizzando l'identificativo del gruppo a cui appartiene il proprietario (GID).
- *Sticky* - se applicato ad una *directory*, i *file* contenuti in essa non possono essere cancellati o rinominati da un utente diverso dal proprietario, pur avendo il permesso di scrittura nella *directory*.

In particolare, se un *file* eseguibile di proprietà di *root* ha il *bit* SUID settato, chiunque lo esegua, lo farà assumendo l'identificativo di *root* (UID = 0) e, quindi, avrà tutti i privilegi del superutente, limitatamente al campo d'azione del comando in questione. Un esempio di eseguibile con il *bit* SUID attivo è il comando passwd, utilizzato per cambiare la *password*: poiché ogni utente deve essere in grado di cambiare la propria *password* e poiché solo *root* ha il diritto di scrivere sui *file* /etc/passwd e /etc/shadow, è necessario che tale comando sia eseguito con i privilegi dell'amministratore. Naturalmente, l'implementazione di passwd è tale da negare comunque ad un utente diverso da *root* la possibilità di cambiare la *password* di altri utenti o dello stesso *root*.

Per verificare se il *bit* SUID del comando passwd sia settato, si può procedere come nell'esempio che segue:

```
[root@host1 root]# ls -l /usr/bin/passwd
-r-s--x--x 1 root root 15368 mag 28 2002 /usr/bin/passwd
```

Il carattere s al posto della x nei permessi del proprietario denota l'attivazione di SUID.

Se si vuole delegare un utente diverso da *root* ad eseguire un particolare comando, si può settare il *bit* SUID dell'eseguibile tramite il comando chmod. Ad esempio, per delegare la creazione degli utenti è necessario che *root* attivi il *bit* SUID al comando useradd:

```
[root@host1 root]# chmod u+s /usr/sbin/useradd
```

A questo punto l'utente *user*1 è in grado di creare un altro utente chiamato *user*2:

```
[user1@host1 user1]$ /usr/sbin/useradd user2
```

cosa che può essere verificata controllando il contenuto del *file* /etc/passwd:

```
[user1@host1 user1]$ cat /etc/passwd |grep user2
user2:x:1002:1002::/home/user2:/bin/bash
```

Allo scopo di delegare l'esecuzione di un comando con i privilegi di *root*, l'utilizzo di SUID non consente tuttavia di ottenere un uso selettivo di questa funzionalità sulla base dell'utente che esegue il comando stesso. In altre parole, qualunque utente può utilizzare il comando in questione con i privilegi di *root*.

Comando sudo

Per una gestione più efficiente delle deleghe si può utilizzare il comando *sudo* (che sta per *superuser do*), disponibile nelle versioni più recenti dei sistemi operativi *Linux/Unix* o reperibile dal sito http://www.sudo.ws. Questo comando svolge la funzione di permettere a determinati utenti (e solo a loro) di eseguire determinati comandi (e solo quelli) da determinati *host* (e solo da quelli) con i privilegi di amministratore. Inoltre, consente di monitorare l'attività dei delegati inserendo nel registro di sistema (*syslog*) tutte le chiamate a *sudo* con i relativi argomenti. L'impostazione delle deleghe viene realizzata tramite il *file* di configurazione /etc/sudoers, che tipicamente è accessibile solo a *root* e che deve essere modificato tramite l'editor visudo (una versione dell'editor vi dedicata a questo scopo).

Esempio

Come esempio, supponiamo di voler impostare le seguenti deleghe:

- consentire all'utente *user*1 di aggiungere utenti tramite il comando useradd da qualunque *host*;
- consentire all'utente *user*2 di cambiare la *password* di ogni utente, *root* escluso, operando solo dall'*host host*1.

Il *file* /etc/sudoers in questo caso dovrà contenere le seguenti righe:

```
...
user1 ALL=/usr/sbin/useradd
user2 host1=/usr/bin/passwd [A-z]*,!/usr/bin/passwd root
...
```

Nella prima riga dell'esempio sono indicati: l'utente delegato (*user*1), gli *host* da cui è permesso l'uso della delega (*ALL*, cioè tutti), il comando del quale si vuole consentire l'esecuzione a *user*1 indicandone il percorso completo (/usr/sbin/useradd).

La seconda riga, invece, mostra che l'utente *user*2 può eseguire solo dall'*host host*1 il comando /usr/bin/passwd con alcune restrizioni: innanzitutto è richiesto di specificare sempre un argomento che inizi con un carattere alfabetico e che contenga uno o più caratteri (che soddisfi, cioè, l'espressione regolare [A-z]*); poi, è negata (tramite il carattere di negazione !) l'esecuzione di /usr/bin/passwd con argomento *root*, che permetterebbe a *user*2 di modificare la *password* dell'amministratore.

Una volta stabilite queste regole, gli utenti possono eseguire i comandi suddetti con i diritti di *root* facendoli precedere dal comando *sudo* e rispettando le restrizioni specificate in /etc/sudoers. La prima volta che si invoca un comando tramite *sudo*, il programma chiede di inserire la *password* dell'utente delegato per effettuarne l'autenticazione, dopodiché controlla che la sintassi del comando da eseguire sia coerente con quanto specificato in /etc/sudoers. Da quel momento in poi, ogni altra invocazione tramite *sudo* non richiede la *password*, a meno che non sia trascorso un certo intervallo di tempo (tipicamente 5 minuti) dall'ultima invocazione.

Quindi, per poter creare un *account* l'utente *user*1 dovrà procedere nel seguente modo:

```
[user1@host1 user1]$ sudo useradd user3
Password: *********
```

Per quanto riguarda *user*2, gli è consentito di cambiare la *password* all'utente appena creato:

```
[user2@host1 user2]$ sudo passwd user3
Password: **************
Changing password for user user3.
New password: ********
Retype new password: ********
passwd: all authentication tokens updated successfully.
```

mentre non può farlo per l'utente *root*:

```
[user2@host1 user2]$ sudo passwd root
Sorry, user user2 is not allowed to execute '/usr/bin/passwd root'
as root on host1.
```

Conclusioni

Le funzionalità messe a disposizione dal comando *sudo* sono molteplici e permettono un'efficiente automazione delle procedure di delega di amministrazione in reti locali di grandi dimensioni basate su sistemi *Linux/Unix*. L'applicazione ad un *case study* particolare è disponibile all'indirizzo:

http://www.komar.org/pres/sudo

Per maggiori informazioni, fare riferimento alle seguenti pagine del manuale in linea: *man sudo*, *man sudoers* e *man visudo*, disponibili anche agli indirizzi:

http://www.sudo.ws/sudo/man/sudo.html,

http://www.sudo.ws/sudo/man/sudoers.html,

http://www.sudo.ws/sudo/man/visudo.html.

Bibliografia

Libri

Infrastrutture per reti di calcolatori

A. S. tanembaum*Reti di computer - terza edizione;* 1998UTET/Prentice Hall

W. Stallings*Local and Metropolitan Area Networks - sixth edition;* 2000Prentice Hall

F. Wilder*A Guide to the TCP/IP Protocol Suite - second edition;* 1998Artech House

S. Gai*Reti locali: dal cablaggio all'internetworking;* 1995Scuola superiore "G. Reiss Romoli"

IEEE*Std 802: Overview and Architecture;* IEEE

ISO/IEC*8802.3;* ISO

IBM*Token-Ring Network: Architecture Reference;* IBM

ISO/IEC*IEEE Std 802.5 Overview and Architecture;* ISO

R. Perlman*Interconnections: Bridges, Routers, Switches, and Internetworking Protocols - second edition;* 1999Addison Wesley

Stalling*ISDN and Broadband ISDN with Frame Relay and ATM;* 2002Prentice Hall

R. P. Davidson*Internetworking LANs: Operation, Design and Management;* 1992Artech House

A cura di S. Giorcelli (TILAB)*Collana ATM;* 1996UTET

S. Gianotti*ATM;* 1998HOEPLI

O. Kyas*ATM networks;* 2002Prentice Hall

Reti di computer in tecnica TCP/IP

U. Black*Internet Architecture: An Introduction to IP Protocols;* 2000Prentice Hall

A. G. Blank*TCP/IP Jumpstart: Internet Protocol Basics - second edition;* 2002Sybex

D. Comer*Internetworking with TCP/IP Vol.1: Principles, Protocols, and Architecture - fourth edition;* 2000Prentice Hall

B. A. Forouzan*TCP/IP Protocol Suite - third edition;* 1999McGraw Hill

W. R. Stevens*The Protocols (TCP/IP Illustrated, Volume 1);* 1994Addison Wesley

Reti di computer in tecnica Windows 2000

Microsoft*MCSE Training Kit - Microsoft Windows 2000 Professional;* 2000Microsoft Press

Microsoft*MCSE Training Kit - Microsoft Windows 2000 Server;* 2000Microsoft Press

Microsoft*Microsoft Windows 2000 Server Resource Kit;* 2000Microsoft Press

Microsoft*Microsoft Windows 2000 Server Administrator's Companion;* 2000Microsoft Press

Gestione di reti

Dr. Feit*SNMP, a guide to Network Management;* 1993McGraw Hill

W. Stallings*SNMP, SNMPv2, SNMPv3, and RMON 1 and 2 - third edition;* 1999Addison-Wesley

Divakara K. Udupa*TMN Telecommunications Management Network;* 1999McGraw Hill

Sicurezza e gestione della sicurezza

B. Schneier*Secrets and Lies: Digital Security in a Networked World;* 2000John Wiley < Sons

Giustozzi*Segreti spie codici cifrati;* 1999Apogeo

Stuart McClure*Hacking Exposed: Network Security Secrets < Solutions - third edition;* 2001Osborne McGraw Hill

B. Schneier*Applied Cryptography: Protocols, Algorithms, and Source Code in C - second edition;* 1995Wiley Computer Publishing

Zwicky*Building Internet Firewalls - second edition;* 2000O'Reilly < Associates

W. Stallings*Network and Internetwork Security: Principles and Practice - second edition;* 1999Prentice Hall

Siti

Infrastrutture per reti di calcolatori

ATM forum; http://www.atmforum.com

RFC 1490: multiprotocol over frame relay; http://www.ietf.org

Frame relay: normative, documenti di survey, glossario; http://www.frforum.com

Frame relay: approfondimenti sulla tecnica e sul protocollo;

http://www.cisco.com/univercd/cc/td/doc/cisintwk/ito_doc/frame.htm

Approfondimenti su reti frame relay in tecnica Ericsson; http://www.ericsson.se

Cisco: documentazione su architettura TCP/IP;
Cisco
http://www.cisco.com/univercd/cc/td/doc/cisintwk/ito_doc/ip.htm

Helmig (tutorial TCP/IP); http://www.wown.com/j_helmig/tcpip.htm

Specifiche di servizi Internet;
IETF
http://www.ietf.org/rfc.html

IEEE - Institute of Electrical and Electronics Engineers;
IEEE
http://www.ieee.org/portal/index.jsp

Forum DSL; http://www.dslforum.org/

Tutorial IBM su TCP/IP;
IBM
http://publib-b.boulder.ibm.com/Redbooks.nsf/RedbookAbstracts/gg243376.html?Open

Reti di computer in tecnica TCP/IP

RFC 791 (IP, Internet Protocol);
IETF
http://www.ietf.org/rfc/rfc0791.txt

RFC 793 (TCP, Transmission Control Protocol);
IETF
http://www.ietf.org/rfc/rfc0793.txt

RFC 950 (Procedura per il supporto del subnetting);
IETF
http://www.ietf.org/rfc/rfc0950.txt

RFC 1009 (Requirements for Internet Gateways);
IETF
http://www.ietf.org/rfc/rfc1009.txt

RFC 1517 (Classless Interdomain routing);
IETF
http://www.ietf.org/rfc/rfc1517.txt

RFC 1518 (Classless Interdomain routing);
IETF
http://www.ietf.org/rfc/rfc1518.txt

RFC 1519 (Classless Interdomain routing);
IETF
http://www.ietf.org/rfc/rfc1519.txt

RFC 1520 (Classless Interdomain routing);
IETF
http://www.ietf.org/rfc/rfc1520.txt

RFC 1631 (The IP Network Address Translator);
IETF
http://www.ietf.org/rfc/rfc1631.txt

RFC 768: User Datagram Protocol;
IETF
http://www.ietf.org/rfc/rfc768.txt

RFC 791: Internet Protocol;
IETF
http://www.ietf.org/rfc/rfc791.txt

RFC 792: Internet Control Message Protocol;
IETF
http://www.ietf.org/rfc/rfc792.txt

RFC 793: Transmission Control Protocol;
IETF
http://www.ietf.org/rfc/rfc793.txt

Reti di computer in tecnica Windows 2000

MCP Magazine Online;
Microsoft
http://www.mcpmag.com

Microsoft Web Site;
Microsoft
http://www.microsoft.com/italy

Microsoft Windows 2000 Web Site;
Microsoft
http://www.microsoft.com/italy/windows2000

Windows and .NET Magazine;
Microsoft
http://www.win2000mag.com

Microsoft Technet Technical Plus;
Microsoft
http://www.microsoft.com/italy/technet

Reti di computer in tecnica Unix/Linux

Documenti su Linux; http://www.linuxdoc.org/

Libri elettronici su Linux; http://www.oreilly.com/catalog

Aspetti di sicurezza per Linux; http://www.seifried.org/lasg/

Servizi di rete integrati in ambienti Microsoft (SAMBA); http://www.samba.org/

Documentazione su Linux; http://www.pluto.linux.it/ildp

Rivista su Linux; http://www.linuxjournal.com

FAQ sul tema Linux; http://www.tldp.org/FAQ/Linux-FAQ

Gestione di reti

Request For Comments (RFC);
IETF
http://www.ietf.org/rfc.html

International Telecommunication Union;
ITU
http://www.itu.int/home/index.html

SNMP Version 3 (snmpv3);
IETF
http://www.ietf.org/html.charters/snmpv3-charter.html

HP Openview (Piattaforma HP: overview);
Hewlett-Packard
http://www.openview.hp.com/

HP online manuals;
Hewlett-Packard
http://docs.hp.com/

Tivoli (Piattaforma IBM);
IBM
http://www-3.ibm.com/software/tivoli/

System management Server (Piattaforma Microsoft);
Microsoft
http://www.microsoft.com/italy/smserver/

Sicurezza e gestione della sicurezza

Sicurezza di Windows 2000;

Microsoft
http://www.microsoft.com/windows2000/security/

Smart Card;
Microsoft
http://www.microsoft.com/whdc/hwdev/tech/input/smartcard/default.mspx

Windows catalog, SmartCard Reader;
Microsoft
http://www.microsoft.com/windows/info/smart404.asp?404;http://www.microsoft.com/windows/catalog/wcbod

Cisco Router Access Control List;
Cisco
http://www.cisco.com/univercd/cc/td/doc/product/software/ios113ed/113ed_cr/secur_c/scprt3/scacls.htm

Sicurezza di Linux; http://www.seifried.org/lasg/

TCSEC - Trusted Computer System Evatuation Criteria;
http://www.radium.ncsc.mil/tpep/library/tcsec/index.html

CERT Coordination Center; http://www.cert.org

The International PGP Home Page; http://www.pgpi.org

PGP documentation; http://www.pgpi.org/doc/

Glossario

Access List : indica un elenco di regole di filtraggio del traffico dati, mediante cui è possibile abilitare l'accesso verso reti e applicazioni in maniera selettiva. Tipicamente una lista di accesso viene configurata su un router posto al confine fra reti, in modo da realizzare un punto di controllo del traffico di transito.

Accounting : funzionalità di gestione per una rete di TLC, relativa alla misurazione delle risorse fisiche e logiche impegnate in una rete, a scopo di addebito nei confronti dell'utilizzatore. In caso di reti private o locali, l'accounting costituisce uno strumento di ripartizione dei costi delle comunicazioni fra settori diversi di un'organizzazione.

ACE : Access Control Entry. Nella tecnologia delle reti Microsoft Windows rappresenta la generica entry di una tabella che definisce l'**ACL** configurata per una risorsa. Specifica gli eventi di protezione da controllare per un utente o un gruppo.

ACL : Abbreviazione per Access Control List, ovvero **Access List**. Nella tecnologia delle reti Microsoft Windows rappresenta l'elenco delle regole di accesso ad una risorsa (es. cartella del file system) e delle restrizioni attive su

tale risorsa. Nell'ambito del networking il termine indica un filtro sul traffico che un router o un firewall effettua a scopo di protezione di una rete, di un'applicazione, di una macchina.

Chiave privata : elemento della coppia di chiavi asimmetriche, destinato ad essere conosciuto soltanto dal soggetto titolare, mediante il quale si appone la firma digitale sul documento informatico o si decifra il documento informatico in precedenza cifrato mediante la corrispondente chiave pubblica.

Chiave pubblica : elemento della coppia di chiavi asimmetriche destinato ad essere reso pubblico, con il quale si verifica la firma digitale apposta sul documento informatico dal titolare delle chiavi asimmetriche o si cifrano i documenti informatici da trasmettere al titolare delle predette chiavi.

Chiavi asimmetriche : coppia inscindibile di chiavi crittografiche, una privata ed una pubblica, correlate tra loro, da utilizzarsi nell'ambito dei sistemi di validazione o di cifratura di documenti informatici

Cifratura : è un sistema di sicurezza usato per nascondere in modo digitale il contenuto di un messaggio in modo che soltanto il destinatario designato possa decodificarlo.

CMIP : Common Management Interface Protocol. Protocollo di gestione in ambito OSI. Standard ITU che definisce il formato dei messaggi e le procedure utilizzate per lo scambio di informazioni gestionali con lo scopo di esercire, amministrare, manutenere una rete e fornire servizi.

Dominio Windows : Insieme di computer forniti di sistema operativo Microsoft Windows (95, 98, NT, 2000, XP) collegati in rete. I servizi di rete forniti dalle macchine server sono tipicamente sotto il controllo di un'unica entità amministrativa (supervisore del dominio).

FAT : File Allocation Table. Tabella o elenco predisposto da alcuni sistemi operativi (es. Windows) per tenere traccia dello stato dei vari segmenti nel disco utilizzati per l'archiviazione dei file. **FAT** è l'acronimo di .

FAT32 : File system derivato dal file system della tabella di allocazione file (**FAT**). La tabella **FAT32** supporta dimensioni di cluster inferiori rispetto alla tabella **FAT**, migliorando l'efficienza dell'allocazione dello spazio sulle unità **FAT32**.

GUI : Graphic User Interface . Interfaccia grafica di Sistema Integratore, basata sullo standard Motif, garantisce un facile colloquio operatore-sistema

LDAP : Lightweight Directory Access Protocol. Protocollo Internet per i servizi di directory.

LDIF : **LDAP** File Interchange Format. Rappresenta un formato standard per la descrizione in formato testo, di una directory (elenco di informazioni) e delle entry in essa contenute.

LILO : LInux LOader. Si tratta di un componente software che consente di effettuare il caricamento del sistema operativo Linux su una macchina. Non ha dipendenze rispetto al file system e consente di effettuare il boot del kernet di Linux (e anche di altri sistemi operativi) da diverse periferiche di computer (floppy, CD, HD).

MIB : Management Information Base. Base di dati strutturata ad albero,contenente le informazioni e le variabili gestite relative ad elementi di rete.

NETBEUI : (NETBios Extended User Interface) protocollo per la realizzazione di reti di PC, appartenente all'architettura SNA ed usato come protocollo standard nelle reti Microsoft.

NetBIOS : (Network Basic Input Output System)

API : standard per le reti di personal computer.

NFS : (Network File System) protocollo sviluppato da SUN Microsystems che si appoggia sull'architettura di rete TCP/IP e consente ad un insieme di elaboratori di condividere i file system. E' spesso utilizzato dai client in ambienti di reti locali di computer per utilizzare porzioni del disco di un server come disco di rete. Prevede una parte client ed una parte server.

NIS : Network Information Service. Servizio Internet mediante il quale è possibile l'amministrazione centralizzata di un numero qualsiasi di file di qualsiasi tipo in rete.

PID : In ambito informatico indica il Process IDentifier, ossia l'etichetta attribuita ad un processo residente in un sistema di elaborazione.

RID : Relative IDentifier. Identificatore utilizzato in reti di computer Windows 2000. Costituisce la parte del **SID** che è variabile da risorsa a risorsa, nell'ambito del Dominio.

SID : Security IDentifier. Identificatore utilizzato in reti di computer windows NT e Windows 2000 per identificare univocamente un utente, oppure una macchina (una risorsa in reti windows 2000) in un dominio di computer Microsoft.

SMI : Structure of Management Information. Formalismo impiegato per descrivere le informazioni di gestione nell'ambito della gestione Internet/**SNMP**.

SMT : (Station ManagemenT) funzionalità di controllo di una stazione FDDI.

SNMP : Simple Network Management Protocol. Protocollo per lo scambio di informazioni di gestione in ambiente TCP/IP. Ad oggi molte implementazioni di **SNMP** consentono la gestione di ambiti anche diversi da reti TCP/IP.

VLAN : Virtual Local Area Network. Rete locale virtuale, realizzata mediante apparati intelligenti in grado di partizionare una molteplicità di stazioni in

sottoinsiemi logici ognuno dei quali si comporta come se fosse una LAN distinta e di associare ciascun calcolatore ad una delle LAN virtuali prestabilite.

Autori

Hanno realizzato il materiale di questo modulo:

Prof. Franco Callegati

Franco Callegati è professore associato di Reti di Telecomunicazioni presso il Dipartimento di Elettronica, Informatica e Sistemistica (D.E.I.S.) dell'Università di Bologna. Presso la Facoltà di Ingegneria di Bologna prima ed ora presso la Facoltà di Ingegneria di Cesena ha tenuto e tiene corsi di base di Reti di Telecomunicazioni e corsi avanzati su teoria del traffico e progettazione di reti. Si interessa di problematiche di dimensionamento e progettazioni di reti di telecomunicazione a larga banda e la sua attività di ricerca più recente ha come oggetto le reti ottiche ad altissima velocità, argomento sul quale ha pubblicato numerosi lavori, partecipando a progetti di ricerca nazionali ed internazionali con ruoli di coordinamento.

Dott.Ing. Walter Cerroni

Walter Cerroni ha ottenuto il titolo di Dottore di Ricerca in Ingegneria Elettronica ed Informatica presso l'Università di Bologna. Svolge attività di ricerca nell'ambito dell'analisi di prestazioni di reti di telecomunicazioni e del progetto di architetture per commutazione ottica a pacchetto, collaborando con il gruppo di Reti di Telecomunicazioni dell'Università di Bologna. Svolge attività didattica come collaboratore per i corsi di Reti di Telecomunicazioni nell'ambito sia delle Lauree in Ingegneria dell'Informazione che del Master in *Management* e *Information Technology* dell'Università di Bologna, oltre a fornire servizi di consulenza a ditte ed enti di formazione nel settore delle tecnologie dell'informazione.

Modulo realizzato sulla base di materiali prodotti nell'ambito di un piano di formazione di 12.000 tecnici delle pubbliche amministrazioni e messi a disposizione del MIUR dall'Autorità per l'Informatica nella Pubblica Amministrazione (AIPA).