Università di Pisa

Facoltà di Ingegneria

Corso di laurea in Ingegneria Elettronica

Tesi di laurea

Studio di un sistema di autorizzazione ed autenticazione centralizzato di sistemi eterogenei

Relatori:
Prof. Francesco Marcelloni
Prof. Giuseppe Anastasi
Ing. Fabio Maria Teti

Candidato:
Giulio Polenta

Anno Accademico 2004-2005

Riassunto analitico

Lo studio ha l'obiettivo di formulare una soluzione per la progettazione di sistemi di autenticazione ed autorizzazione centralizzata, partendo dall'analisi dello stato dell'arte di tali tecnologie. I risultati ottenuti forniscono le linee guida per la realizzazione di una infrastruttura per la gestione centralizzata di sistemi e servizi eterogenei in aziende di grosse dimensioni.

Prefazione

La complessità in continua crescita delle infrastrutture informatiche, la diversificazione tecnologica degli ambienti applicativi, l'utenza sempre più numerosa e la necessità di individuare con precisione compiti e responsabilità nel trattamento delle informazioni determinano l'esigenza di identificare ed autorizzare gli operatori in modo sicuro e robusto, evitando la frammentazione delle credenziali che deriva dalla necessità di controllare dati e servizi su sistemi distribuiti.

Questa esigenza da sempre sentita nella gestione dei servizi informatici è stata accresciuta sensibilmente dalla recente normativa sulla tutela dei dati personali che individua precise norme nella gestione dei profili di accesso del personale incaricato del trattamento dei dati.

A tale esigenza si è risposto con sistemi di identificazione più sicuri, ma sono state trascurate le criticità che derivano dalla gestione di un numero elevato di utenti che accedono a servizi eterogenei. Il controllo della validità delle credenziali, l'aggiornamento dei diritti di accesso su sistemi eterogenei e distribuiti sono processi tanto più gravosi quanto maggiori sono le dimensioni dell'infrastruttura informatica e tanto più pericolosi, per l'introduzione di errori di configurazione, quanto più la frammentazione dei servizi conduce ad una ridondanza delle credenziali su diversi sistemi.

Un sistema di autenticazione centralizzato ed integrato al quale si collegano tutti i servizi distribuiti per identificare gli utenti ed autorizzarli allo sfruttamento delle risorse consente una gestione ottimizzata e sicura dei profili di accesso, rendendo unica la banca dati che contiene le informazioni di autenticazione ed autorizzazione ed eliminando i rischi precedentemente descritti.

Lo studio si prefigge una disamina dei sistemi di autenticazione al fine di evidenziarne i pregi ed i limiti, nel tentativo di determinare come e se sia possibile migliorarli ed integrarli in strutture complesse ed eterogenee, al fine di ottenere un sistema di autenticazione ed autorizzazione realmente efficace e di facile gestione.

RINGRAZIAMENTI

Un doveroso e sentito grazie a tutto il personale dell'U.O.S.I. per la sua disponibilità e l'affetto con cui ha saputo accogliermi.

Grazie a Fabio Teti per l'opportunità che mi ha offerto, per il tempo che mi ha dedicato e per gli spunti che ha saputo darmi.

Grazie ai miei genitori per la pazienza che hanno dimostrato.

Grazie a Odile per il supporto e l'affetto.

Indice generale

1 Introduzione

In questo capitolo vengono esposte le motivazioni che portano alla necessità di un sistema di autenticazione, vengono quindi introdotti alcuni concetti di base ed alcune caratteristiche che un sistema di autenticazione deve possedere.

Vengono altresì illustrati i vantaggi che è possibile conseguire utilizzando un sistema di autenticazione ed autorizzazione centralizzato ben strutturato.

1.1 Necessità di un sistema di autenticazione

I calcolatori ormai da diversi anni non sono più isolati ma connessi ad una rete che, se inizialmente aveva una dimensione ridotta all'ambito di un ufficio, si è via via allargata sino a raggiungere con Internet dimensione mondiale. I protocolli di comunicazione utilizzati sono con il tempo diminuiti ed attualmente il *TCP/IP* è di gran lunga il più diffuso; tale protocollo è alla base di Internet come di reti domestiche, la versione attualmente in uso è la ipv4 mentre conosce una sempre maggiore diffusione la più recente ipv6.

Lo scopo fondamentale di una rete di calcolatori è la condivisione delle informazioni, basta ripercorrere la storia dell'evoluzione di Internet per convincersene. Nel 1969 con i finanziamenti del ministero della difesa degli Stati Uniti nasce Arpanet (Advanced Research Project Agency Net) al fine di verificare le possibilità delle tecnologie di networking collegando molte università e centri di ricerca nordamericani; il primo protocollo che permette lo scambio di informazioni è del 1971, si tratta del *File Transfer Protocol (FTP)*, che permette lo scambio di file, l'anno successivo venne integrato con la possibilità di scambiare messaggi di posta elettronica. Negli anni seguenti sono stati creati altri modi di comunicare sfruttando Internet, quello di maggior successo è senza dubbio il *World Wide Web (WWW)*, le cui potenzialità hanno portato all'enorme sviluppo di Internet che si è avuto negli ultimi anni.

La diffusione sempre più capillare dei calcolatori ha cambiato la natura stessa di Internet, inizialmente utilizzata nel ristretto ambito dei ricercatori universitari per lo scambio di informazioni a carattere scientifico, e come tali liberamente fruibili, e quindi considerata un ambiente “amico”, trasformandola in una comunità di enormi dimensioni frequentata anche da soggetti poco raccomandabili e con informazioni non necessariamente libere ed il cui accesso deve essere consentito solo a chi ne ha diritto.

Pertanto, la necessità di effettuare il controllo dell'accesso e verificare i diritti dei navigatori di

Internet rende inevitabilmente i sistemi di autenticazione sempre più diffusi e sempre più di frequente utilizzo, come descritto nei semplici esempi riportati in seguito:

- Le reti locali a carattere aziendale sono ormai una presenza costante; la possibilità di condividere file ed i sistemi di messaggistica hanno permesso un aumento dell'efficienza nel lavoro e sono una necessità anche in aziende a bassa tecnologia, ma la condivisione delle informazioni intrinsecamente porta alla gestione degli accessi alle medesime.
- La possibilità di accedere alla rete aziendale da una qualsiasi postazione collegata ad Internet ha dato impulso allo sviluppo di *applicazioni distribuite*, questo tipo di applicazione è generalmente realizzato sfruttando il paradigma *client-server*: una applicazione residente su una macchina, situata nel centro di elaborazione aziendale, accetta e gestisce richieste provenienti da applicazioni remote; queste applicazioni vengono utilizzate nelle aziende con crescente successo per il collegamento di sedi remote, di agenti sparsi sul territorio ed altro. Anche il modo di lavorare è stato modificato dalle possibilità di collegamento offerte dalla rete, sono sempre più numerosi i casi di collaborazione tra persone che risiedono anche a migliaia di chilometri di distanza. Esempio lampante di questo nuovo modo di lavorare è lo sviluppo del software open source: la possibilità di condividere i file sorgente con l'utilizzo di sistemi quali il *Concurrent Versions System (CVS)* ha permesso a grandi comunità di sviluppatori di lavorare insieme e senza intralciarsi, e realtà ormai consolidate come Linux, Apache, Samba devono la loro esistenza alla presenza della rete: ma il garantire l'identità di chi accede al deposito dei sorgenti è fondamentale, si sono verificati casi di manomissione dei sorgenti ma anche di furto degli stessi in progetti non open source. Il telelavoro, ovvero la possibilità di lavorare da remoto, viene da molti considerato la modalità di lavoro del futuro e certamente sarà foriera di un incremento dell'utilizzo di applicazioni distribuite e della necessità di autenticare gli utenti.
- Soluzioni per l'automazione dei servizi e del commercio in ambito aziendale conoscono uno sviluppo crescente, ad esempio l'applicazione che gestisce il magazzino di un supermercato può richiedere ai fornitori la merce quando le scorte scendono al di sotto di un livello critico comunicando direttamente con l'applicazione che gestisce gli ordini del fornitore, per ottenere questo risultato è necessario un linguaggio comune ed allo scopo è

stato sviluppato *XML*, ma è assolutamente fondamentale l'identificazione sicura dei sistemi in comunicazione.

- Nell'ambito delle applicazioni condivise hanno avuto negli ultimi anni uno sviluppo tumultuoso quelle web-based che utilizzano come interfaccia delle pagine *HTML*, l'utilizzo di tali applicazioni è diventata pratica comune per le aziende ma anche per le strutture pubbliche che le utilizzano per fornire servizi di facile utilizzo ai cittadini. Il customer care viene sempre più indirizzato verso lo strumento informatico a discapito di mezzi più tradizionali, e più costosi come i call-center. Anche i servizi messi a disposizione degli utenti sono aumentati e diventati più sofisticati, solo per citare pochi esempi: si possono acquistare biglietti del treno e di teatro, richiedere l'allacciamento alla linea telefonica, conoscere gli esami sostenuti e la situazione delle tasse all'università, gestire il conto corrente, acquistare titoli di borsa, ma per il corretto utilizzo di tutti questi servizi è necessario identificare con ragionevole sicurezza le parti coinvolte.
- La possibilità di accedere ad Internet mediante telefonia cellulare e la convergenza di computer palmari e telefoni verso apparecchi integrati fa intravedere la prospettiva di nuovi servizi e con essi la presenza ancora più massiccia di informazioni riservate da proteggere in modo adeguato.

Da queste considerazioni risulta evidente che la necessità di autenticare gli utenti ed i servizi con ragionevole certezza è un requisito di base di una infrastruttura informatica. E che tutto il sistema di gestione dell'autenticazione è un punto critico per il corretto funzionamento di tutti gli altri servizi disponibili in rete.

1.2 Necessità di un sistema di autorizzazione

Un utente è autenticato quando, fornendo delle credenziali, è riuscito a garantire la propria identità, ma l'autenticazione in quanto tale è un mezzo e non un fine, ci si autentica per poter svolgere una qualche attività non liberamente fruibile.

In un sistema informatico generalmente sono disponibili più servizi e gli utenti che possono accedervi sono molteplici, inoltre è probabile che non tutti gli utenti abbiano uguali caratteristiche; in generale vi sono utenti che hanno la possibilità di accedere ad alcuni servizi mentre altri sono loro preclusi, per una gestione sicura di un sistema informatico è necessario gestire i permessi dei singoli utenti.

I permessi possono essere definiti come una relazione tra utenti e risorse, ogni permesso collega un utente ad una risorsa ed eventualmente fornisce anche informazioni sulla modalità di accesso alla stessa, il percorso che porta un utente all'accesso delle risorse è visibile in Figura 1. Un esempio può essere fornito dai file system evoluti: i permessi per l'accesso alle

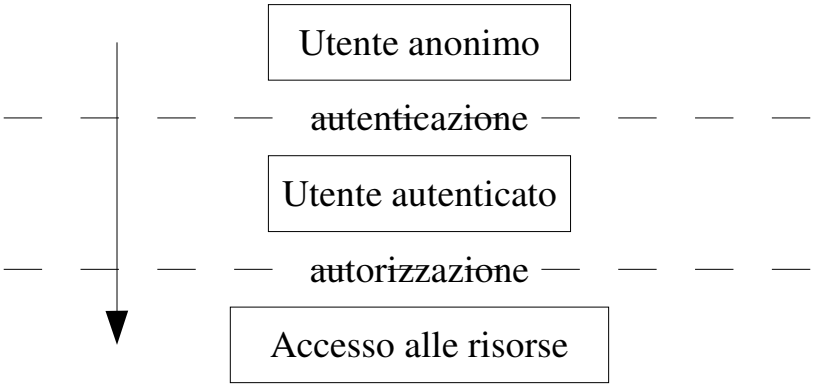

Figura 1 - Procedura di accesso alle risorse

risorse, ovvero ai file, sono caratterizzati da un proprietario ed un gruppo e dalle modalità di accesso (lettura, scrittura, esecuzione). Ad un utente autenticato viene associato uno *userid* ed un gruppo e l'accesso alle risorse è regolato dalla relazione: (utente==proprietario) or (gruppo utente==gruppo proprietario), se la relazione è soddisfatta l'accesso è consentito secondo le modalità specificate nel permesso.

L'amministrazione dei permessi è indispensabile per garantire la sicurezza dei sistemi e dei dati in essi contenuti, l'aggiornamento dei permessi è una attività continua e le variazioni del personale e delle attività svolte si deve riflettere in variazioni sui permessi.

Limitare le possibilità di utilizzo dei sistemi informatici in base al ruolo ricoperto dagli utenti porta benefici in molti aspetti della vita informatica ed economica, alcuni esempi:

- L'utilizzo sempre più diffuso di basi di dati per la gestione aziendale comporta una gestione degli accessi; eventuali accessi non autorizzati possono portare a furti o manipolazione delle informazioni anche con intenti dolosi come per lo spionaggio industriale che potrebbe avere come obiettivi i dati dei clienti o dei progetti: queste intrusioni possono rivelarsi molto costose. La perdita di informazioni protette dalla privacy può avere conseguenze sul piano legale e amministrativo.
- La protezione dei sistemi informatici tramite il controllo degli accessi da malfunzionamenti dovuti a modifiche per errore o dolo, ha benefici effetti sui livelli di servizio, non deve nemmeno essere trascurata la protezione delle workstation utilizzando utenti con privilegi

diversi al fine di evitare frequenti ricorsi all'assistenza per manomissioni dovute ad imperizia o all'utilizzo di applicazioni non direttamente legate al lavoro.

- L'utilizzo di risorse del sistema informatico come l'accesso ad Internet per scopi non lavorativi porta a maggiori costi e minore efficienza del personale.

Un vantaggio collaterale dell'autorizzazione è la possibilità di mantenere traccia delle attività degli utenti, utilizzando ulteriori accorgimenti è possibile sapere chi ha usufruito di una risorsa o di una informazione e cosa ne ha fatto, con l'ulteriore vantaggio della garanzia che soltanto gli utenti autorizzati possono svolgere certe attività e questo permette di affidare incarichi e conseguenti responsabilità.

La garanzia di un corretto utilizzo dei sistemi informatici è legata a molti aspetti della struttura aziendale e non solo prettamente informatici: lo studio di procedure di accesso ai servizi ed ai dati, e la formazione del personale sulle stesse e sui rischi connessi ad un utilizzo improprio sono la base minima per la sicurezza. Anche il sistema più sofisticato e correttamente configurato può essere messo in crisi dall'utilizzo di password banali o da computer lasciati incustoditi, è opportuno che i permessi rispecchino fedelmente la realtà aziendale e che vengano mantenuti costantemente aggiornati; cambiamenti nella struttura del personale o nelle mansioni possono e devono comportare modifiche agli utenti o ai permessi loro associati.

In strutture di grandi dimensioni è probabile che vi siano molti utenti con caratteristiche analoghe, può allora risultare utile, per una semplificazione della gestione, creare dei profili per utenti tipo, risulterà così più semplice creare delle procedure standard per le operazioni di routine.

1.3 Struttura generale di un sistema di autenticazione

Un sistema di autenticazione ed autorizzazione è costituito da un insieme di dati che caratterizzano gli utenti e le risorse e dai meccanismi di accesso ai dati, tale struttura è riassunta in Figura 2, più in dettaglio:

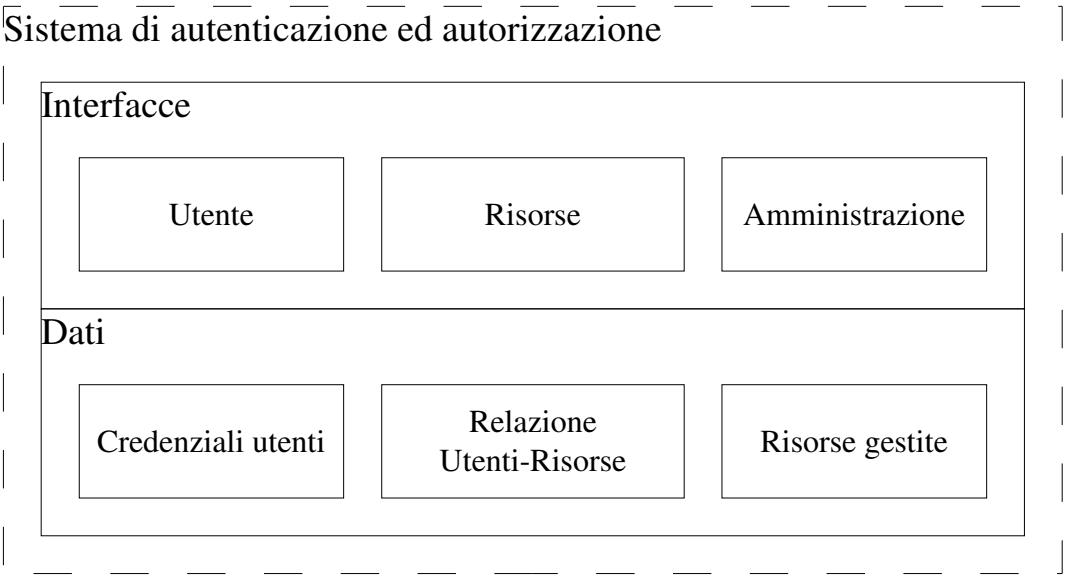

Figura 2 - Struttura del sistema di autenticazione ed autorizzazione

- L'interfaccia utente può essere utilizzata per fornire all'utente informazioni sul suo stato attuale, ad esempio i sistemi sui quali ha accesso, lo stato di eventuali richieste come l'attivazione di nuovi servizi o la richiesta di certificati digitali. Deve inoltre permettere la modifica delle password, per ottemperare alla normativa sulla privacy, ed eventualmente di ulteriori dati personali. Questa interfaccia deve poter essere utilizzata da tutti gli utenti ed è quindi necessario che sia facilmente raggiungibile e di semplice utilizzo, per cui la soluzione più conveniente probabilmente è realizzarla come applicazione web-based.
- Le risorse utilizzano il sistema di autenticazione ed autorizzazione per richiedere se accogliere o meno le richieste provenienti dagli utenti. Generalmente le risorse sono molteplici ed eterogenee, per rispondere all'eterogeneità il sistema di autenticazione deve essere centralizzato e capace di erogare il servizio per tutti i sistemi in funzione.
- L'interfaccia di amministrazione deve permettere in maniera semplice ed efficace la gestione degli utenti e delle risorse, data la delicatezza del compito, deve essere adeguatamente protetta da accessi indesiderati.
- Il sistema di autenticazione deve “conoscere” gli utenti, necessita quindi delle credenziali e di informazioni per l'accesso e la configurazione del servizio. Ad esempio un sistema Unix per dare accesso ad un utente deve conoscere la *home directory,* la *shell* ed altro.
- L'insieme delle risorse accessibili è necessario al sistema di autorizzazione e all'applicazione di gestione.
- Le relazioni utente-risorse sono il cuore del sistema di autorizzazione che le utilizza per

stabilire i diritti di accesso degli utenti.

1.4 Sistema di autenticazione centralizzato

Gli apparati connessi in rete prevedono generalmente la possibilità di limitare gli accessi mediante autenticazione, in una infrastruttura informatica non banale sono presenti vari sistemi, quali: firewall, server con vari sistemi operativi e vari servizi, workstation con vari sistemi operativi, stampanti di rete e generalmente ognuno di questi apparati ha un sistema di autenticazione locale e la possibilità di accedere a sistemi di autenticazione esterni. La possibilità di utilizzare un servizio di autenticazione centralizzato comporta numerosi benefici:

- Un servizio di autenticazione centralizzato è caratterizzato da una base dati unica, l'univocità dei dati semplifica la gestione evitando la duplicazione di informazioni ed eliminando il rischio del disallineamento, il responsabile della sicurezza può sapere facilmente i permessi accordati agli utenti su tutte le risorse disponibili.
- Ulteriori benefici si hanno dalla semplificazione nella gestione dei sistemi che è sgravata della gestione degli utenti, sistemi in cluster hanno in maniera trasparente lo stesso set di utenti, un unica struttura dedicata all'autenticazione riduce il tempo necessario agli amministratori per rendere sicuro l'accesso ai sistemi concentrando le loro attenzioni sul sistema centralizzato.
- Un sistema di autenticazione centralizzato comporta anche dei rischi: se viene compromesso l'intero sistema informatico può essere violato ed anche un disservizio si ripercuote su tutto il sistema, ne consegue che la robustezza, il dimensionamento e la sicurezza del sistema di autenticazione devono essere curate con la massima attenzione.

Conseguenza della centralizzazione è la necessità che il sistema di autenticazione sia accessibile dal maggior numero di apparati e servizi, per ottenere questo risultato devono essere supportati i protocolli a maggiore diffusione, con l'ulteriore vantaggio di diminuire i costi e semplificare la migrazione di infrastrutture preesistenti.

1.5 Gestione del sistema di autenticazione

La gestione del sistema di autenticazione è costituita dalle normali pratiche sistemistiche per le quali si rimanda ai testi specialistici e dalla gestione dei dati e quest'ultima può essere affidata ad un responsabile che può anche non essere l'amministratore di sistema.

I dati del sistema di autenticazione devono essere costantemente allineati con la realtà. Modifiche nella struttura informatica o nell'organigramma che non vengono riportate prontamente nella base dati possono essere foriere di conseguenze negative che spaziano dai piccoli malfunzionamenti a veri e propri disastri informatici.

La gestione del sistema di autenticazione consiste in massima parte nell'aggiornamento dei dati. Si può distinguere una gestione corrente ed una straordinaria: la gestione corrente riguarda gli account ed i permessi, tipicamente la creazione e la distruzione di account o la modifica di alcuni parametri, la gestione straordinaria si ha quando sono necessarie modifiche all'infrastruttura informatica, ad esempio nuovi server o modifiche delle politiche di sicurezza che comportano un aggiornamento del sistema.

La gestione corrente, che è costituita da procedure ben determinate, può essere semplificata dall'utilizzo di applicazioni appositamente realizzate. In tal modo è possibile affidare la gestione a personale non specializzato e che non necessita di una conoscenza approfondita delle tecnologie. L'utilizzo di una applicazione consente verifiche dei dati inseriti ed in tal modo è possibile ridurre i rischi.

Per la gestione straordinaria, che non può essere ristretta a procedure standard e necessita di conoscenze approfondite della infrastruttura informatica e delle tecnologie utilizzate, conviene affidarsi ad un responsabile che segua l'intero sistema di autenticazione e collabori con i responsabili degli altri sistemi.

I rischi di una cattiva gestione possono essere divisi in tre categorie: utilizzabilità delle risorse, sicurezza dei sistemi, sicurezza delle informazioni.

Il sistema di autenticazione ed autorizzazione utilizza informazioni sugli account, le credenziali, le risorse ed i permessi. Cerchiamo di delineare alcuni problemi che si possono verificare per la presenza di dati non corretti e l'impatto che possono avere:

- Accesso negato.

  Sicurezza sistema: nullo Sicurezza informazioni: nullo Danno: da lieve a grave

  Il non poter accedere ad una risorsa non comporta rischi per la sicurezza, il danno può essere lieve se riguarda un numero limitato di utenti ma anche molto grave se viene negato l'accesso ad un intero sistema di rilevante importanza, con il blocco delle attività.

  Se è coinvolto un solo utente può essere dovuto a dati sbagliati nell'account o nei permessi,

se è un intero sistema a risultare inaccessibile è più probabilmente legato ad una cattiva configurazione delle risorse.

- Accesso con account di altri utenti ma con gli stessi privilegi

  Sicurezza sistema: nullo Sicurezza informazioni: media Danno: da medio a grave

  Se un utente riesce ad accedere con un account diverso dal suo, potrebbe accedere ad informazioni non di sua pertinenza con eventuali danno della privacy e furto di informazioni riservate.

  Potrebbe essere dovuto ad informazioni scorrette negli account ad esempio omonimie che portano allo stesso userid o credenziali incustodite.

- Accesso con account di altri utenti e privilegi diversi

  Sicurezza sistema: grave Sicurezza informazioni: grave Danno: estremamente grave

  Se un utente riesce ad accedere con un account con privilegi diversi da quelli di sua pertinenza potrebbe arrivare alla manomissione del sistema, al furto o alla modifica di informazioni protette.

  Un sistema di autenticazione progettato correttamente deve evitare che la gestione corrente possa portare a problemi di questa gravità.

2 Tematiche e tecnologie

In questo capitolo vengono descritti gli aspetti principali coinvolti nella realizzazione di un sistema di autenticazione rispondente alle caratteristiche tracciate nel capitolo precedente. Vengono inoltre presentati argomenti accessori, ma non di secondo piano come le normative di legge che regolano la privacy.

2.1 Privacy e sistemi di autenticazione

La normativa sulla privacy attualmente in vigore considera molti aspetti dei sistemi informatici. In questo paragrafo vengono brevemente descritti soltanto gli aspetti collegabili all'autenticazione ed all'autorizzazione.

La legge prevede che il trattamento dei dati sia consentito solo previa autenticazione mediante segreto condiviso od un meccanismo più sofisticato. Con tale obbligo non è difficile prevedere un crescente utilizzo dei sistemi di autenticazione.

La gestione degli account deve soggiacere ad alcuni vincoli di legge. Tali vincoli possono essere implementati mediante procedure di gestione eventualmente automatizzate con vantaggi facilmente immaginabili. Nel seguito verranno esposti i vincoli e ne viene descritta una possibile soluzione:

1. La password deve essere a conoscenza solo del legittimo incaricato e devono essere previste opportune procedure per la modifica della password.
   - L'applicazione di gestione deve prevedere la generazione di una password temporanea che l'incaricato provvederà a modificare al primo accesso.

     Un'altra possibile soluzione è la generazione casuale della password senza che venga resa disponibile al personale che amministra gli account.
2. La password deve essere modificata almeno con la scadenza programmata definita dalla legge e che dipende dal tipo di dato trattato, 6 mesi per i dati personali e 3 mesi per i dati sensibili.
   - Per ottemperare a questa richiesta l'applicazione di gestione deve rendere possibile la modifica e prevedere la disabilitazione degli account con password scadute; sistemi di avviso dell'imminente scadenza e il garantire la possibilità di modifica della password anche se già scaduta sono delle possibilità interessanti.
3. La password deve essere costituita da almeno 8 caratteri.

- L'applicazione di gestione deve verificare la validità della password.

  Una possibile estensione oltre gli obblighi di legge ma che incrementa la sicurezza è la verifica della difficoltà con la quale la password può essere violata; tale verifica si può basare sul tipo di caratteri utilizzati, sull'utilizzo di parole reperibili nel dizionario ed altro.

4. Il medesimo codice di identificazione non può essere utilizzato da più incaricati anche in tempi diversi.
- Potrebbe risultare interessante la possibilità di disabilitare gli account anziché cancellarli, in tal modo durante l'inserimento di un nuovo account sarebbe possibile verificarne l'eventuale precedente utilizzo.

5. Se per gli incaricati sono previsti profili di autorizzazione in ambiti diversi, deve essere utilizzato un sistema di autorizzazione per limitare l'accesso ai soli dati necessari al particolare trattamento.
- Rende necessario l'utilizzo di un sistema di autorizzazione.

2.2 Autenticazione

Le varie modalità di garantire l'identità si possono ricondurre a tre: è possibile autenticarsi mediante qualcosa che si conosce, qualcosa che si possiede o "qualcosa di noi". Esempi di identificazione del primo tipo sono i codici che vengono utilizzati per l'accesso a call center, o la parola d'ordine di ogni buon romanzo di spionaggio, nel secondo tipo possiamo annoverare le chiavi di casa, i tagliandi da esporre sul parabrezza dell'automobile o sulla giacca per accedere ad aree riservate, l'ultimo metodo è riconducibile alle impronte digitali, l'esame del DNA o la firma.

Nei successivi paragrafi verranno introdotti alcuni concetti al fine di definire le caratteristiche di un sistema di autenticazione.

2.2.1 Metodi di autenticazione

L'autenticazione di un utente su un sistema informatico non differisce se non nei mezzi da quanto visto, vediamo più in dettaglio i metodi di autenticazione di uso comune.

1. Qualcosa di conosciuto: questa modalità nei sistemi informatici si traduce nell'utilizzo delle password, è il sistema più utilizzato ma anche il più vulnerabile.

   Tra i rischi cui vanno soggette le password vi sono gli attacchi di forza bruta[1] che possono

1 Gli attacchi di forza vengono portati provando un numero molto elevato di password, per rendere più

essere resi più efficaci dalla tendenza degli utenti a scegliere password facili da ricordare, inoltre molti protocolli le trasmettono in chiaro e possono essere intercettate nel transito. Non sono particolarmente gradite dagli utenti che devono memorizzarle ed oltretutto devono avere periodi di validità limitata per aumentarne la sicurezza, con conseguente aggravio per la memorizzazione. Per limitare i rischi di questa modalità di autenticazione, sono stati introdotti dei sistemi come il one time password 2.2.10.

2. Qualcosa di posseduto: su un dispositivo vengono memorizzate informazioni utilizzate dal sistema di autenticazione per identificarlo, i supporti utilizzati sono tipicamente smart card, chiavi hardware, cd-rom. Il livello di sicurezza garantito è superiore a quello delle password, ma i supporti possono essere persi, dati ad altri e in alcuni casi sono di facile duplicazione, non possono quindi garantire che l'utilizzatore sia effettivamente colui che ne ha diritto. Tra i difetti non va trascurato il costo che nel caso dell'utilizzo di smart card, che sono il supporto più diffuso, comprende l'acquisto dei supporti ma anche dei lettori.
3. Informazioni biometriche: si possono utilizzare impronte digitali, retina, voce ed altro. Sono il sistema più sicuro per garantire l'identità, ma anche il più costoso ed il meno pratico.

Per garantire una maggiore sicurezza è possibile associare più metodi, nella letteratura si parla di *one two* o *three factor authenticathion* a seconda di quanti metodi vengono utilizzati contemporaneamente, ad esempio le smart card contenenti certificati digitali protetti da password utilizzano il primo metodo (password) ed il secondo (possesso della smart card) si tratta quindi di two factor authentication.

Riferimenti: [RFC1704], [draft-iab-auth-mech-03].

2.2.2 Autenticazione in sistemi UNIX e Linux

I meccanismi di autenticazione nei sistemi Unix si sono evoluti nel tempo al pari dell'ambito di utilizzo del sistema. I primi metodi sono stati sviluppati quando UNIX era un sistema centralizzato al quale accedevano dei terminali su linea seriale, mentre ora viene utilizzato in reti di dimensioni mondiali e non è pensabile che le esigenze di sicurezza e flessibilità richieste al sistema di autenticazione siano rimaste le stesse.

---

efficace questo tipo di attacchi vengono utilizzati dei vocabolari. Ci si può difendere utilizzando password lunghe e/o generate casualmente.

2.2.2.1 File locale

Il sistema più antico utilizza come deposito per le credenziali il file */etc/passwd*, questo file è ancora necessario per il corretto funzionamento del sistema. Ogni riga del file contiene le informazioni relative ad un singolo utente e sono così strutturate:

`account:password:UID:GID:GECOS:directory:shell`

il significato dei singoli campi è descritto nella Tabella 1.

Campo	Descrizione
`account`	Il nome utente nel sistema, deve essere unico e con caratteri minuscoli.
`password`	La password crittografata dell'utente. Vi può essere un asterisco se vengono utilizzate le shadow password.
`UID`	User Identification number, un numero utilizzato per identificare l'utente.
`GID`	Group Identification number, numero del gruppo di appartenenza.
`GECOS`	General Electric Comprehensive Operating System, il nome del campo è storico e viene attualmente utilizzato per il nome dell'utente o per un commento.
`directory`	La home directory dell'utente.
`shell`	La shell utilizzata dall'utente.

Tabella 1 - Campi del file /etc/passwd

I diritti di accesso al file */etc/passwd* consentono la scrittura solo per l'utente root, ma devono necessariamente prevedere l'accesso in lettura per tutti gli utenti, perché le informazioni relative al GID e all'account devono essere liberamente accessibili. Questo rende possibile a tutti gli utenti la lettura delle password, e siccome gli algoritmi di crittografia utilizzati sono reversibili il sistema è soggetto ad attacchi di forza bruta. Per eliminare questo rischio sono state sviluppate le *shadow password*, con tale sistema il campo *password* di *passwd* contiene un asterisco e la password è contenuta nel file /etc/shadow accessibile in lettura solo da root.

2.2.2.2 Pluggable Authentication Modules (PAM)

PAM è stato sviluppato dalla *Open Software Foundation* in collaborazione con SunSoft, ed implementa un meccanismo modulare che permette la facile integrazione con vari sistemi di autenticazione in maniera trasparente al resto del sistema. Le informazioni necessarie al sistema oltre quelle di autenticazione vengono ottenute dai file standard di sistema. Nei

sistemi UNIX moderni è possibile utilizzare *nsswitch* (Name Service Switch), che permette di definire quali database di sistema utilizzare, tra questi vi sono *passwd* e *groups*, che possono essere ridiretti in modo da non utilizzare più i file di sistema.

Con PAM non solo è possibile utilizzare più sistemi di autenticazione contemporaneamente ma anche differenziarne l'utilizzo a seconda del servizio al quale si tenta di accedere.

La configurazione del PAM permette di modificare quattro aspetti dell'autenticazione:

- Account: viene utilizzato per la gestione dell'account, ad esempio verificare se la password è scaduta.
- Authentication: in base al meccanismo di autenticazione utilizzato identifica l'utente.
- Password: stabilisce come gestire la password, ad esempio se e come l'utente può modificare la password.
- Session: per operazioni che devono essere svolte subito prima o subito dopo che l'utente ha avuto accesso ad un servizio, ad esempio scrittura sui file di log.

2.2.3 Autenticazione nei sistemi Windows

I modelli utilizzati per l'autenticazione nei sistemi Windows negli anni hanno subito delle evoluzioni. Il cambiamento più importante si è avuto con l'avvento di Windows 2000 che, pur mantenendo la compatibilità con i sistemi precedenti, implementa una architettura diversa.

La prima condivisione di risorse utilizzata nei sistemi Microsoft, si basa sul protocollo SMB/CIFS che è stato sviluppato dalla IBM nel 1985, e con innumerevoli variazioni è presente anche nei sistemi operativi più moderni. L'evoluzione del protocollo nel corso degli anni è stata frammentaria e ha seguito percorsi poco lineari. Attualmente sono implementate un centinaio di operazioni diverse alcune delle quali hanno funzioni che si sovrappongono e parametri non documentati, l'IETF e la Storage Network Industry Association (SNIA) stanno tentando di riordinare e razionalizzare il protocollo.

SMB/CIFS si basa sul paradigma client-server, i client attualmente disponibili oltre a quelli Microsoft sono: “The PATHWORKS” e i progetti open source: smbclient del progetto Samba e la libreria in Java jCIFS, i server sono stati realizzati da tutti i più importanti produttori, tra questi: The PATHWORKS della Digital, LAN Manager per OS/2, SCO ed altri, VisionFS della SCO , TotalNET Advanced Server della Syntax, Advanced Server for UNIX della AT&T, LAN Server for OS/2 della IBM ed il già citato Samba server.

Il protocollo SMB/CIFS prevede due metodi di accesso: lo *user level* che utilizza password e nome utente ed il server permette l'accesso al client solo previa autenticazione dell'utente, in tal modo è possibile al server la gestione di liste di accesso. Questo schema è utilizzato da Windows NT e successivi, il client che si autentica riceve un *token* che può essere poi utilizzato per le successive richieste al server evitando il ricorso ad ulteriori autenticazioni. Lo *share level* prevede la protezione di una risorsa mediante password, ed è utilizzato dalle versioni client di Windows come il 95 ed il 98. Con l'introduzione di Windows NT è stata realizzato anche un sistema di autenticazione centralizzato basato sui domini, la struttura prevede server che erogano i servizi e server di autenticazione detti Primary Domain Controller (PDC), vi possono essere più domini che attraverso relazioni di fiducia condividono utenti e risorse, informazioni dettagliate sui possibili utilizzi dei domini possono essere rintracciate nella documentazione ufficiale Microsoft.

I sistemi operativi Microsoft da Windows 2000 in poi mantengono il supporto per i domini ma possono utilizzare una diversa infrastruttura, basata su LDAP, per la gestione delle informazioni e Kerberos 2.6.9 per l'autenticazione .

2.2.4 Mutua autenticazione

Nell'autenticazione sono coinvolti almeno due soggetti, uno dei quali cerca di provare la sua identità all'altro, l'identità dell'altro è data per scontata, mentre nella mutua autenticazione tutti i soggetti coinvolti devono provare la loro identità.

Tipicamente l'autenticazione è necessaria per accedere ad un servizio, la non certezza dell'identità del servizio può esporre a rischi, è possibile ad esempio per un attaccante creare un servizio fittizio e sfruttarlo per conoscere informazioni riservate dell'utente, può inoltre rubare le credenziali ed usarle per accedere al servizio vero, questo tipo di attacchi è conosciuto con il nome di *Man-In-The-Middle (MITM)*.

La mutua autenticazione è una esigenza particolarmente sentita quando si trattano dati sensibili come informazioni mediche, pagamenti con carte di credito o servizi come l'home banking. Per ovviare a questi problemi sono stati realizzati protocolli come il S*ecure HTTP (HTTPS)* che sfruttando la *Public Key Infrastructure (PKI)* realizza la mutua autenticazione rendendo sicure le transazioni via web.

2.2.5 Third-party authentication

La Third-party authentication può essere illustrata efficacemente descrivendo il funzionamento del meccanismo di autenticazione ufficiale degli esseri umani, ovvero il documento. L'identificazione tramite documento è caratterizzata da:

- Chi presenta il documento che deve corrispondere alle caratteristiche elencate nel documento: fotografia, altezza ecc.
- La fiducia da parte di chi accetta il documento nell'autorità che lo ha rilasciato.

In questo tipo di autenticazione entra in gioco un terzo attore, l'autorità, che viene riconosciuta dalle parti come ente in grado di identificare e certificare l'identità di una persona, l'identificazione vera e propria è avvenuta prima dell'utilizzo del documento, ad esempio ricorrendo a testimoni. Chi richiede il documento deve essere certo che l'ente certificatore è chi dice di essere, nella maggioranza dei casi è il luogo dove si richiede il documento che garantisce l'identità dell'ente, chi accetta il documento ne verifica l'autenticità controllando le caratteristiche del documento e magari, nel caso di forze dell'ordine, può contattare l'autorità per avere ulteriori prove sulla sua validità.

I documenti rilasciati dagli enti certificatori possono essere validi ovunque, come il passaporto, od avere un ambito di validità più ristretto, ad esempio il libretto universitario è valido solo all'interno dell'università.

Anche nell'informatica sono stati sviluppati sistemi di autenticazione con uno schema analogo a quello dei documenti, i più utilizzati sono il *Kerberos* (2.6.9) e la *Public Key Infrastructure (PKI)* (2.6.5), in particolare nella PKI si ritrovano tutti i concetti qui esposti.

2.2.6 Autenticazione tra sistemi

In una rete vi sono servizi che vengono utilizzati non soltanto dagli esseri umani ma anche da altri computer. Esempi di questi servizi sono i database che vengono utilizzati da applicazioni residenti su altri computer, i sistemi di backup e i file system distribuiti. Va anche osservato che si può trattare di accessi delicati e che meritano una protezione adeguata: un database può contenere dati sensibili ed un accesso non autorizzato od un mancato accesso con conseguente blocco di una applicazione vanno accuratamente evitati.

Un sistema di autenticazione deve quindi fornire all'amministratore gli strumenti necessari anche per la gestione di computer come utenti.

2.2.7 Classificazione dei sistemi di autenticazione

I sistemi di autenticazione possono essere classificati in base ad alcune caratteristiche salienti, una prima distinzione può essere tra sistemi locali ed esterni.

I sistemi locali sono quelli che non fanno uso di risorse esterne, rientrano in questa categoria i sistemi di autenticazione standard nel mondo Unix (passwd e shadow), ma anche quelli utilizzati da molte applicazioni con informazioni contenute localmente su file o database.

Nei sistemi esterni, l'autenticazione viene delegata ad un terzo attore, che stabilisce se le credenziali fornite sono corrette, rientrano in questa categoria: NIS, LDAP, Kerberos, RADIUS ed altri.

Risulta evidente che per realizzare un sistema di autenticazione centralizzato non è utilizzabili un sistema locale.

Un'altra importante classificazione si ottiene considerando quali informazioni possono essere associate all'account oltre a quelle strettamente necessarie all'autenticazione, in sistemi come il NIS nessuna, in altri come LDAP non vi sono limiti. La possibilità di associare informazioni è di vitale importanza per lo sviluppo di un sistema di autorizzazione.

Data la natura stessa dell'autenticazione un occhio di riguardo deve essere dato alla sicurezza del sistema, e quindi alla possibilità di comunicazioni crittografate, password sofisticate e tutto quanto può rendere più difficili le intrusioni illegali.

2.2.8 Procedure di autenticazione

I sistemi di autenticazione, possono avere strutture diverse ed implementare procedure diverse per autenticare un utente, alcune procedure sono di carattere generale ed utilizzate da più di un sistema, vengono qui illustrati gli schemi più utilizzati.

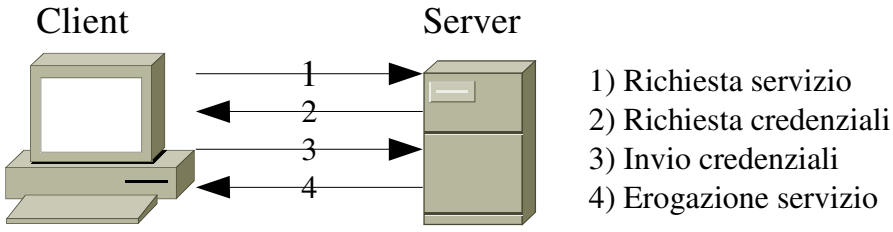

Figura 3Autenticazione locale

- L'autenticazione locale è il meccanismo con lo schema più semplice, dalla Figura 3: il client richiede l'erogazione di un servizio al server, il server richiede le credenziali al client quindi le confronta con una copia locale e stabilisce l'identità dell'utente, può quindi

stabilire se ha diritto ad usufruire del servizio richiesto. In questo schema il server ha tre incombenze che vengono eseguite sequenzialmente: autenticare, autorizzare ed erogare il servizio, deve quindi avere informazione sugli utenti e sui loro permessi.

I vantaggi di questa soluzione sono la semplicità e l'essere disponibile praticamente su ogni sistema, che però la implementa liberamente costringendo i sistemisti ad adattarsi alle possibilità fornite.

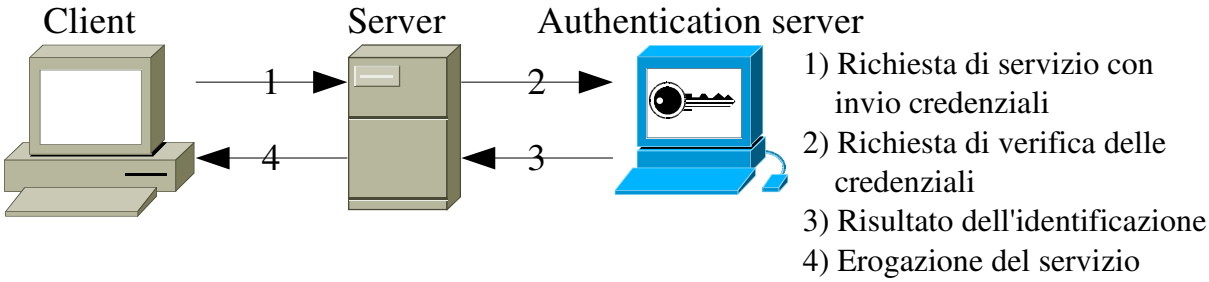

Figura 4Autenticazione con server di autenticazione

- Uno schema implementato da più sistemi di autenticazione è rappresentato in Figura 4 e prevede l'utilizzo di un server di autenticazione che ha il compito di verificare le credenziali dell'utente. Il dialogo tra client e server è identico al precedente ma le credenziali non vengono verificate localmente ma inviate al server di autenticazione che successivamente informa il server sul risultato della verifica.

  Identificato il client, il server può decidere se autorizzare l'erogazione del servizio, in questo modo al server restano due compiti: autorizzare ed erogare il servizio.

  Si osserva che l'utilizzo di un server di autenticazione non è di nessuna rilevanza per il client, e questo facilita l'adozione di un sistema che implementa tale schema perché non richiede modifiche ai client ma solo ai server che sono meno numerosi, inoltre se i servizi utilizzano gli utenti del sistema per l'autorizzazione, solo quest'ultimo deve essere in grado di accedere al server di autenticazione e quindi anche i servizi non necessitano di aggiornamento. La facilità con la quale si integra nella struttura esistente è anche il limite maggiore di questa soluzione. Infatti la transizione tra server e server di autenticazione può essere resa sicura con l'adozione di tecniche opportune mentre la transizione tra client e server resta immutata con tutti gli eventuali limiti. Tra i sistemi che implementano questa soluzione vi sono: LDAP, RADIUS e TACACS.

  Va osservato che con la stessa struttura è possibile realizzare esternamente anche l'autorizzazione, se il server di autenticazione, e a questo punto anche di autorizzazione,

fornisce una risposta positiva alla richiesta di identificazione in base alle credenziali fornite ma anche al servizio richiesto.

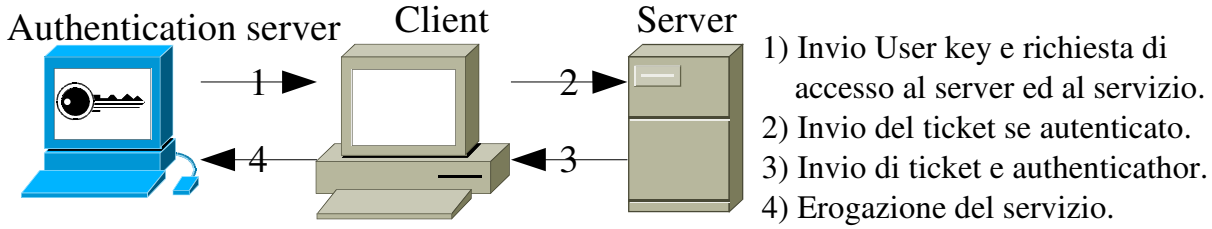

Figura 5Schema di autenticazione del Kerberos

- Tra le soluzioni a maggiore sicurezza merita senza dubbio menzione lo schema utilizzato dal Kerberos e rappresentato in Figura 5. Tale soluzione utilizza un Authentication Server (AS) che viene considerato dalle altre parti coinvolte come l'autorità che garantisce l'identità.

  Il client richiede, inviando le credenziali all'AS, di accedere al server, se autenticato ne riceve in cambio un ticket che è l'equivalente di un documento, che può poi essere utilizzato dal client per accedere al server. Differentemente dagli schemi precedentemente visti non vi è nessuna interazione tra server ed AS e la procedura è tutta a carico del client. Uno schema di questo tipo richiede la modifica dei client e dei server ed è quindi di difficile inserimento in infrastrutture preesistenti e limita all'utilizzo di software compatibile.

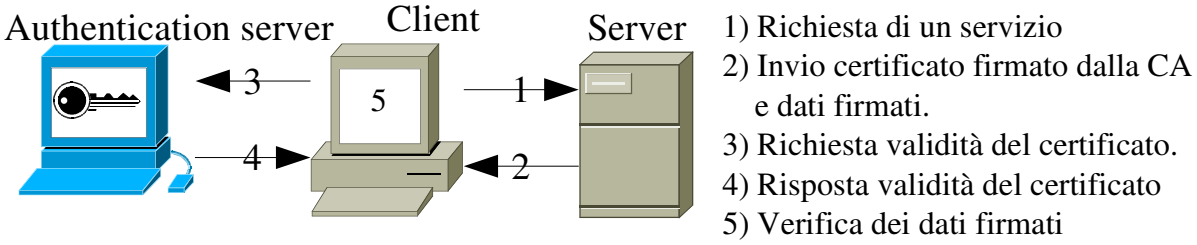

Figura 6 Autenticazione tramite certificati

- Un altra soluzione che utilizza un ente esterno è la PKI: un utente possiede un certificato rilasciato dalla autorità e può utilizzarlo per essere identificato dai server che riconoscono l'autorità. Anche questa soluzione necessità di software appositamente realizzato, lo schema utilizzato è in Figura 6.

2.2.9 Single sign-on

Alcuni sistemi di autenticazione offrono la possibilità di utilizzare il single sign-on, che consiste nella possibilità di autenticarsi una volta e di poter accedere a tutti i servizi senza

dover immettere nuovamente le credenziali per un lasso di tempo ragionevole e corrispondente tipicamente ad una giornata lavorativa. La comodità per l'utente è evidente, la sicurezza può trarre vantaggio dall'utilizzo di password più complesse che, essendo utilizzate solo una volta, possono essere accettate più facilmente dagli utenti, lo svantaggio è che una postazione lasciata incustodita dopo l'autenticazione può essere utilizzata da chiunque.

2.2.10 One time password

La tecnica del one time password sfrutta password temporanee per l'autenticazione. In tal modo, anche se intercettate, possono essere utilizzate solo una volta; al di là del nome è un trucco utilizzato anche da sistemi di autenticazione che non utilizzano le password per l'autenticazione.

Tra le implementazioni del one time password vi sono: S/Key, OTP e SecureID, l'ultimo di questi è un sistema commerciale della Security Dynamics.

A chiarificazione del funzionamento viene illustrata la procedura utilizzata da S/Key:

La password temporanea viene calcolata dalla password reale inserita dall'utente mediante una funzione di *hash*[2], l'algoritmo utilizzato nella funzione di *hash* deve essere tale che la password temporanea sia facilmente calcolabile ed il procedimento inverso estremamente costoso, S/Key utilizza l'algoritmo MD4 Message Digest di Ronald Rivest, funzione che ha come parametri la password, un seme ed un indice utilizzato per ottenere password sempre diverse.

I passaggi per l'autenticazione di un utente sono:

1. L'utente richiede l'autenticazione.
2. Il sistema di autenticazione risponde fornendo seme e indice e calcola la password temporanea.
3. L'utente elabora la password temporanea e la invia al sistema di autenticazione.
4. Il sistema di autenticazione confronta la password ricevuta con quella precedentemente calcolata e permette o meno l'accesso.

Riferimenti: [RFC1760], [RFC2444], [RFC2289].

2 Le funzioni di hash per la crittografia trasformano l'input costituito da una stringa in un'altra stringa di lunghezza fissa. Tra le caratteristiche che una buona funzione di hash deve possedere vi è la non reversibilità e il generare output molto diversi anche per piccole variazioni dell'input.

2.3 Autorizzazione

L'autorizzazione a differenza dell'autenticazione è più intimamente legata all'erogazione del servizio, ed è quindi più difficile da generalizzare in quanto ogni applicativo o sistema operativo utilizza un proprio sistema di permessi. Forse per tale motivazione non esistono sistemi di autorizzazione centralizzati se non per particolari funzioni come i file system condivisi.

In un sistema Unix, ma con qualche piccola differenza anche i sistemi Windows, gli utenti sono caratterizzati da userid e da gruppi primari e secondari, queste caratteristiche vengono utilizzate dai software e dai sistemi operativi per la gestione dei permessi. Ad esempio soltanto gli utenti che appartengono al gruppo *disk* hanno accesso diretto ai dischi del sistema, oppure solo gli utenti del gruppo *ftp* possono utilizzare l'omonimo servizio. La configurazione del server ftp può poi utilizzare gli userid per una configurazione di maggiore dettaglio. Ad esempio vietare l'utilizzo all'utente *root*.

Lo sviluppo del software negli ultimi anni tende sempre più all'utilizzo di schemi e strutture standard per facilitare l'interoperabilità che con l'impetuoso sviluppo delle comunicazioni è diventata una necessità più che una possibilità, con l'ulteriore vantaggio non trascurabile di ridurre i costi ed i tempi di sviluppo, Java che è uno dei campioni indiscussi dell'interoperabilità fornisce con la distribuzione standard *Java Authentication and Authorization Service (JAAS)*. Con tale libreria è possibile definire delle risorse, che possono essere firmate in modo da garantire a chi le utilizza la certezza della risorsa, e le politiche di accesso alle stesse.

2.3.1 Tracciamento degli account di amministrazione

L'autorizzazione degli utenti che amministrano i sistemi, per la delicatezza del ruolo e per alcune peculiarità non può essere gestita come una risorsa qualunque.

L'amministrazione di un sistema Unix non ha particolare cura della sicurezza e dell'integrità del sistema, esiste un solo account con diritti di amministrazione e non vi sono limiti alle sue possibilità. Inoltre in sistemi con configurazioni standard i servizi possono essere gestiti solo con l'account di amministrazione.

Nella pratica non è possibile avere un solo amministratore, in tale situazione l'assenza dell'amministratore comporterebbe l'impossibilità di gestire i sistemi di sua competenza,

inoltre non è detto che un solo amministratore possa avere competenze per la gestione di tutti i servizi, ed inoltre è pratica comune sgravare l'amministratore dai compiti ripetitivi per affidarli a personale meno qualificato ma che necessità per svolgere l'attività dei diritti di amministrazione.

La situazione normale è di avere più amministratori che accedono al sistema con lo stesso account, in tal modo ogni amministratore può modificare qualsiasi aspetto del sistema anche se non di sua pertinenza senza che ne resti traccia. Le conseguenze sono configurazioni in conflitto, scarsa affidabilità, mancanza di responsabilità: in sostanza un sistema gestito male.

L'autorizzazione degli account di amministrazione dovrebbe permettere di conoscere chi effettivamente sta amministrando il sistema e quali modifiche ha apportato.

2.3.2 File system condivisi

Una risorsa che richiede una gestione particolare sono i file system; in un ambiente di lavoro collaborativo si ha l'esigenza di accedere allo stesso file da parte di più utenti contemporaneamente e talvolta anche all'insaputa degli altri, se l'accesso è in lettura non vi sono particolari problemi, ma se sono necessarie modifiche, può succedere che il lavoro dell'uno venga distrutto dall'altro. Un sistema per evitare le cancellazioni involontarie è l'utilizzo delle revisioni, con cui ad ogni nuovo salvataggio la vecchia copia non viene distrutta ma mantenuta in maniera trasparente; esistono sistemi di condivisione dei file che gestiscono automaticamente le revisioni, tra questi ricordiamo *Web-based Distributed Authoring and Versioning (WebDAV)*, *Revision Control System (RCS)* e *Concurrent Versions System (CVS)*.

WebDAV è una estensione al protocollo HTTP di recente realizzazione che permette l'upload dei file in modo sicuro, il locking ed anche le revisioni; è stato progettato inizialmente per l'upload di pagine web, ma può essere utilizzato efficacemente anche per la gestione documentale. Il reperimento di software che supportano WebDAV non è particolarmente difficile, lato server è possibile utilizzare Apache e lato client Internet Explorer con il vantaggio di ridurre al minimo il tempo di apprendimento degli utenti. Riferimenti: [RFC2518], [RFC3253].

RCS (http://www.gnu.org/software/rcs/) e CVS (http://www.gnu.org/software/cvs/) sono due dei più utilizzati tra i molti software per la gestione di set di documenti che vengono acceduti

da più utenti, gestiscono le versioni ed i rami di sviluppo e l'utilizzo tipico è la gestione dei sorgenti di progetti software con sviluppatori sparsi sul territorio. Altri software simili sono: PRCS, Aegis, SCCS.

2.4 Amministrazione della sicurezza

Il raggiungimento di un livello di sicurezza soddisfacente in un sistema informatico complesso non è facile da ottenere e coinvolge il sistema nel suo insieme, le persone che lo utilizzano; il software e l'hardware non vanno considerati come elementi isolati, ma come oggetti che interagiscono tra di loro, i singoli oggetti e le interazioni che intercorrono vanno tutte parimenti considerate e valutate in termini di pericolosità. La sicurezza è frutto della valutazione costo beneficio, in quanto ogni miglioramento della sicurezza non si ottiene gratuitamente e va bilanciata correttamente,.E' inutile investire in sistemi di autenticazione sofisticati e inviolabili se gli utenti lasciano le password su foglietti attaccati ai monitor.

Il procedimento corretto per gestire la sicurezza prevede una *analisi e valutazione del rischio.* Senza scendere troppo nel dettaglio, vanno valutati nell'ordine:

1. Cosa può succedere?
2. Quali danni può provocare?
3. Con quale frequenza?
4. Quanto sono corrette le stime precedenti?

Valutati i rischi, vanno esaminate le contromisure:

1. Quali contromisure sono possibili?
2. A quale costo?
3. Il costo vale i benefici?

Le contromisure che rientrano nel budget, devono essere attivate e verificate.

La gestione della sicurezza non è una attività statica ma le tre attività: analisi del rischio, analisi delle contromisure e attivazione e verifica delle stesse devono ripetersi periodicamente.

Nei prossimi paragrafi verranno illustrati alcuni punti deboli tipici di un sistema di autenticazione centralizzato e delle tecniche per ridurre i rischi dei punti deboli. Tralasceremo tutte quelle pratiche per la messa in sicurezza di un sistema informatico che sono di normale attuazione tra i sistemisti.

2.4.1 Utilizzatori consapevoli

Un sistema di autenticazione è per sua natura utilizzato da molti utenti che non possiedono necessariamente importanti conoscenze informatiche, per cui un utilizzo scorretto del sistema può ridurre a zero la sicurezza anche di quelli meglio gestiti e progettati.

L'utenza deve essere istruita su questi punti:

- Procedure di utilizzo del sistema di autenticazione, ad esempio: password complesse e cambiate di frequente, disconnettersi a lavoro finito, non lasciare postazioni autenticate incustodite, non utilizzare e non dare le proprie credenziali ad altri. Una lettura interessante sulla gestione delle password è [Password Management] di Matt Bishop.
- I rischi di accessi non autorizzati e le eventuali conseguenze lavorative e legali.
- Registrazioni delle attività svolte.

2.4.2 Comunicazioni sicure

Le comunicazioni tra le varie componenti del sistema di autenticazione vanno adeguatamente protette, alcuni sistemi garantiscono nativamente una elevata sicurezza non utilizzando per l'identificazione dati riutilizzabili da un eventuale intercettatore, in altri la protezione dei dati deve essere aggiunta con tecniche di crittografia.

Tipicamente le credenziali () transitano tra client e server e tra server e server di autenticazione ma, mentre la comunicazione tra i server dipende esclusivamente dal sistema di autenticazione utilizzato ed è generalmente ben protetta dall'adozione di tecnologie come quelle illustrate nei prossimi paragrafi, il punto debole è la comunicazione tra client e server. Alcuni dei protocolli più utilizzati come il *pop* e l'*ftp* prevedono la trasmissione delle password in chiaro con i rischi che è facile immaginare, se si considera che è comune l'utilizzo delle stesse credenziali per più servizi, il rischio dell'utilizzo di questi protocolli è evidente. Le soluzioni possibili sono due: obbligare gli utenti ad avere credenziali distinte per i servizi meno sicuri, soluzione praticabile se l'utilizzo non autorizzato di questi protocolli non comporta rischi elevati, mentre una soluzione più complessa è la sostituzione dei protocolli standard con altri equivalenti ma a maggiore sicurezza. La sostituzione dei protocolli insicuri ha comunque dei costi che vanno valutati.

2.4.2.1 Secure Socket Layer (SSL) e Transport Layer Security (TLS)

SSL e TLS sono stati sviluppati per impedire l'intercettazione, l'alterazione e la falsificazione

di messaggi nella comunicazione tra client e server. SSL è stato creato nel 1996 da Netscape Communications Corporation [draft-freier-ssl-version3-02], mentre TLS [RFC2246] è una evoluzione di SSL del 1999 ed è un protocollo standard dell'IETF.

Il protocollo TLS è realizzato in due strati, il *TLS Record Protocol* ed il *TLS Handshake Protocol*, il primo si posiziona al di sopra di un protocollo di trasporto affidabile quale può essere il TCP, che nel modello OSI corrisponde al *session layer*, ed ha il compito di garantire la sicurezza della connessione tramite l'utilizzo della crittografia simmetrica. Il TLS Record Protocol può essere utilizzato per incapsulare protocolli di più alto livello e tra questi il TLS Handshake Protocol; questo protocollo permette la mutua autenticazione e la negoziazione di algoritmi e chiavi per crittografare le successive comunicazioni.

2.4.2.2 Simple Authentication Security Layer (SASL)

SASL è stato sviluppato dall'Internet Working Group dell'IETF per risolvere i problemi di autenticazione nello sviluppo di nuovi protocolli di rete e per risolvere le carenze dei protocolli esistenti. Si tratta di un framework che si propone di interfacciare i protocolli di rete con i meccanismi di autenticazione, in tal modo si sgravano i progettisti di protocolli dall'inventarsi dei metodi di autenticazione ed i programmatori dall'implementare nelle loro applicazioni più metodi di autenticazione. I protocolli già esistenti possono essere aggiornati per supportare SASL, al costo di una loro riscrittura e del conseguente aggiornamento dei client e dei server. Attualmente i protocolli che supportano SASL sono: BEEP, IMAP, LDAP, POP e SMTP e tutti i client ed i server ragionevolmente moderni.

SASL è pensato per essere aggiornabile e capace di sfruttare nuovi meccanismi di autenticazione, attualmente è possibile utilizzare: anonymous, plain, cram-md5, digest-md5, otp, S/Key, Kerberos, GSS Kerberos, external ed altri. External permette l'utilizzo di meccanismi non ancora supportati da SASL. Un'opzione fornita da SASL e particolarmente utile per la sicurezza è la possibilità di negoziare un security layer per le successive interazioni tra client e server.

SASL è definito in [RFC2222]. Le librerie più utilizzate per implementare il protocollo sono le Cyrus SASL della Carnegie Mellon University (http://asg.web.cmu.edu/sasl).

2.5 Robustezza

Il sistema di autenticazione è un punto critico dell'intera struttura informatica, l'intero sistema

deve essere affidabile e garantire un elevato livello di servizio, perché un mancato funzionamento può portare all'inaccessibilità di tutti i servizi.

Alti livelli di servizio sono ottenibili soltanto con una adeguata progettazione che prenda in considerazione l'hardware, il software e la gestione del sistema. L'hardware deve essere affidabile e con prestazioni adeguate ai massimi carichi di lavoro che si possono avere nelle ore di punta, che tipicamente sono gli orari di ingresso del personale. Il software deve essere scelto in modo che sia possibile utilizzare un sistema ad alta affidabilità; alcuni sistemi di autenticazione prevedono nativamente la possibilità di replicazione e la coesistenza di più server, mentre per quelli che non la prevedono è necessario implementare delle soluzioni in cluster. I server devono essere anche distribuiti sul territorio, con tale accorgimento si limita il traffico sulle dorsali e si evita che inconvenienti sui collegamenti isolino dal sistema di autenticazione delle sedi remote.

2.6 Sistemi di autenticazione

Viene proposta una panoramica dei sistemi di autenticazione disponibili, cercando di evidenziarne le caratteristiche salienti e confrontando le opzioni disponibili. Alcuni sistemi che vengono proposti non sono realmente utilizzabili ma presentano caratteristiche interessanti ed innovative.

I sistemi di autenticazione esistenti hanno caratteristiche e ambiti di utilizzo diversi. E' quindi opportuno vagliarne le caratteristiche, con particolare riguardo ai temi trattati nei precedenti paragrafi.

2.6.1 Radius (Remote Authentication Dial-In User Server/Service)

Il Radius è un sistema nato per l'autenticazione, l'autorizzazione e la configurazione del collegamento di utenti che utilizzano un Network Access Server[3]. Il RADIUS è utilizzato come Authentication Server condiviso e questa peculiarità ne ha reso possibile un suo utilizzo anche in ambiti diversi da quello originale.

3 Un Network Access Server è un dispositivo utilizzato per permettere ad utenti che si collegano in dial-up di accedere ad una rete.

I dati di configurazione utilizzati dal RADIUS sono liberamente modificabili, rendendolo utilizzabile anche per nuovi tipi di collegamento come nel caso delle Virtual Private Network (VPN) e delle reti wireless.

Esistono implementazioni del RADIUS commerciali e non per i più diffusi sistemi operativi, alcune di questi server utilizzano LDAP come base dati, i sistemi Unix che utilizzano il PAM hanno la possibilità di autenticare gli utenti con il RADIUS [RFC2865]

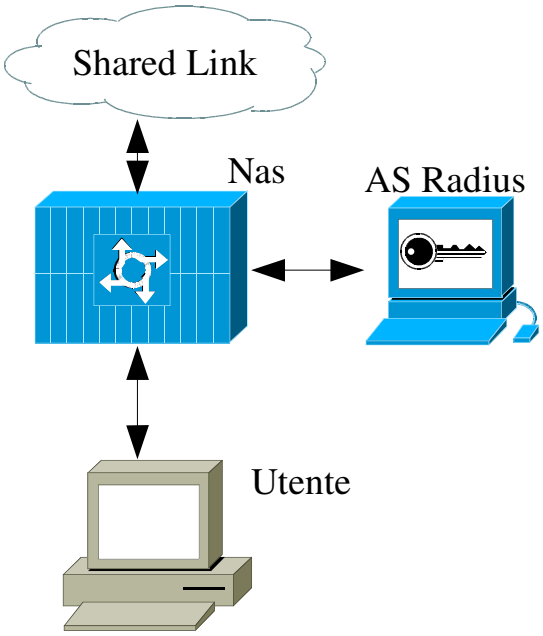

Figura 7 Schema di accesso con server RADIUS

2.6.2 TACACS/TACACS+ (Terminal Access Controller Access Control System)

TACACS ed il nuovo TACACS+ sono protocolli di autenticazione della Cisco sviluppati per l'utilizzo con Network Access Server.

Come per il RADIUS, il TACACS è stato utilizzato in ambiti diversi da quello iniziale come l'autenticazione degli utenti per l'accesso a servizi; i sistemi UNIX possono accedere al TACACS utilizzando i moduli PAM.

TACACS non è un protocollo standard ma proprietario anche se il server TACACS della CISCO è rilasciato con licenza open source, forse proprio questa sua chiusura ne ha limitato la diffusione agli apparati CISCO. Esistono delle versioni non ufficiali che possono utilizzare LDAP come base dati, ma nel complesso non è facilmente integrabile in ambienti eterogenei.

Tra i linguaggi di programmazione che supportano il TACACS con opportune librerie, l'unico di larga diffusione è il Perl.

Sono state evidenziate gravi lacune nella sicurezza di questo protocollo che ne limitano la possibilità di utilizzo, al riguardo [OW-001-tac_plus].

2.6.3 NIS/NIS+ server (Sun Yellow Pages)

Network Information Server (NIS) è un name service ideato dalla Sun per la distribuzione di informazioni sulla struttura della rete, le informazioni gestibili sono nomi ed indirizzi delle

macchine, utenti, reti e servizi di rete e l'organizzazione prevede una struttura simile a quella dei DNS con macchine master e macchine slave che replicano le informazioni.

NIS+ non è una estensione del NIS ma una sua evoluzione che permette l'aggiornamento dinamico dei dati e che può gestire oltre a quelle del NIS informazioni quali: informazioni di sicurezza, mail, interfacce ethernet. La struttura non è piatta come nel NIS ma gerarchica, con la possibilità di definire domini gestibili separatamente. Il NIS+, a causa della complessità e delle difficoltà di configurazione non ha avuto una larga diffusione e non ha sostituito il NIS. La SUN ha ufficialmente sostituito il NIS con LDAP [Web 4].

La presenza degli utenti tra le informazioni gestite rende possibile l'autenticazione centralizzata, non è invece previsto alcun supporto per l'autorizzazione.

NIS è largamente supportato dai sistemi UNIX e Linux ma non è disponibile per le piattaforme Microsoft, la scarsa sicurezza con credenziali che transitano in chiaro è uno dei punti deboli di questo sistema ed un ulteriore difetto è l'impossibilità di interfacciare i dati con altri sistemi di autenticazione.

2.6.4 LDAP (Lightweight Directory Access Protocol)

LDAP non è un sistema di autenticazione come RADIUS o TACACS, ma un protocollo sviluppato dalla University of Michigan nel 1995 per l'accesso via IP a directory basate su standard OSI X.500 con la promessa di essere, diversamente da OSI DAP (Directory Access Protocol), un protocollo “leggero”.

I client che desiderano accedere ai dati tramite LDAP necessitano di autenticazione, le credenziali fornite per l'accesso vengono confrontate con dati residenti nella directory e viene dato accesso ai dati se il confronto ha esito positivo. Il sistema di autenticazione per l'accesso ai dati è stato sfruttato per realizzare un sistema di autenticazione centralizzato per l'accesso a servizi diversi da LDAP.

L'utilizzo di un servizio di directory offre la possibilità di utilizzare le informazioni contenute per scopi diversi dall'autenticazione, i più diffusi server Unix hanno la possibilità di autenticare e di configurare gli utenti con informazioni ottenute dal server LDAP, tra i servizi che offrono questa possibilità: Apache, ProFTPD, Samba, Courier. Anche altri sistemi di autenticazione possono utilizzare LDAP come base dati, rendendo possibile l'autenticazione su più protocolli.

LDAP è completamente standardizzato dall'IETF in una lunga serie di RFC con il vantaggio di facilitare l'interoperabilità tra server e client di produttori diversi, l'essere uno standard aperto ha anche aiutato nella realizzazione di librerie e moduli per molti linguaggi di programmazione: basti citare al riguardo il supporto per Perl, Java, Python e C++.
I meccanismi di autenticazione supportati da LDAP sono definiti in [draft-ietf-ldapbis-authmeth-11], quelli di nostro interesse sono basati su username e password eventualmente protetti da TLS o SASL con i meccanismi IPSec, MD5-DIGEST ed EXTERNAL.
LDAP ha incontrato negli anni un crescente successo ed attualmente molte software house producono server LDAP, tra queste possiamo citare: Netscape, Innosoft, MessagingDirect, Oracle, Microsoft e Novell.
La struttura dei server LDAP è piuttosto robusta e prevede la presenza di master e slave per la replicazione dei dati e la distribuzione fisica dei servizi.
Il non essere un sistema di autenticazione lascia l'amministratore completamente libero di implementare la soluzione che ritiene migliore, con lo svantaggio di una progettazione che deve essere accurata per evitare lacune e rischi [RFC2251].

2.6.5 PKI (Public Key Infrastructure)

La PKI è un sistema di autenticazione con scopi che vanno ben oltre l'accesso a servizi di rete come la firma digitale e la codifica di documenti mediante una coppia di chiavi asimmetriche, una privata ed una pubblica. La struttura della PKI prevede l'utilizzo di certificati contenenti la chiave pubblica ed informazioni relative all'utente, i certificati vengono rilasciati e garantiti dalla certification authority (CA) mediante firma digitale. Lo schema utilizzato è riconducibile alla third-party authentication (vedere 2.2.5) con una struttura a catena: la CA garantisce l'autenticità firmando mediante l'utilizzo di un certificato che è stato rilasciato da una CA di livello superiore.
La gestione della CA non è banale, il certificato della CA e le chiavi private devono essere gestite con la massima sicurezza; tutte le procedure di gestione, comprese quelle per la richiesta ed il rilascio dei certificati sono definite in [RFC2510] e [RFC2585]. L'utilizzo dei certificati è piuttosto dispendioso, sia per le spese di gestione che per i supporti utilizzati per contenere le chiavi private, che tipicamente sono smart card o dispositivi USB (se i certificati vengono utilizzati per applicazioni di firma digitale i dispositivi impiegati devono rispondere

a precise norme di legge) l'utilizzo dei quali aumenta considerevolmente la sicurezza in quanto i dati contenuti sono protetti da password, trasformando il meccanismo dal tipo *one factor* (possesso) al *two factor* (possesso e conoscenza). Se si intende utilizzare certificati pubblici questi vanno acquistati con ulteriore aggravio dei costi per l'implementazione del

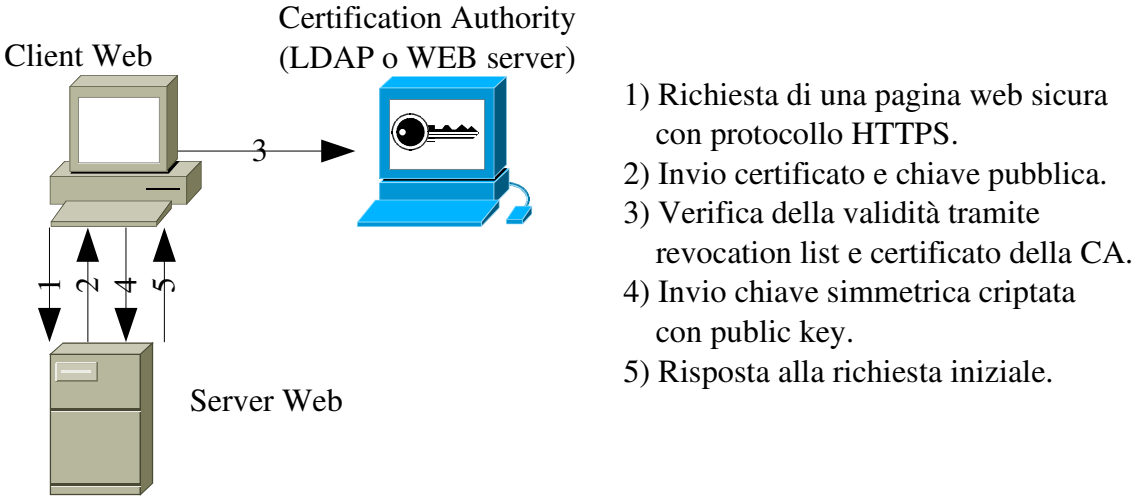

Figura 8 Autenticazione di un server web tramite certificato

sistema.

Per il funzionamento della PKI sono previsti dei metodi per la pubblicazione del certificato della CA, dei certificati rilasciati e delle liste di quelli non più validi: i sistemi previsti sono server Web o LDAP che devono essere accessibili da tutta la rete che utilizza la certification authority.

Avendo a disposizione i certificati per uso interno è possibile utilizzarli sia per l'identificazione degli utenti che per la codifica delle informazioni mediante protocolli quali TLS e VPN e con tali accorgimenti è possibile garantire livelli di sicurezza piuttosto elevati.

I certificati non possono essere utilizzati in maniera trasparente ma necessitano di client e server appositamente realizzati, tra i vantaggi va ricordata la possibilità di avere la mutua autenticazione. In Figura 8 è illustrato lo scambio di informazioni tra un browser ed un web server che si identifica tramite PKI; nel punto 3 il browser verifica la validità del certificato del server in base alla data di scadenza, alla lista dei certificati revocati ottenuta dalla CA e verificando la firma digitale con il certificato pubblico della CA. Va osservato che il certificato pubblico del server in molti protocolli compreso l'HTTPS non viene scaricato dalla CA ma fornito direttamente dal server, senza perdita di sicurezza perché è validato dalla firma digitale della CA.

2.6.6 NDS (Novell Directory Services)

NDS è un directory server, utilizzabile anche per l'autenticazione, che è possibile interfacciare con LDAP e ADS; la documentazione liberamente utilizzabile è scarsa ed è difficile avere una chiara idea delle possibilità di questo sistema. Viene pubblicizzato come un sistema capace di sopportare grandi carichi di lavoro, con la possibilità di essere amministrato da più persone con una precisa divisione delle responsabilità.

2.6.7 ADS (Active Directory Services)

Active Directory Services è il sistema di directory della Microsoft equivalente a NDS ed è fornito con i sistemi operativi a partire da Windows 2000: l'autenticazione viene realizzata tramite Kerberos e grazie a questa scelta sono disponibili il single sign-on ed il one time password; i dati per l'autenticazione e l'autorizzazione sono accessibili tramite LDAP e questa opportunità può essere sfruttata per l'utilizzo in infrastrutture comprendenti sistemi non Microsoft. L'autorizzazione si limita all'accesso alle postazioni ed ai file system condivisi.

2.6.8 NTLM (NT Lan Manager)

NT Lan Manager, è un protocollo di autenticazione sviluppato in ambito Microsoft per l'autenticazione in ambienti DCE (Distributed Computing Environment) e RPC (Remote Procedure Call), ma poi utilizzato anche per l'autenticazione di servizi di rete: HTTP, POP3, IMAP e SMTP.

I pregi principali di questo sistema sono il single sign-on ed il one time password, ne esiste una versione open source nell'ambito del progetto Samba.

NTLM è stato formalmente sostituito da ADS anche se è tuttora supportato. L'autenticazione dei protocolli di rete avviene con estensioni proprietarie che non sono riconosciute da server di altri produttori; questi difetti ne fanno un sistema da non prendere in considerazione per la realizzazione di un sistema di autenticazione eterogeneo.

Bibliografia: [Web 1]

2.6.9 Kerberos

Kerberos è il sistema di autenticazione sviluppato dal Massachusetts Institute of Technology per l'autenticazione sulla rete di campus dell'università, viene generalmente ritenuto un sistema sicuro ed adatto a grandi strutture anche distribuite. Alla implementazione del MIT ne

sono seguite altre che lo hanno reso disponibile per un elevato numero di sistemi.

Le caratteristiche più avanzate contemplano il one time password, il single sign on, la mutua autenticazione e la duplicazione della base dati. Per le infrastrutture di grandi dimensioni è possibile suddividere i domini di autenticazione in blocchi chiamati realm, i server che gestiscono un realm autenticano un insieme ristretto di utenti.

Per essere utilizzato è necessario ricorrere a software appositamente realizzato, e l'integrazione con sistemi esistenti e/o con altri sistemi di autenticazione è problematica; la gestione degli utenti è complessa e non prevede meccanismi per l'autorizzazione.

Riferimenti: [Web 2], [RFC2942], [RFC2712], [RFC1510], [RFC1411].

2.6.10 Globus Grid Security Infrastructure

Si tratta di un sistema avanzato per la gestione dell'autenticazione basato su certificati e SSL, sviluppato per garantire l'autenticazione e la comunicazione crittografata in griglie computazionali. Può essere utilizzato anche in ambiti diversi ma le uniche applicazioni esistenti in grado di utilizzarlo sono un client SSH ed uno FTP. Implementa il single sign-on ed il one time password, ed altre caratteristiche specializzate per la gestione delle griglie.

Riferimenti: http://www-unix.globus.org/toolkit/

2.6.11 Secure European System in A Multivendor Environment (SESAME)

SESAME è stato sviluppato in parte con fondi della comunità europea nell'ambito del programma RACE 2 (Research and development in Advanced Communications technologies in Europe), le caratteristiche principali sono: mutua e singola autenticazione basata su tecnologia Kerberos o PKI, comunicazioni crittografate e garantite, autorizzazione basata su Role Based Access Control (RBAC).

L'autorizzazione RBAC utilizza i ruoli e delle Access Control List (ACL) che definiscono i diritti dei ruoli, l'amministratore associa ad ogni utente uno o più ruoli e definisce opportunamente le ACL.

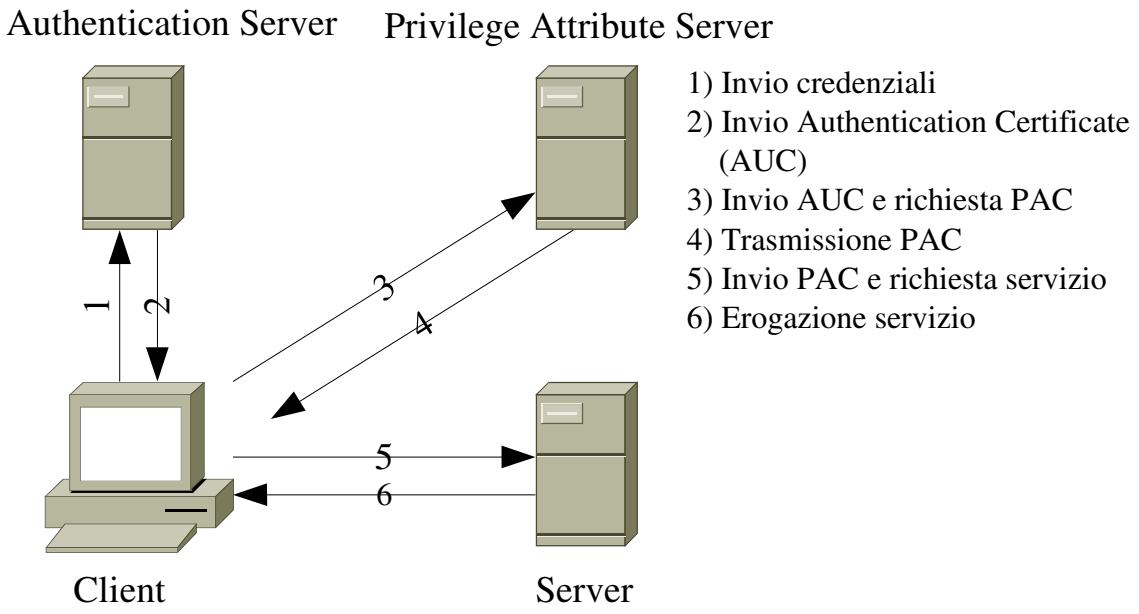

Figura 9Autenticazione ed autorizzazione in SESAME

La procedura è raffigurata in Figura 9 e prevede l'utilizzo di un Previlege Attribute Server che previa autenticazione rilascia un Previlege Attribute Certificate (PAC) contenente informazioni su tutte le risorse accessibili all'utente, il PAC è poi utilizzabile con tutti i server.

I software predisposti all'utilizzo di SESAME sono: client e server Telnet, una versione di rtools (strumenti per l'accesso remoto ed il trasferimento dei file), librerie Remote Procedure Call (RPC) per l'invocazione da remoto di procedure; applicazioni che utilizzano SESAME sono sviluppabili sfruttando l'interfaccia GSS-API[4] disponibile nella distribuzione standard.

SESAME è disponibile in formato sorgente per Unix generico o Linux RedHat 5.0, il sito ufficiale è http://www.isrc.qut.edu.au/sesame/sesame.html.

SESAME è indubbiamente una soluzione interessante al problema della autorizzazione centralizzata, si deve però osservare che una sua implementazione richiede software sviluppato ad hoc, risultando quindi indicato come sistema di autenticazione per scopi specifici.

2.6.12 Considerazioni finali

I requisiti base con cui valutare i sistemi sono: autenticazione centralizzata, autorizzazione centralizzata, interoperabilità (Unix Windows), integrabilità con altri sistemi, sicurezza, robustezza, diffusione e supporto; nella Tabella 2 sono riassunte queste caratteristiche.

4 Con l'utilizzo di GSS-API è possibile accedere al sistema di autenticazione in maniera trasparente, ovvero senza dover implementari i dettagli della procedura di autenticazione ma soltanto lo scambio di informazioni strettamente necessario. Per maggiori dettagli [RFC2743].

	Aut. centr.	Autor. centr.	Interoperabilità (*)	Integrabilità (**)	Sicurezza	Robustezza	Supporto / Diffusione
ADS	si	solo f.s.	si con accorgimenti	tramite LDAP	buona	buona	Windows recenti
NTLM	si	solo f.s.	si	no	suff.	buona	ottimo
Kerberos	si	no	si	no	buona	buona	scarso
NDS	si	forse	si	tramite LDAP	buona	buona	-
NIS	si	no	no	no	scarsa	buona	solo Unix
LDAP	si	si	si con limiti	si	buona	buona	Ottimo e crescente
PKI	si	no	si	no	ottima	buona	in crescita
TACACS	si	no	no	con LDAP	scarsa	scarsa	specialistico
RADIUS	si	no	no	con LDAP	suff.	buona	specialistico
SESAME	si	si	no	no	buona	-	scarso
GSI	si	no	no	no	ottima	buona	scarso
(*)	*Capacità di interfacciarsi con sistemi di vario tipo (es. Linux, Windows apparati di rete)*						
(**)	*Capacità di interfacciarsi con altri sistemi di autenticazione*						

Tabella 2Caratteristiche dei sistemi di autenticazione

Nessuno dei sistemi di autenticazione considerati raggiunge pienamente i requisiti richiesti, la carenza maggiore è la mancanza di un sistema di autorizzazione che sia utilizzabile per tutti i servizi di rete.

2.7 Applicazione di gestione

I sistemi di autenticazione esaminati prevedono dei tool di gestione: i prodotti commerciali forniscono solitamente delle comode interfacce grafiche per la gestione dell'intero sistema, altrimenti sono disponibili software a riga di comando che presentano difficoltà di utilizzo per il personale meno qualificato. In taluni casi sono comunque disponibili software di gestione grafici sviluppati indipendentemente dal sistema di autenticazione o librerie per linguaggi di alto livello con le quali realizzare i software necessari.

Per sistemi di autenticazione basati su server LDAP, Java si è rivelato un ottimo linguaggio

per lo sviluppo in quanto supporta nativamente e semplicemente tale protocollo. Per l'accesso a tali server è possibile ricorrere anche a client generici, ve ne sono per tutte le piattaforme, anche web-based, e di buona qualità; l'utilizzo di tali software però deve essere affidato a personale qualificato, in quanto non effettuano nessuna gestione dei dati inseriti e possono quindi portare a risultati sintatticamente corretti ma non validi. I possibili danni possono essere limitati da un'opportuna configurazione delle access list sul server LDAP.

Nel caso non siano presenti o sufficienti i software forniti con il sistema di autenticazione, devono essere realizzati degli strumenti ad hoc. Le caratteristiche e l'impegno da profondere nella realizzazione dell'applicazione di gestione sono legate all'ambiente nel quale dovrà operare, in piccole strutture possono essere sufficienti degli script che facilitino il lavoro dell'amministratore, tale soluzione non è attuabile in grandi strutture con un cospicuo numero di utenti e di sistemi da gestire. La soluzione migliore ma più dispendiosa è la realizzazione di una applicazione con linguaggi di alto livello, ma si deve osservare al riguardo che non tutti i sistemi di autenticazione forniscono strumenti adeguati per un facile sviluppo di applicazioni ad essi rivolte.

- Inserimento, rimozione e aggiornamento degli utenti.
- Inserimento, rimozione e aggiornamento delle risorse.
- Inserimento, rimozione e aggiornamento dei permessi degli utenti.
- Informazioni sugli utenti che possono usufruire di una risorsa.
- Informazioni sulle risorse accessibili da un utente.

Eventualmente potrebbe prevedere l'utilizzo di profili per facilitare la configurazione in ambienti che prevedono un elevato numero di utenti con privilegi simili.

3 Analisi e integrazione dei sistemi

Nel precedente capitolo sono state esposte un cospicuo insieme di tecnologie, ma non sono state esaminate le possibilità di reale e utile integrazione al fine di ottenere un sistema di autenticazione e autorizzazione centralizzato che risponda alle caratteristiche descritte nel primo capitolo.

Gli obiettivi che è necessario raggiungere sono riassumibile nel seguente prospetto:

- Base di dati unica.
- Maggior numero di sistemi di autenticazione possibile.
- Facilità di amministrazione.
- Ragionevole livello di sicurezza.
- Sufficiente livello di robustezza.
- Utilizzo preferenziale di software open source.
- Compatibilità con il maggior numero di sistemi operativi.
- Compatibilità con il maggior numero di servizi.

3.1 Schema del sistema di autenticazione

Il punto centrale del sistema di autenticazione è la base di dati alla quale accedono l'applicazione di gestione ed i sistemi di autenticazione che deve quindi essere facilmente accessibile dai linguaggi di programmazione più utilizzati per la realizzazione dell'applicazione di gestione, dal maggior numero di sistemi di autenticazione e dal più ampio numero di servizi possibili.

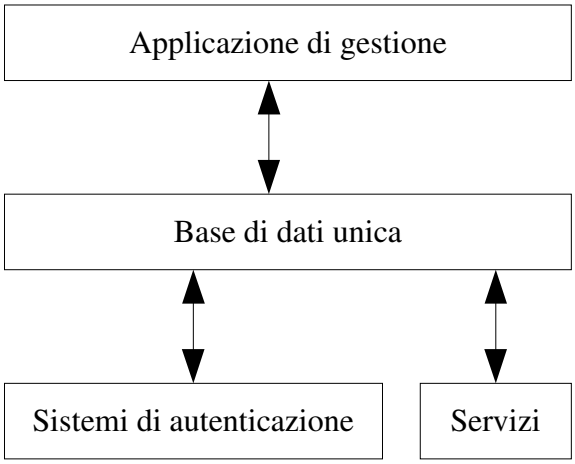

Figura 10 Struttura del sistema di autenticazione

3.2 Scelta della base di dati

Le basi dati tra cui scegliere sono due: LDAP o database relazionale; l'utilizzo di un database relazionale è in realtà impraticabile perché i servizi in grado di utilizzarlo sono limitati, una soluzione di compromesso consiste nell'utilizzare il database relazionale come base dati per LDAP, in tal modo verrebbe meno il limite dei servizi utilizzabili con il vantaggio che l'applicazione di gestione deve interagire con un database relazionale, che è una pratica standard e di facile implementazione. Lo svantaggio principale di questa soluzione, oltre all'aggiunta di un ulteriore punto critico, è il calo delle prestazioni, dovuto alla scarsa compatibilità della struttura dati ad albero tipica di LDAP con quella tabellare dei database relazionali[5] per cui, da quanto esposto, emerge che l'unica reale possibilità è l'utilizzo di server LDAP.

I sistemi di autenticazione che possono utilizzare LDAP come base dati sono il RADIUS, il TACACS, con opportune modifiche alla versione fornita dalla Cisco, il Samba, che dalla versione 2 ha il supporto LDAP, e PKI, che utilizza LDAP come sistema standard per la distribuzione dei certificati. L'integrazione di questi sistemi copre la quasi totalità delle esigenze di autenticazione, anche se i sistemi Microsoft più recenti non possono utilizzare Active Directory ma devono essere limitati all'utilizzo dei domini.

Active Directory è fondato su un server LDAP ed è quindi possibile l'utilizzo di tale server come base dati dell'intero sistema di autenticazione. Tra la documentazione online della

5 Alcuni test ufficiosi hanno evidenziato che il server LDAP di Oracle che utilizza l'omonimo DBMS come back-end ha prestazioni piuttosto scarse nonostante la qualità del database, anche gli sviluppatori di OpenLDAP che ha un supporto sperimentale per i database relazionali evidenziano il calo di performance.

Microsoft è reperibile il documento "Mixing It Up: Windows, UNIX, And Active Directory" [Web 3] dal sottotitolo "As the world becomes more and more connected, a problem has emerged. How do organizations and partners store sensitive data in heterogeneous environments, and how do they verify the identity of users requesting the information on any platform?", che descrive come integrare sistemi Unix e Windows utilizzando Active Directory e sono inoltre reperibili sul mercato dei software come *Vintela Authentication Services* che semplificano le operazioni necessarie all'integrazione.

LDAP può essere sfruttato da alcuni server senza usufruire dell'autenticazione di sistema. I software che forniscono questa possibilità sono in continuo aumento e ne viene presentata nel seguito una lista ottenuta con ricerche in Internet:

- WWW: Apache, Caudium e Lighttpd.
- SMTP : Sendmail, Postfix, QMail e Exim.
- POP3, IMAP: Courier, tpop3d, QMail e Dovecot (IMAP).
- FTP: PureFTPD, ProFTPD
- Proxy: Squid.

Tutti i linguaggi di programmazione di uso comune supportano LDAP, quindi non vi sono vincoli per la realizzazione dell'applicazione di gestione.

In Figura 11 è rappresentato un possibile schema di sistema di autenticazione basato su LDAP, con le applicazioni di gestione, i sistemi di autenticazione ed i servizi che accedono alla base dati principale.

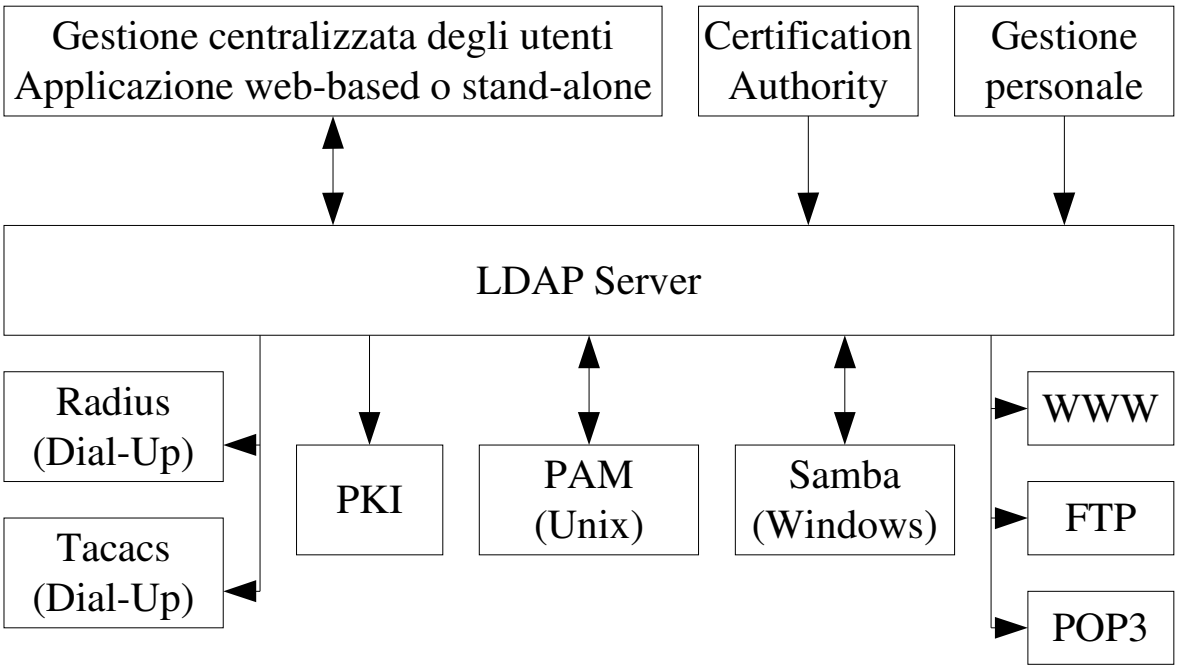

Figura 11 Sistema di autenticazione

3.2.1 LDAP (Lightweight Directory Access Protocol)

LDAP è un servizio di directory i cui standard sono definiti dall'IETF (Internet Engineering Task Force) l'attuale versione di LDAP è definita in: [RFC2251], [RFC2252], [RFC2253], [RFC2254], [RFC2255], [RFC2256], [RFC2829], [RFC2830], [RFC3377]. Non essendo il concetto di directory ampiamente diffuso, può risultare utile un esempio di directory di comune utilizzo come il Domain Name System (DNS) per esemplificare alcune delle caratteristiche tipiche dei servizi di directory:

- Operazioni di lettura ottimizzate. Le operazioni di lettura sono le più frequenti e vengono ottimizzate: generalmente un DNS server mantiene in memoria le zone gestite mentre per le operazioni di scrittura deve rileggersi tutte le informazioni della zona da disco. Al contrario dei DBMS che prevedono, salvo ulteriori configurazioni, che le operazioni di lettura e scrittura siano dello stesso ordine di grandezza, nei server LDAP la velocità dell'operazione di lettura è decisamente superiore a quella in scrittura.
- Implementazione di un modello distribuito per l'archiviazione delle informazioni. Esistono migliaia di DNS server, gestiti da altrettanti amministratori, ma capaci di interagire nella realizzazione di un servizio su scala mondiale.
- Struttura delle informazioni estendibile. Non esistono schemi rigidi ed è possibile aggiungere informazioni con diverse caratteristiche, nei server LDAP la libertà nella

struttura, e nel tipo delle informazioni è notevole.

- Sistema di ricerca evoluto. Permette la ricerca su ogni campo, LDAP permette ricerche complesse.
- I dati sono replicabili automaticamente su più server senza interruzioni. I DNS server sono coadiuvati da server secondari che vengono periodicamente aggiornati, i server LDAP hanno una struttura analoga.

Può risultare utile anche un rapido confronto con i database, le caratteristiche che li accomunano sono l'estendibilità dei dati e la velocità delle ricerche, ma i database assumono che le operazioni di lettura e di scrittura abbiano la stessa probabilità di essere eseguite mentre un servizio di directory privilegia la lettura e non implementa funzioni fondamentali in un database come il lock in scrittura e le transazioni.

Il sistema utilizzato per archiviare fisicamente i dati non è specificato nei protocolli che descrivono LDAP, il metodo utilizzato è quindi una scelta della specifica implementazione.

3.2.2 Struttura dei dati in LDAP

La struttura dei dati in LDAP è a *grafo diretto aciclico* (DAG), ma comunemente viene utilizzata la struttura ad albero. La struttura dell'albero e dei nodi non presenta particolari limiti, più in dettaglio:

- Da ogni nodo possono partire infiniti rami.
- Ogni nodo può avere dati con strutture diverse.
- La struttura dei dati del singolo nodo è liberamente definibile ed è *object oriented*.
- Uno stesso oggetto può essere riferito in due nodi diversi dell'albero con un meccanismo simile ai *soft link* dei file system.

I dati contenuti in un nodo vengono chiamati *entry*, e sono formati da uno o più oggetti, gli oggetti vengono dichiarati in opportuni file di configurazione con sintassi non standard. Tali file vengono generalmente indicati col nome di schemi e definiscono degli *objectClass* che determinano il nome, il tipo, la semantica e la sintassi dei campi.

Le entry sono formate da:

- *Distinguished Names* (DN): Una lista di coppie nome-valore che individua la posizione del nodo all'interno dell'albero. Ad esempio `dn: ou=servizi, dc=aop, dc=int`, l'ultima coppia individua la radice dell'albero, la prima distingue il figlio dai fratelli, le altre

individuano un percorso nell'albero, la prima coppia viene chiamata *Relative Distinguished Name* (RDN).

- Uno o più objectClass che definiscono il tipo di dati contenuti; gli attributi del nodo si ottengono mediante la composizione degli objectClass utilizzati.
- I dati del nodo come coppie attributo-valore, che rispettano le definizioni degli objectClass; si deve osservare che possono essere generalmente presenti più coppie attributo-valore con lo stesso attributo.

3.2.2.1 Alias objectClass

L'objectClass *Alias* permette di realizzare collegamenti tra nodi dell'albero, è l'equivalente dei soft link nei file system Unix. Ha un solo attributo *aliasedObjectName* il cui valore è il *distinguished name* del nodo da riferire.

Nell'esecuzione di ricerche è possibile specificare se gli oggetti di tipo *Alias* devono essere deferenziati, se vengono deferenziati è come se l'oggetto puntato si trovasse nel nodo che effettivamente contiene l'Alias.

Gli Alias vanno usati con cautela perché nella configurazione dei software che accedono ad LDAP non è sempre possibile specificare la deferenziazione automatica, dunque è opportuno limitare l'utilizzo degli Alias ai rami cui accedono software in grado di sfruttarli o nella porzione di albero destinata all'applicazione di gestione.

3.2.3 Informazioni necessarie al sistema di autenticazione

Le informazioni che è necessario archiviare nei nodi dipendono dal singolo sistema di autenticazione, da eventuali informazioni aggiuntive per la gestione dell'autorizzazione e da informazioni ausiliarie che si desidera associare agli utenti per la gestione.

Possono essere previste anche ulteriori informazioni non strettamente necessarie per l'autenticazione e la gestione ma utili per altri scopi, ad esempio la rubrica aziendale.

I sistemi che utilizzano LDAP come base di dati per l'autenticazione hanno tipicamente tre parametri configurabili per l'accesso e la ricerca dei dati:

- Un nodo di inizio della ricerca che viene poi effettuata su tutti i figli. Talvolta è possibile specificare la profondità con la quale effettuare la ricerca.
- Un filtro di ricerca in base al quale selezionare i nodi contenenti informazioni utili al sistema di autenticazione, i filtri possono utilizzare lo standard LDAP [RFC2254] che

permette strutture anche sofisticate o più semplicemente solo sul tipo di objectClass del nodo.

- Informazioni su come mappare gli attributi contenuti nel nodo con i dati necessari al sistema di autenticazione, ad esempio quale attributo è lo userid e quale è la password.

3.2.4 ObjectClass standard

Esistono schemi LDAP standard per le applicazioni più diffuse, alcuni di questi sono utilizzabili per la realizzazione di sistemi di autenticazione e la rappresentazione di organigrammi ed è opportuno sfruttarli.

Dal cosine.schema definito in [RFC1274]:

- *account*: definisce un account generico, che può essere specializzato utilizzando altri objectClass quali *posixAccount* e *sambaAccount*.

```
account OBJECT-CLASS
     SUBCLASS OF top
     MUST CONTAIN {
          userid }
     MAY CONTAIN {
          description,
          seeAlso,
          localityName,
          organizationName,
          organizationalUnitName,
          host }
```

Gli attributi più significativi sono: *userid* per l'identificazione univoca dell'utente, d*escription* ed *host* possono essere sfruttati per il filtraggio, in particolare *host* può essere utilizzato per definire su quali host l'account è valido e *description* per identificare i servizi utilizzabili con questo account

- *simpleSecurityObject*: permette di associare una password al nodo, ad esempio se utilizzato insieme ad *account*, rende disponibile la coppia *userid* (da *account*) e *userPassword* (da *simpleSecurityObject*).

```
simpleSecurityObject OBJECT-CLASS
     SUBCLASS OF top
     MUST CONTAIN {
     userPassword }
```

Dal nis.schema definito in [RFC2307]:

- *posixAccount*: definizione di un account Unix è un objectClass di tipo Auxiliary, questo

tipo di oggetti deve essere associato ad almeno un objectClass di tipo strutturale per poter essere utilizzato.

```
posixAccount OBJECT-CLASS
     SUBCLASS OF top
     MUST CONTAIN {
          cn,
          uid,
          uidNumber,
          gidNumber,
          homeDirectory}
     MAY CONTAIN {
          userPassword,
          loginShell,
          gecos,
          description}
```

Tipicamente viene associato ad *account* per definire un account standard Unix.

- *shadowAccount*: definisce gli attributi per realizzare un account standard Unix di tipo shadow.
- *posixGroup*: per la gestione dei gruppi Unix.

In samba.schema sono presenti gli objectClass per la definizione di account per l'accesso ai servizi samba e la gestione dei domini: *sambaSamAccount*, *sambaDomain* e *sambaGroupMapping*, è presente anche *sambaAccount* che è stato sostituito da *sambaSamAccount* per rimuovere dei conflitti con i nomi di altri oggetti.

Da radius.schema:

- *radiusprofile*: per implementare account radius.

Per la descrizione degli utenti sono disponibili *person* che contiene un numero esiguo di dati e *inetOrgPerson* che è stato realizzato per completare il precedente con gli attributi che negli anni si sono resi necessari, tra questi uno dei più interessanti è *userCertificate* per l'utilizzo di certificati digitali.

La gestione dei certificati in objectClass che non la prevedono può essere ottenuta utilizzando *strongAuthenticationUser* il cui unico attributo è per l'appunto *userCertificate*.

3.2.5 Requisiti della struttura dell'albero LDAP

L'albero LDAP è caratterizzato non solo dai dati contenuti ma anche dalla sua struttura, la scelta della struttura può essere determinante per semplificare la gestione e ridurre la ridondanza dei dati, inoltre una struttura progettata correttamente può semplificare le

configurazioni dei servizi e conseguentemente i problemi. La scelta della struttura, fatti salvi i pochi limiti imposti da LDAP, è assolutamente libera, per una adeguata progettazione deve essere presa in considerazione la struttura del sistema informativo ed i limiti che si vogliono imporre agli utenti nell'utilizzo della stessa.

3.2.5.1 Albero LDAP ed autorizzazione

I dati contenuti e la struttura dell'albero devono essere progettati in modo da permettere la realizzazione di un sistema di autorizzazione così da raggiungere uno degli obiettivi fondamentali quale?, esaminiamo come sia possibile ottenere questo risultato considerando il caso di due servizi che utilizzano LDAP per l'autenticazione.

L'autorizzazione è ottenibile sfruttando i dati contenuti nei nodi o la struttura dell'albero: se si utilizza la struttura i servizi avranno come radice della ricerca due nodi distinti, ogni nodo avrà come figli solo quelli relativi agli utenti che possono usufruire di tale servizio; se si utilizzano i dati, i servizi dovranno avere delle impostazioni di filtraggio diverse ed in grado di discriminare solo gli utenti autorizzati all'utilizzo del servizio. In Figura 12 è riportato un semplice esempio con filtraggio sull'attributo *servizio*.

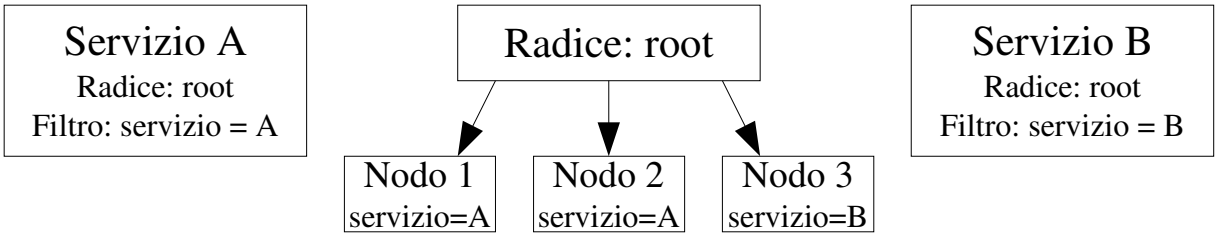

Figura 12Autorizzazione tramite filtraggio

L'autorizzazione tramite struttura presenta il vantaggio di una separazione netta tra i dati dei due servizi con conseguente semplificazione dell'applicazione di gestione, di incremento delle prestazione, perché devono essere letti un insieme ridotto di nodi, ed inoltre il non utilizzo dei filtri elimina una possibile fonte di bug dovuta alla maggiore complessità delle configurazioni.

3.2.5.2 Albero LDAP ed utenti

L'autorizzazione prevede l'associazione tra account e servizi, ma gli account sono anonimi ed è quindi opportuno integrare questa informazione con l'associazione tra utenti ed account e per transizione tra utenti e servizi. Con tale relazione è possibile conoscere chi utilizza un account e a quali servizi può accedere.

La struttura dell'albero deve essere progettata in modo da permettere tale relazione e a tale

scopo può essere utilizzato efficacemente l'objectClass Alias, una possibile soluzione è un albero con solo le informazioni degli utenti, ad esempio con un nodo per ogni utente che ha tanti figli quanti account, i figli potrebbero essere di tipo Alias che li rimandano al nodo di account relativo.

3.2.6 Tipi di alberi LDAP per l'autenticazione

In questo paragrafo vengono illustrate alcune possibili implementazioni della struttura ad albero nel tentativo di identificarne i pregi ed i difetti, alla ricerca del migliore compromesso tra semplicità di realizzazione e flessibilità.

3.2.6.1 Albero ad un livello

Si tratta della struttura ad albero più semplice, la complessità si sposta sulla struttura e sui dati dei nodi.

Ad ogni utente è associato un solo nodo contenente le informazioni necessarie per l'accesso e l'autorizzazione, i nodi sono costituiti da più objectClass, dipendentemente dai tipi di servizi accessibili, ed è inoltre presente un objectClass di tipo *person,* o analogo, per le informazioni relative all'utente.

Un ulteriore semplificazione si può ottenere utilizzando nodi tutti con la stessa struttura, in tal caso devono essere previsti tutti gli objectClass necessari ai vari sistemi di autenticazione più eventuali informazioni sulle persone associate agli utenti.

I sistemi di autenticazione accedono tutti alla stessa radice e l'autorizzazione è ottenuta mediante filtraggio, nell'esempio di Figura 13 viene utilizzato solo l'attributo *host* ma è possibile realizzare filtraggi più sofisticati utilizzando altri attributi come *description*.

Un vantaggio di questa struttura è di utilizzare l'UID direttamente nel RDN, questo comporta l'univocità dell'UID data l'impossibilità di avere due nodi con lo stesso DN, l'univocità dell'UID è un requisito di carattere generale e deve sempre essere sempre garantita, altrimenti due utenti diversi con lo stesso UID potrebbero accedere l'uno ai servizi dell'altro.

L'utilizzo di un solo account per lo stesso UID comporta che ogni sistema al quale l'utente può accedere deve essere configurato esattamente nello stesso modo, ad esempio se un utente ha accesso a due sistemi Unix, nei due sistemi deve avere la stessa home directory e lo stesso uidnumber. Questo comporta un vantaggio nella maggiore semplicità della gestione dei dati, ma una pesante riconfigurazione per l'allineamento dei sistemi esistenti ed una scarsa

flessibilità.

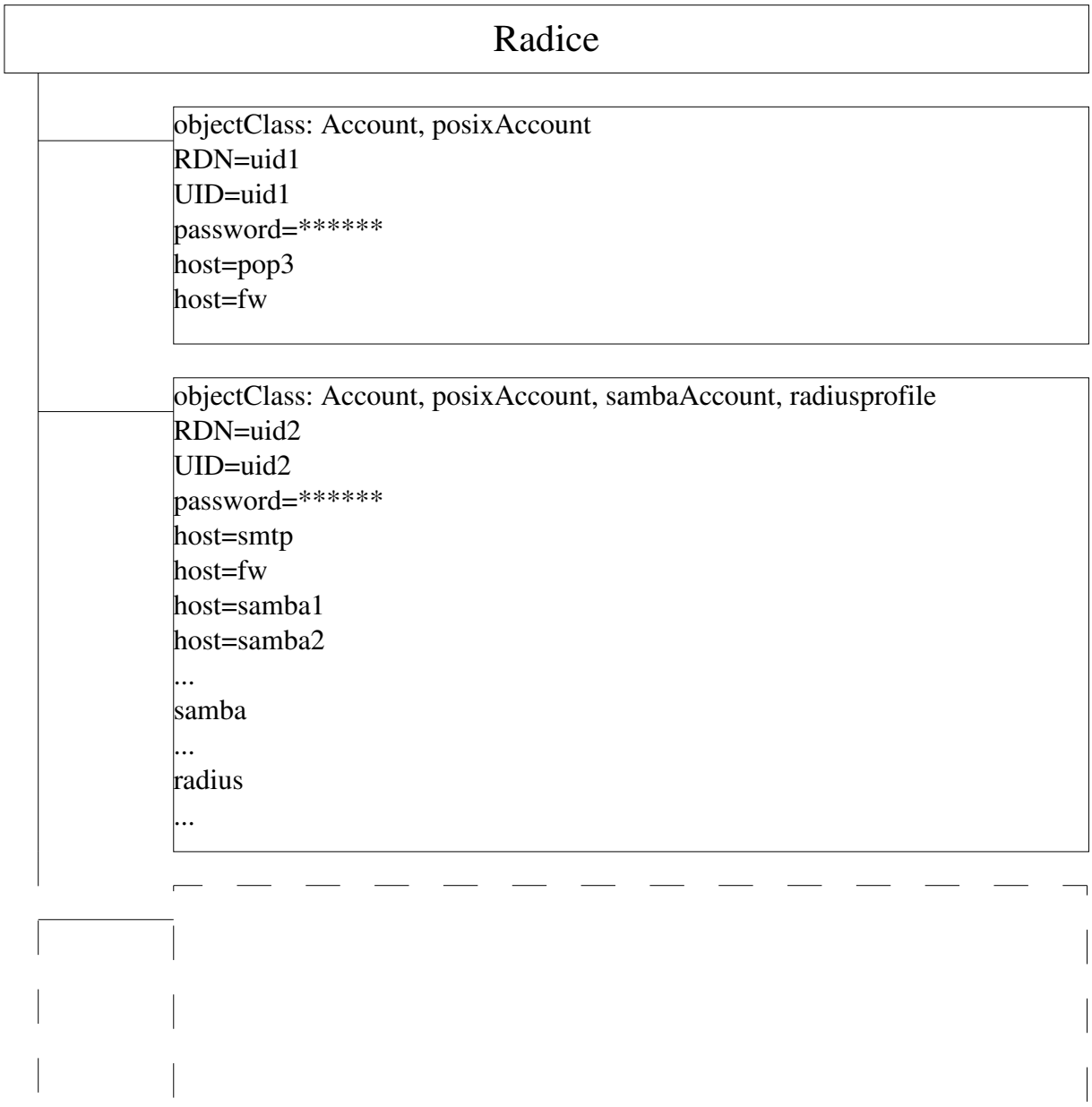

Figura 13Albero ad un livello

3.2.6.2 Albero a due livelli

I nodi dell'albero sono a due livelli, i nodi del primo livello contengono le informazioni dell'utente, quelli del secondo livello gli account, con tale struttura ogni utente può avere più nodi e quindi si ottiene una maggiore flessibilità.

Nello schema proposto in Figura 14 gli utenti sono caratterizzati dall'objectClass *person* e distinti in base al nome, mentre in una realizzazione pratica si possono utilizzare altri objectClass o modificare *person* in modo da evitare problemi tra utenti omonimi.

L'associazione tra utente ed account è ottenibile direttamente dalla struttura e non richiede

ulteriori informazioni.

La struttura degli account è completamente libera, è ad esempio possibile:

- Utilizzare lo stesso account per più servizi.
- Un account diverso per ogni servizio.
- Più account per lo stesso servizio.
- Uno stesso utente può avere più UID e password diverse per ogni servizio, l'univocità degli UID è lasciata al software di gestione che nel creare gli account deve verificare che per lo stesso servizio l'UID sia unico.

I servizi accedono tutti alla stessa radice e l'autorizzazione è ottenuta con l'utilizzo di filtri, devono quindi essere previsti allo scopo opportuni attributi nel nodo.

La flessibilità ottenibile con questo schema è considerevole ma viene pagata con la maggiore complessità della struttura e dell'applicazione di gestione.

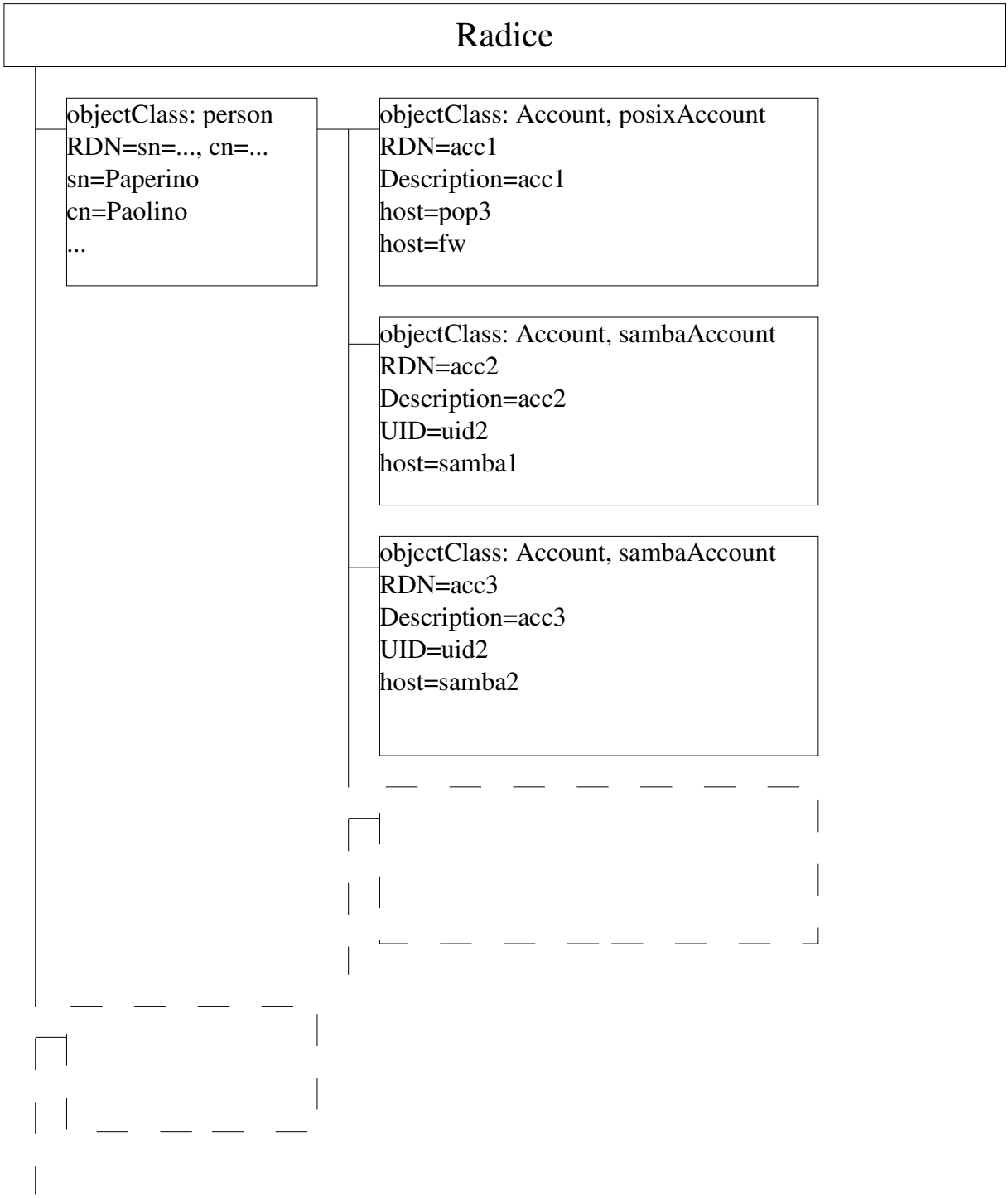

Figura 14Albero LDAP a due livelli

3.2.6.3 Albero LDAP a servizi separati

I servizi con insiemi di utenti diversi accedono a radici diverse, l'autorizzazione è legata alla struttura dell'albero e non è necessario far ricorso al filtraggio, questo comporta una semplificazione dei dati contenuti nei nodi. La flessibilità è analoga allo schema precedente, ed ogni utente può avere account con caratteristiche diverse per ogni servizio Figura 15.

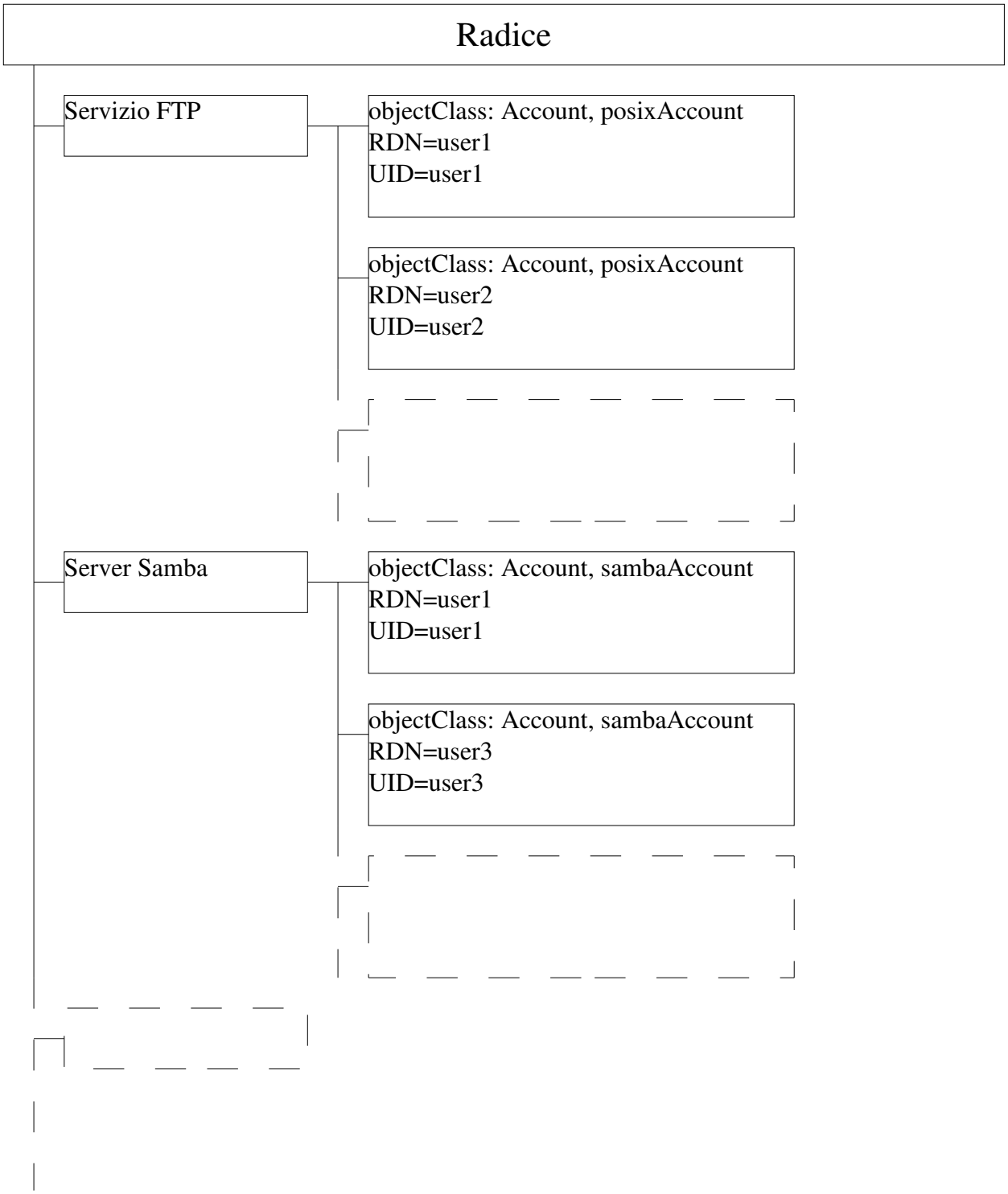

Figura 15Albero LDAP a servizi separati

L'applicazione di gestione deve evitare che utenti diversi abbiano lo stesso UID, con questa struttura se più utenti hanno lo stesso UID non si hanno problemi di accesso non autorizzato perché i servizi accedono a radici diverse, ma i due utenti con lo stesso UID non possono accedere agli stessi servizi.

L'associazione tra utenti ed account non è immediata e si rende necessario l'utilizzo di strutture ausiliarie; se si evitano utenti con UID uguali è possibile utilizzare lo schema di Figura 16 che utilizza l'objectClass inetOrgPerson definito in [RFC2798]. Questo objectClass ha un attributo UID che può essere utilizzato per ricercare gli account associati agli utenti.

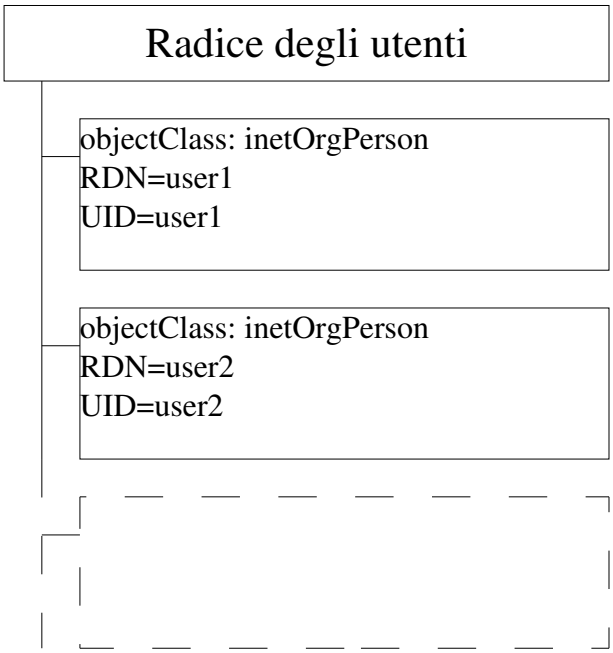

Figura 16Albero LDAP per la gestione degli utenti

3.3 Protocolli di autenticazione

II protocolli di autenticazione utilizzabili sono RADIUS, TACACS, Samba e PKI; i primi tre sono dei servizi che utilizzano LDAP come base dati, PKI prevede l'accesso diretto dei client all'albero LDAP.

- L'unico server RADIUS open source con supporto LDAP è FreeRADIUS (http://www.freeradius.org) rilasciato con licenza GNU General Public License (GPL), si tratta di un ottimo prodotto di larga diffusione che viene fornito con gli schemi per OpenLDAP con i quali si definiscono l'objectClass radiusprofile con gli attributi standard per l'autenticazione e l'accesso in dial-up.
- Il server TACACS viene fornito dalla Cisco senza supporto LDAP, ma vi sono patch non ufficiali per l'utilizzo di LDAP (http://www.gazi.edu.tr/tacacs/); diversamente dal server RADIUS, il server TACACS utilizza LDAP solo per l'autenticazione e non per le informazioni necessarie all'accesso in dial-up.

3.3.1 Samba

Il server Samba può essere utilizzato in infrastrutture che utilizzano i Workgroup, i Domini o Active Directory, con diverse opzioni:

- Workgroup: il server Samba funziona da file server e LDAP può essere utilizzato per le informazioni relative agli utenti. Questa possibilità è sfruttabile dalla versione 2.0 del Samba.
- Dominio: Samba può funzionare da membro del dominio, ed in tal caso gli utenti sono gestiti dal Primary Domain Controller (PDC) e LDAP non viene utilizzato, oppure da PDC, con questa configurazione Samba deve avere informazioni su tutti gli utenti ed i gruppi del dominio, tutte informazioni gestibili con LDAP.

  LDAP prevede l'utilizzo di server master e slave, i domini prevedono dei server di backup (Backup Domain Controller BDC) che si affiancano al PDC, l'associazione tra server master e slave e BDC e PDC non è libera per problemi sui tempi di aggiornamento; dalla documentazione del Samba risulta che deve essere assolutamente evitato che il PDC utilizzi un server slave mentre i BDC possono utilizzare sia server slave che master.
- Active Directory: Samba può essere utilizzato come client di AD e non come server (è prevista l'implementazione di tale caratteristica dalla prossima versione); una possibilità interessante è l'utilizzo di AD come server LDAP.

La distribuzione standard di Samba include gli schemi per OpenLDAP con la definizione di tutti gli objectClass necessari.

3.3.2 Public Key Infrastructure

Per il corretto funzionamento di una certification authority (CA) è necessario pubblicare il certificato della CA, la lista dei certificati revocati e inoltre può risultare utile pubblicare anche i certificati degli utenti.

L'objectClass *pkiCA* può essere utilizzato per la pubblicazione, più in dettaglio:

```
pkiCA OBJECT-CLASS
      SUBCLASS OF top
      MAY CONTAIN {
           authorityRevocationList,
           certificateRevocationList,
           cACertificate,
           crossCertificatePair
      }
```

authorityRevocationList e *certificateRevocationList,* per i certificati revocati.

cACertificate: il certificato della CA.

crossCertificatePair: per l'autenticazione tra CA.

Per i certificati degli utenti possono essere utilizzati *inetOrgPerson* e *pkiUser* (o l'equivalente *strongAuthenticationUser);* il primo descrive una persona ed ha molti attributi, il secondo ha un solo attributo, il certificato, e può essere utilizzato per aggiungere il certificato a objectClass che non lo contemplano.

I certificati sono codificati con ASN.1 DER (Distingushed Encoding Rules) ed essendo dati binari per essere pubblicati con LDAP devono essere codificati in BASE64 per ottenere un dato testuale.

3.4 Strategie per la sicurezza

3.4.1 Accessi al server LDAP

Al server LDAP devono poter accedere l'applicazione di gestione, i server che lo utilizzano per l'autenticazione, gli altri sistemi di autenticazione ed i client che necessitano dei dati della CA.

Per limitare gli accessi può essere opportuno duplicare i dati della CA su un altro server LDAP, in tal modo il server principale deve essere raggiungibile solo da altri server ed è possibile filtrare pesantemente i diritti di accesso.

LDAP permette una configurazione estremamente dettagliata degli accessi mediante Access Control List (ACL), dove una ACL è definita da tre parametri: un insieme di dati, un insieme di utenti e dalle modalità di accesso.

- Gli utenti possono essere anonimi, autenticati e utenti il cui nome è contenuto nell'albero, per la definizione è possibile utilizzare delle regular expression.
- Gli oggetti sui quali si applica la ACL sono definibili in tre modi che possono essere anche utilizzati contemporaneamente.

1. Mediante una regular expression che corrisponde al DN è anche possibile specificare se la ACL si applica al solo nodo corrispondente all'espressione o anche ai figli o a tutto il sotto albero che si diparte dal nodo.
2. Un filtro standard LDAP.
3. Una lista di attributi sui quali si applica la ACL.

Il tipo di accesso può essere in lettura, scrittura, ricerca, comparazione, autorizzazione[6], nessuno.

L'applicazione di gestione deve avere accesso in lettura e scrittura a tutto l'albero LDAP, i server devono accedere in lettura a tutti gli account di loro pertinenza, ma è anche possibile lasciargli accesso in lettura a tutto l'albero. Se si vuole permettere la modifica delle password direttamente dai server deve essere dato anche l'accesso in scrittura. Alcuni server come Samba possono modificare i dati dell'albero LDAP mediante i normali tool di gestione e quindi richiedono l'accesso in scrittura se si intende usufruire di questa possibilità ed è opportuno creare delle ACL che limitino i diritti di scrittura ai soli dati di pertinenza del server.

L'accesso in lettura agli utenti anonimi deve essere disabilitato almeno per le parti di albero contenenti le password che, anche se sono memorizzate in forma criptata, devono restare inaccessibili per evitare attacchi di forza bruta.

3.4.2 Password crittografate

Le password contenute nell'albero LDAP possono essere memorizzate in sei formati: clear, MD5 ([RFC1321]), SHA ([FIPS PUB 180-1]), SMD5, SSHA, crypt (standard Unix). Tranne il primo che è in chiaro, gli altri sfruttano diversi algoritmi di hashing della password, e, per un livello sufficiente di sicurezza, si può tranquillamente utilizzare il formato crypt.

3.4.3 Crittografia delle comunicazioni

Un possibile attacco alla sicurezza del sistema è la cattura durante il transito in rete delle password; il sistema più semplice per arginare il problema è ricorrere a comunicazioni crittografate. In Figura 17 sono evidenziate le comunicazioni che riguardano le password in una rete con una zona protetta da firewall ed una non protetta con client ed eventuali server distribuiti sul territorio e quindi non inseribili nell'area protetta.

Le comunicazioni tra server LDAP e server, e tra server LDAP master e slave prevedono nativamente il possibile utilizzo di TLS e sono quindi facili da proteggere, mentre per le comunicazioni tra client e server la situazione è più complicata perché alcuni dei più diffusi protocolli di rete, e tra questi POP3, FTP e TELNET, trasmettono in chiaro. Questo è un punto di debolezza dell'intero sistema, aggravato dal fatto che gli utenti tendono ad utilizzare

6 I dati sono utilizzabili al solo fine di autenticare un utente.

la stessa password su più sistemi; l'unica protezione possibile è evitare l'utilizzo di protocolli non protetti sostituendoli con equivalenti protetti e tra questi possiamo citare POP3S, SSH, SFTP che sostituiscono rispettivamente POP3, TELNET e FTP.

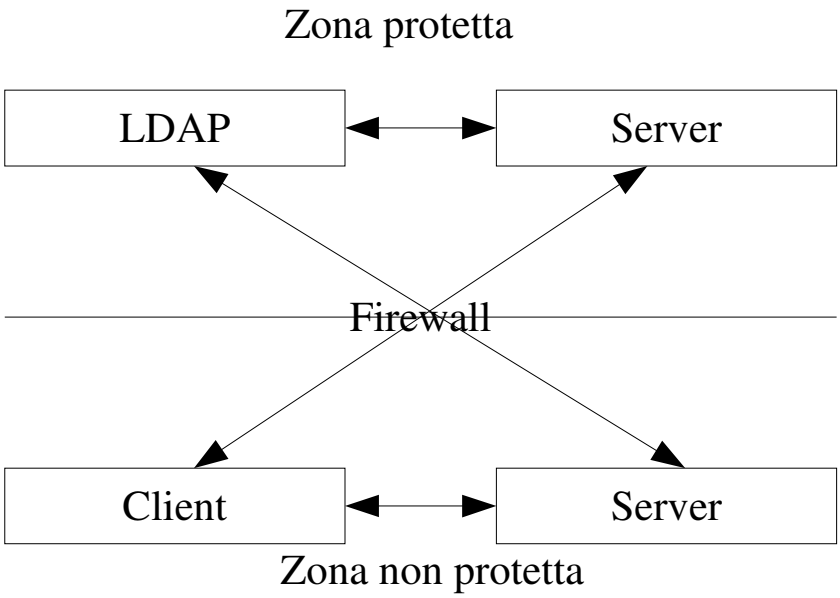

Figura 17 Transito delle password

3.5 Affidabilità e prestazioni

Il sistema di autenticazione è un servizio di base ed un suo malfunzionamento può bloccare l'intera struttura informatica, per questo è necessario renderlo affidabile: per questo scopo LDAP prevede la replicazione automatica dei dati tra più server.

La moltiplicazione dei server LDAP migliora anche le prestazioni suddividendo i carichi di lavoro, in reti WAN è opportuno distribuire i server nelle varie sedi in modo da ridurre il traffico sulle dorsali ed avere una maggiore velocità di risposta.

Le prestazioni di un sistema UNIX che utilizza LDAP per la gestione degli utenti può essere migliorata utilizzando NSCD (Name Service Cache Daemon), sistema di cache per tutti i name service, mediante il quale si riduce anche il carico di lavoro dei server LDAP. NSCD introduce il problema del ritardo nell'aggiornamento delle password contenute nella cache, ma due parametri di configurazione permettono di governare il problema: *positive-time-to-live* e *negative-time-to-live* che specificano rispettivamente il tempo di vita degli account nella cache che sono stati utilizzati con successo o senza successo. Determinante per l'efficacia della cache è la dimensione che viene determinata dal parametro *suggested-size*, al quale, per problemi di ottimizzazione, è opportuno associare un numero primo come valore.

I server LDAP prevedono la possibilità di utilizzare degli indici e l'indicizzazione delle informazioni, in maniera analoga a quanto avviene per i database, aumenta la velocità di reperimento delle informazioni ma rallenta quella di inserimento, e in ogni caso l'utilizzo di indici è assolutamente consigliato.

Test sulle prestazioni sono stati effettuati dalla Università di Salford e in [1] vengono confrontate le prestazioni di 7 server LDAP, mentre in [2] vengono verificati, utilizzando OpenLDAP, quali componenti influenzano le prestazioni di un server LDAP.

3.6 Tracciamento degli account di amministrazione

Per tracciare le operazioni svolte dagli utenti di amministrazione, e limitarne le possibilità, esistono diverse soluzioni, la più drastica delle quali è di utilizzare ogni machina con un solo servizio o per i servizi gestiti da un solo amministratore. Questa soluzione è limitata dai costi e dalla necessità sempre più frequente di avere macchine in cluster per garantire la disponibilità dei servizi. In linea teorica è possibile ricorrere, per l'amministrazione di un servizio, ad account non privilegiati opportunamente configurati, ma tale opportunità si scontra con notevoli difficoltà di implementazione e con la necessità di avere alcune risorse condivise. La soluzione più largamente utilizzata è di ricorrere ad opportuni software che permettono ad utenti non privilegiati l'esecuzione di un predeterminato set di comandi, e uno degli strumenti più diffusi a questo scopo è il Super User Do (SUDO).

Con SUDO è possibile limitare i comandi eseguibili da singoli o gruppi di utenti, eventualmente proteggendoli con password ed eseguendoli come altro utente, mentre il software tiene traccia, mediante log, di ogni comando eseguito e da chi è stato eseguito; al riguardo è opportuno predisporre una macchina destinata a ricevere i log di tutti i sistemi, configurata in modo che non sia gestita dagli stessi amministratori. Nell'utilizzo di SUDO o di software analoghi si deve prestare attenzione ad un fenomeno chiamato *Shell Escape*:

Generalmente è difficile limitare il set di comandi per un amministratore, senza limitarne l'efficienza, si tende quindi a lasciargli utilizzare tutti i comandi e ci si accontenta di tenere traccia di quanto eseguito con i log, in tal modo è facile eludere i controlli lanciando una nuova shell, in quanto i comandi eseguiti all'interno della shell non vengono tracciati: l'unica traccia è l'utilizzo della shell. Un problema più subdolo è dato da tutti quei software che prevedono una shell interna, come ad esempio Vi. Se nell'utilizzo di Vi viene usata la shell, i

comandi sfuggono al log, ma rimedi possibili sono l'utilizzo di librerie non standard, che impediscono l'esecuzione in cascata di programmi da software mandati in esecuzione con SUDO, oppure utilizzare versioni opportunamente modificate di tutti i software che presentano shell interne.

4 Implementazione presso l'Azienda Ospedaliera Pisana

In collaborazione con i responsabili del sistema informativo dell'Azienda Ospedaliera Pisana è stato valutato il possibile impiego di un sistema di autenticazione centralizzato presso il centro di calcolo aziendale; in questo capitolo viene descritto quanto è stato sperimentato al riguardo.

4.1 Struttura dei servizi dell'AOP

La struttura della rete è descritta in Figura 18: il nodo centrale è il firewall, ad esso affluiscono la connessione ad Internet, la DMZ (Demilitarized Zone) con i server pubblici, e la rete di campus costituita da varie reti IP distribuite sui 45 palazzi ed i tre siti principali dell'ospedale. La DMZ ospita i servizi pubblici costituiti dal sito web dell'azienda, il server DNS ed il server SMTP utilizzato come relay verso il server interno. Quest'ultimo ospita il sito web interno, il DNS interno, il server SMTP, il POP3 ed il file server Samba; la rete di campus è accessibile anche da remoto sfruttando un NAS e i server utilizzati sono realizzati con cluster a 2 nodi in bilanciamento di carico.

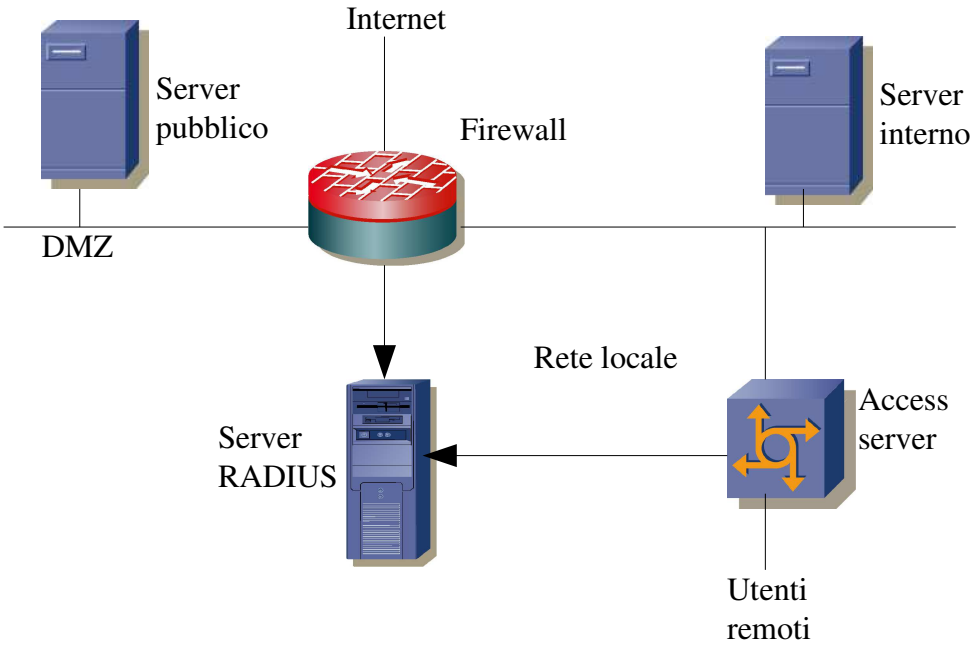

Figura 18 Struttura della rete e dei servizi dell'AOP

I servizi erogati necessitano di autorizzazione. In particolare i server utilizzano gli utenti locali per il POP3, il Samba e l'FTP che viene utilizzato dai reparti per l'aggiornamento delle sezioni di loro interesse del sito web. L'accesso ad Internet è consentito solo agli utenti autorizzati e per tale scopo il firewall utilizza un server RADIUS, utilizzato anche dal NAS per gli utenti

che accedono tramite linea commutata e sono infine presenti, in alcuni palazzi, file server di reparto con autenticazione locale.

Riepilogando, le informazioni di autorizzazione sono presenti in quattro punti distinti: il server RADIUS, i server di reparto, il server interno e quello esterno; in considerazione della dimensione della rete locale, costituita da circa 3000 personal computer ed utilizzata da 5000 dipendenti, l'aggiornamento delle informazioni in strutture di queste dimensioni è un problema consistente e costoso.

4.2 Individuazione dell'ambito di utilizzo

I servizi utilizzati in azienda possono essere modificati al fine di utilizzare LDAP per l'autenticazione e l'autorizzazione centralizzata. L'hardware disponibile inizialmente per lo sviluppo del sistema, che verrà successivamente messo in produzione, è descritto nella Tabella 3 e nella Tabella 4.

Per motivi di sicurezza i servizi sono stati divisi su due sistemi, uno dedicato all'autenticazione (RADIUS, CA, LDAP) che chiameremo *ATC* e l'altro per i rimanenti servizi (web, posta e file server), quest'ultimo verrà chiamato *Service*.

Cluster dei servizi di autenticazione	
Hardware	
Processore	Atlhon XP 3000
Memoria	1GByte
Disco fisso	40GByte
Software	
Sistema operativo	OpenBSD 3.5
Server LDAP	OpenLDAP v2.1.xx
Server RADIUS	FreeRADIUS v1.6.x
Secure Socket Layer	OpenSSL v0.9.x

Tabella 3Sistema per l'autenticazione

Cluster dei servizi	
	Hardware
Processori	2 Pentium III a 933MHz
Memoria	1GByte
Controller/Disco fisso	Adaptec 3960D Ultra160 SCSI
Storage	2 controller Promise RM 15000 con 2,5TByte
	Software
Sistema operativo/Kernel	Suse 9.1 Professional/Kernel 2.6.9 SMP
Server SMTP	Postfix 2.0.x
Server POP3	Courier IMAP 3.0.x
Software per la realizzazione del cluster	Heartbeat 1.2.x
Client LDAP	OpenLDAP 2 client 2.2.x
Server FTP	ProFTPD 1.2.x

Tabella 4 Sistema per la posta elettronica

4.3 Configurazione dei servizi di autenticazione

L'utilizzo della Certification Authority al momento è limitata esclusivamente alla gestione dei certificati utilizzati per l'autenticazione dei servizi che accedono al server LDAP, gli amministratori dell'infrastruttura ne prevedono in futuro un uso più esteso; per le attuali esigenze è sufficiente l'utilizzo di OpenSSL, costituito di un insieme di strumenti, programmi e librerie, per l'implementazione dei protocolli SSL e TLS.

Gli strumenti messi a disposizione sono sufficienti per implementare una CA, ma va segnalato che la gestione della CA è completamente manuale e se ne è previsto un uso intensivo è consigliabile ricorrere a software dedicati quali ad esempio OpenCA (http://www.openca.org) e ElyCA (http://elyca.eurodev.net), che si trattano di software web-based con licenza open source; per una disamina della struttura e delle funzioni di una CA si rimanda a [OSPKI].

I certificati possono essere creati seguendo le indicazioni contenute in [OpenSSLHOWTO], i certificati necessari sono quelli della CA di tipo self-signed, che verrà poi utilizzato per firmare i certificati dei server; tutti i certificati sono crittografati secondo lo standard RSA ed esportati in formato PEM. Durante il processo di creazione dei certificati viene richiesto il Common Name. Questo parametro nei certificati utilizzati dai server si riferisce al nome di dominio, affinchè l'autenticazione vada a buon fine è necessario che il nome riportato nel

certificato corrisponda al nome con il quale il client risolve l'indirizzo.

I server LDAP presenti sui due nodi del cluster vengono utilizzati in modalità Master-Slave, in tale configurazione le modifiche ai dati possono essere apportate solo al Master, e successivamente verranno riportate sul server slave. La replicazione in OpenLDAP avviene con le modalità illustrate in Figura 19: il server master ad ogni modifica aggiorna il *replication log*, che viene successivamente utilizzato dal *demone slurpd,* che provvede ad aggiornare i server slave. La comunicazione tra i server può essere opportunamente protetta con l'utilizzo di SSL.

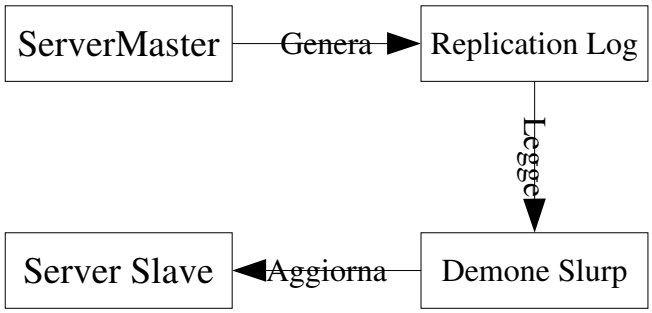

Figura 19 Duplicazione in OpenLDAP

La struttura dell'albero LDAP è rappresentata in Figura 20; in tale struttura vi sono due rami principali, uno destinato alle informazioni generali inerenti gli utenti e l'altro per i servizi. In Figura 21 è rappresentata la struttura dei nodi relativi ai servizi, in tale struttura tutti i rami che

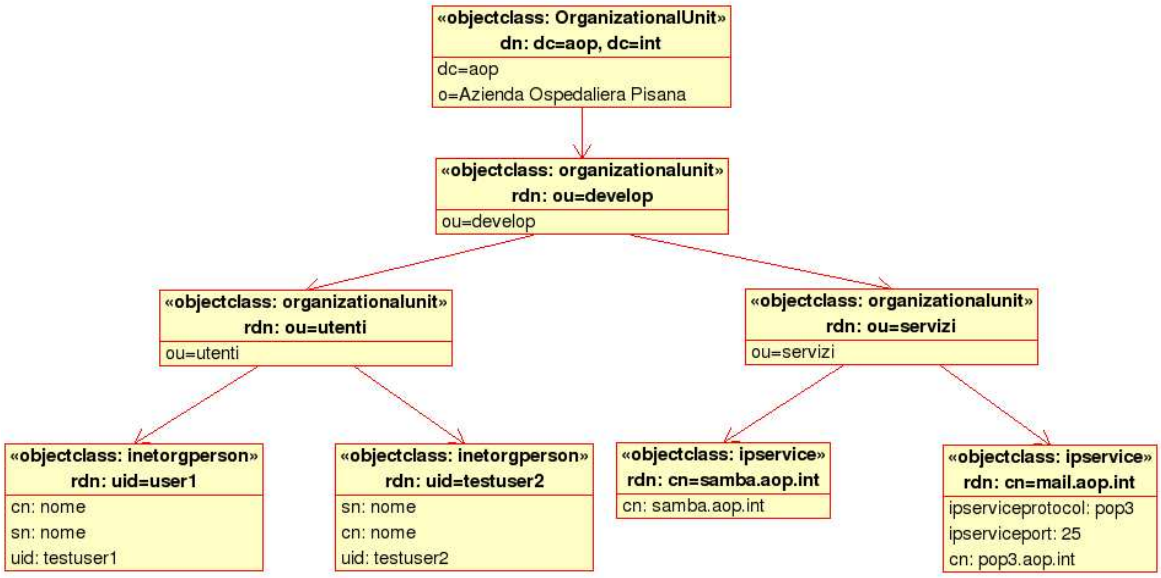

Figura 20 Struttura dell'albero LDAP

si dipartono dai servizi rappresentano gli account che possono accedere al servizio.

L'utilizzo del server RADIUS su OpenBSD si è rivelata difficoltosa con problemi a runtime nell'utilizzo di LDAP, per tale motivo, almeno in questa fase, è stato spostato sul cluster dei servizi, la configurazione del servizio è risultata priva di difficoltà ed è chiaramente illustrata in [HOWTOLDAP].

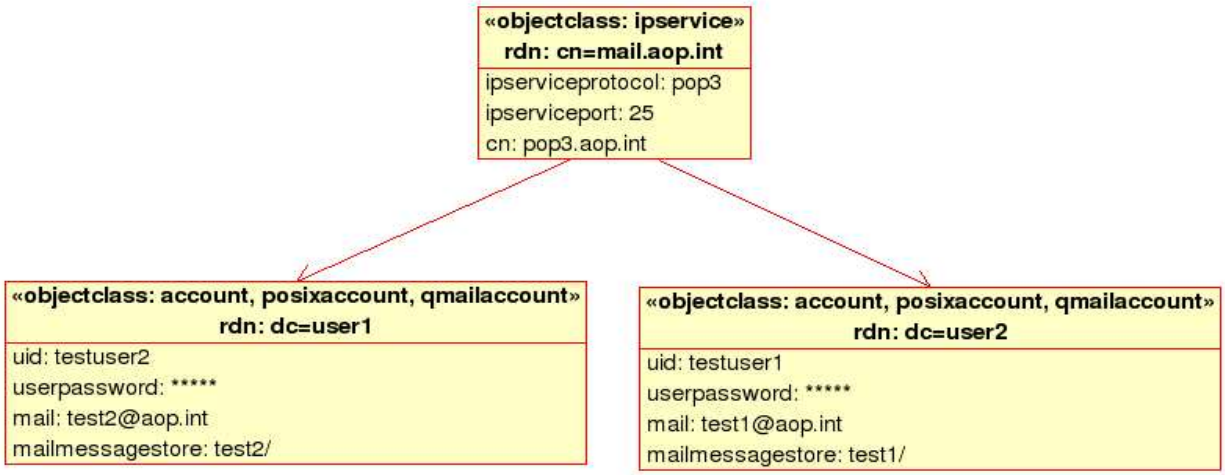

Figura 21 Ramo LDAP di un servizio

4.4 Configurazione dei servizi

Il server di posta permette l'utilizzo dei domini virtuali e degli utenti virtuali, con tale configurazione non è necessario che gli utenti del servizio siano anche utenti del sistema, con effetto benefico sulla sicurezza. Postfix permette di spostare sul server LDAP tutte le informazioni di configurazione che vengono tradizionalmente poste su file, tra queste la lista dei domini, l'associazione utenti email e gli alias.

La configurazione scelta prevede l'utilizzo di LDAP solo per la mappatura utenti-email, e l'utilizzo di *Maildir* come sistema di archiviazione per problemi di compatibilità con il server POP e di realizzazione del cluster. Di seguito viene riportata la porzione del file di configurazione riguardante l'accesso LDAP:

```
1.virtual_mailbox_domains = aop.int
2.virtual_mailbox_base = /home/virtual
3.virtual_mailbox_maps = ldap:/etc/postfix/ldap-aliases.cf
4.virtual_uid_maps = static:5000
5.virtual_gid_maps = static:5000
6.virtual_home_mailbox = /
```

Nella 3 riga viene specificato l'utilizzo di LDAP per la mappatura delle caselle di posta ed il file di configurazione per l'accesso al server LDAP, nella 4 e nella 5 l'UID ed il GID da utilizzare nell'accesso al file system.

Postfix quando riceve una nuova email per un dominio gestito, deve trasferirla nella Maildir del destinatario, l'associazione destinatario email tramite LDAP viene realizzata mediante una ricerca basata sulla email, uno degli attributi della entry eventualmente trovata viene utilizzato per determinare il percorso della Maildir.

```
1.server_host = ldap://ldap.aop.int ldap://ldap1.aop.int
2.version = 3
3.search_base=cn=smtp.aop.int,ou=servizi,ou=develop,dc=aop,dc=
  int
4.query_filter = (&(mail=%s))
5.result_attribute = mailMessageStore
```

Le righe più significative sono la prima che specifica un pool di server LDAP da utilizzare in sequenza in caso di mancato accesso al precedente, la terza indica il nodo di partenza per la ricerca, la quarta il filtro (%s viene sostituito dalla email) e nella quinta il nome dell'attributo da utilizzare.

Il server POP3 utilizza LDAP per l'autenticazione e per costruire il percorso della Maildir, l'autenticazione può essere svolta direttamente dal server LDAP utilizzando i meccanismi di autenticazione standard (*binding* degli utenti) o dal server POP3 che in tal caso utilizza LDAP solo come base dati delle credenziali ed effettua la verifica delle stesse in proprio.

La configurazione del server FTP non riserva particolari problemi e si risolve nel configurare il server LDAP da utilizzare e l'utente LDAP per il binding iniziale.

La configurazione del server Samba è dipendente dalla versione utilizzata. Tutte le versioni successive alla 2.x supportano l'autenticazione e la definizione degli utenti tramite server LDAP, dalla 3.x anche i domini Windows possono essere definiti con LDAP, inoltre, per conflitti sui nomi utilizzati, gli objectClass della versione 3.x differiscono dalla precedente pur mantenendo la compatibilità. Il server di sviluppo rispecchia la configurazione del servizio attualmente in produzione che non utilizza i domini, tale limite semplifica la messa in opera che si riduce a specificare il server LDAP e le modalità di accesso.

L'autenticazione tramite PAM non viene attualmente utilizzata in quanto al server hanno accesso solo gli utenti di amministrazione per i quali si è preferito mantenere l'autenticazione standard, per completezza ed in previsione di possibili utilizzi futuri ne è stato comunque verificato il funzionamento.

Il PAM può utilizzare LDAP per l'autenticazione e per le tabelle *passwd*, *shadow* e *group*. La

configurazione, oltre ai normali parametri quali server LDAP e filtri, presenta delle particolarità interessanti:

- Utilizzando il parametro *pamgroupdn* è possibile specificare il dn di un gruppo al quale appartengono gli utenti autenticati tramite LDAP.
- Il range di uid per i quali accettare l'autenticazione su LDAP può essere specificato tramite *pam_min_uid* e *pam_max_uid*, in tal modo è possibile che gli account amministrativi utilizzino l'autenticazione locale.
- Le tabelle residenti su LDAP vengono specificate con parametri del tipo:
  `nss_base_[nome tabella]  base?scope?filter`
  è possibile indicare il nodo di partenza (base), la profondità di ricerca (scope) ed il filtro da applicare, ma solo il primo parametro è obbligatorio.

La configurazione del PAM è modulare e può essere diversificata per servizio, ad esempio il modulo relativo al programma *su* (permette l'utilizzo di shell con utente diverso da quello attuale) è così configurato:

```
auth       required  pam_ldap.so
account    required  pam_ldap.so
password   required  pam_ldap.so
session    required  pam_unix2.so  none
```

La prima riga richiede l'autenticazione tramite LDAP, la seconda è per la gestione delle shadow password, la terza per la modifica delle password e l'ultima gestisce il log.

4.5 Applicazione di gestione

La soluzione più semplice ed immediatamente disponibile è l'utilizzo di script di shell che facilitino l'inserimento dei dati nell'albero LDAP, mentre una soluzione di più largo respiro e di implementazione ragionevolmente complessa può essere sviluppata utilizzando Java. Sono stati effettuati dei test con tale linguaggio che hanno evidenziato l'ottimo e semplice supporto di LDAP. Per un esempio vedere 6.2.1.

La distribuzione di OpenLDAP fornisce dei tool per l'inserimento (*ldapadd* e *ldapmodify*) e la ricerca dei dati (*ldapsearch*). Questi possono essere opportunamente configurati per connettersi al server LDAP in modalità protetta, in appendice vengono riportati alcuni degli script realizzati e, data la semplicità degli stessi, in questo paragrafo sono commentate soltanto le istruzioni più significative:

L'inserimento dei dati avviene utilizzando *ldapmodify*, questo comando accetta entry in standard ldif da standard input o da file, ad esempio:

```
ldapmodify 1>>ldap.log 2>>ldap.log -f temp.ldif -x -D \
"cn=Manager,dc=aop,dc=int" -w **** -v -H
'ldaps://ldap.aop.int' -a
```

I parametri più interessanti sono: l'utente LDAP utilizzato per l'inserimento che è specificato dall'opzione -D, il server da contattare con l'opzione -H, relativamente alla quale si osserva che l'URI *ldaps://* fa riferimento al protocollo criptato tramite SSL, infine l'opzione -a (add) trasforma *ldapmodify* in *ldapadd*.

La verifica dell'esistenza di un attributo può essere ottenuta ad esempio con il seguente codice:

```
ldapsearch -LLL -H ldaps://ldap.aop.int\
-b 'ou=utenti,ou=develop, dc=aop,dc=int'\
"(&(objectclass=inetorgperson)($1=$2))" cn sn uid > tmp.txt
ris=$(awk '{if ($1 == "uid:") print $2}' tmp.txt)
```

in tale esempio viene verificato che tra gli oggetti corrispondenti al filtro ve ne sia almeno uno con attributo uid uguale a '$1'.

4.6 Test del sistema

Il corretto funzionamento dei servizi è stato sottoposto a test non esaustivi ma sufficienti a determinarne il corretto funzionamento.

Il servizio di replicazione è stato controllato apportando modifiche al server master e verificando l'avvenuta modifica sul server slave; ad ulteriore prova del funzionamento sono state effettuate modifiche con il server slave non attivo che sono state correttamente riportate quando è tornato disponibile.

La struttura generale dell'albero LDAP è riportata in Figura 20, nel ramo utenti sono state inserite due entry con le informazioni minime necessarie, del tipo:

```
dn: uid=test,ou=utenti,ou=develop,dc=aop,dc=int
objectClass: inetorgperson
uid: test
cn: test
sn: test-one
```

Nel ramo dei servizi sono stati inseriti 4 nodi relativi al RADIUS, al Samba, la posta elettronica ed un nodo generico utilizzato per PAM e FTP, ad esempio viene riportata la entry del servizio di posta:

```
dn: cn=smtp.aop.int,ou=servizi,ou=develop,dc=aop,dc=int
objectClass: ipservice
cn: smtp.aop.int
ipServiceProtocol: smtp
ipServicePort: 25
```

Tutti i servizi sono stati configurati per accedere ai due server LDAP ed è stato controllato l'effettivo utilizzo del server di supporto in assenza del server principale.

Il corretto funzionamento dell'autenticazione tramite certificati è stata verificata sfruttando la possibilità offerta da OpenSSL di simulare un client generico, al riguardo viene riportata la riga di comando utilizzata per il test dell'autenticazione al server LDAP:

```
openssl s_client -connect ldap.aop.int:636 -showcerts -state \
-CAfile LocalCA/cert.pem -cert LocalCA/test_crt.pem -key \
LocalCA/test_key.pem
```

L'effettivo utilizzo di protocolli crittografati è stato verificato sfruttando analizzatori di protocollo.

4.6.1 Posta elettronica

Al nodo del servizio di posta sono stati aggiunti i due account:

```
dn:
uid=test1,cn=smtp.aop.int,ou=servizi,ou=develop,dc=aop,dc=int
objectClass: account
objectClass: qmailuser
uid: test1
userPassword:: Z3No
mail: test1@aop.int
mailMessageStore: test1/
dn:
uid=test,cn=smtp.aop.int,ou=servizi,ou=develop,dc=aop,dc=int
accountStatus: test
objectClass: account
objectClass: qmailuser
uid: test
uid: test-one
userPassword:: dGVzdGluYQ==
userPassword:: dGVzdG9uZQ==
mail: test@aop.int
mailMessageStore: test/
```

I test effettuati sono stati l'invio e la ricezione di email, la seconda entry utilizza 2 uid e 2 password differenti ed è stato verificato che le quattro possibili combinazioni consentano l'accesso alla casella di posta.

4.6.2 RADIUS

Le entry inserite come foglie del nodo relativo al servizio RADIUS sono del tipo:

```
dn:
uid=test,cn=radius.aop.int,ou=servizi,ou=develop,dc=aop,dc=int
objectClass: account
objectClass: radiusprofile
objectClass: simplesecurityobject
uid: test
userPassword:: dGVzdA==
cn: test
```

si evidenzia l'utilizzo dell'objectClass radiusprofile con le informazioni specifiche del server RADIUS e simplesecurityobject per la password.

La verifica del corretto funzionamento è stata effettuata utilizzando il tool radtest fornito con il pacchetto FreeRADIUS, con tale software è possibile analizzare il dialogo che intercorre tra client e server e l'esito dell'autenticazione.

4.6.3 PAM

I dati inseriti nel server LDAP per sottoporre a test l'autenticazione tramite PAM sono del tipo:

```
dn:uid=test,cn=generic.aop.int,ou=servizi,ou=develop,dc=aop,dc
=int
gidNumber: 500
homeDirectory: /home/test
userPassword:: Z3No
loginShell: /bin/bash
cn: test
objectClass: account
objectClass: posixAccount
objectClass: shadowAccount
uid: test
shadowMin: 0
shadowMax: 10
shadowWarning: 7
shadowInactive: 5
shadowExpire: 12739
shadowLastChange: 12730
uidNumber: 500
```

Il servizio su del PAM è stato configurato come:

```
auth       required  pam_ldap.so
account    required  pam_ldap.so
password   required  pam_ldap.so
```

```
session   required  pam_unix2.so  none
```

I test hanno consistito nell'autenticazione sul sistema utilizzando la password inserita in LDAP e verificando che la stessa non permette l'accesso agli altri servizi.

4.6.4 Samba

I test sono stati effettuati utilizzando il tool *smbclient* fornito con la suite Samba. Sfruttando smbclient è possibile ottenere la lista dei servizi e delle condivisioni, il risultato viene di seguito riportato:

```
Domain=[TEST] OS=[Unix] Server=[Samba 3.0.4]
Sharename       Type      Comment
homes           Disk      Home Directories
home            Disk      My home
IPC$            IPC       IPC Service (SAMBA-LDAP PDC Server)
ADMIN$          IPC       IPC Service (SAMBA-LDAP PDC Server)
test            Disk      Home Directories
Domain=[TEST] OS=[Unix] Server=[Samba 3.0.4]
Server               Comment
TEST                 Samba 3.0.4
```

Lo stesso tool può essere utilizzato per accedere alle cartelle condivise con funzionalità simili ai client FTP, e con tale modalità è stato verificato l'effettivo funzionamento delle cartelle condivise.

4.6.5 ProFTPD

I test si sono risolti nell'autenticazione e nel trasferimento di file.

5 Conclusioni

L'elevato numero di attacchi informatici, l'importanza dei dati, i costi di una loro eventuale perdita, la legge sulla privacy, che obbliga alla protezione delle informazioni, spingeranno con forza ad implementare sistemi di autenticazione centralizzata; questa esigenza viene coperta con difficoltà dai sistemi oggetto di questo studio, che ha evidenziato lacune e carenze non trascurabili.

La ricerca ha suggerito sostanzialmente due possibili soluzioni individuabili in sistemi "preconfezionati" e sistemi custom.

I sistemi di autenticazione "preconfezionati" presentano il vantaggio non trascurabile della facilità di utilizzo e di impostazioni chiare e non derogabili; tali caratteristiche comportano la relativa facilità di progettazione e di messa in produzione, anche se va comunque osservato che, talvolta, nel tentativo di facilitare l'utilizzo e di coprire esigenze che vanno dal piccolo ufficio alle multinazionali, si è giunti a strutture farragginose, dispersive e tutt'altro che chiare. Le note dolenti di tali sistemi sono la scarsa possibilità di integrarsi in strutture eterogenee e la conseguente necessità di utilizzare più sistemi di autenticazione contemporaneamente, ma soprattutto la mancanza assoluta di un sistema di autorizzazione centralizzato.

I sistemi custom, ed in particolare la soluzione esposta in questo studio basata su LDAP, consentono una effettiva integrazione in strutture eterogenee e la possibilità di essere inseriti in contesti preesistenti anche complessi, con costi di migrazione contenuti. L'elevato numero di dispositivi e di software capaci di sfruttare tale sistema di autenticazione fanno supporre che sia un sistema destinato a costituire uno degli standard del futuro, giustificando in questo senso investimenti su questa tecnologia.

Non vanno certo sottovalutati gli aspetti negativi che derivano dall'adozione di tali tecnologie e che consistono principalmente nella necessità di una progettazione accurata dell'intero sistema, al fine di evitare lacune nel processo di autenticazione e di garantire una effettiva integrazione e scalabilità, e nell'esigenza di una maggiore preparazione degli amministratori di un sistema che coinvolge numerose tecnologie.

Una caratteristica decisiva nella valutazione di tali sistemi è la flessibilità dimostrata anche dal fatto che la soluzione custom è stata l'unica in grado di implementare un sistema di autorizzazione centralizzato, che pur con limitazioni, si è rivelato efficace ed in grado di

soddisfare esigenze diversificate come quelle di una realtà complessa come l'Azienda Ospedaliera Pisana, dove è stata abbracciata dagli amministratori del sistema informativo, che hanno deciso di utilizzarla per i sistemi in produzione.

6 Appendici

6.1 Acronimi

CIFS	Common Internet File System
CVS	Concurrent Versions System
DMZ	Demilitarized Zone
RDN	Relative Distinguished Name
DN	Distinguished Names
FTP	File Transfer Protocol
GSS-API	Generic Security Services Application Program Interface
HTTPS	Secure HTTP
IANA	Internet Assigned Numbers Authority
IETF	Internet Engineering Task Force
LDAP	Lightweight Directory Access Protocol
LDIF	LDAP Data Interchange Format
MITM	Man-In-The-Middle
NSCD	Name Service Cache Daemon
OSI	Open Systems Interconnection
PKI	Public Key Infrastructure
SASL	Simple Authentication and Security Layer
SMB	Server Message Block
SSL	Secure Sockets Layer
TLS	Transport Security Layer
URI	Uniform Resource Identifier
WWW	World Wide Web
WebDAV	Web-based Distributed Authoring

6.2 Script di gestione

Script per l'inserimento di un nodo relativo ad un nuovo servizio:

```
#!/bin/bash
BASE="ou=servizi, ou=develop, dc=aop, dc=int"
echo "Inserimento nuovi server"
echo "Digitare il nome del server (es. smtp.aop.int)"
read cn
if [ "$cn" = "" ]; then
  echo "ERRORE: il nome non può essere nullo"
  exit 1
fi
echo "Digitare il protocollo (es. smtp, pop3 ecc.)"
read proto
if [ "$proto" = "" ]; then
  echo "ERRORE: il protocollo non può essere nullo"
  exit 1
fi
echo "Digitare la porta utilizzata (25, 110 ecc.)"
read port
if [ "$port" = "" ]; then
  echo "ERRORE: la porta non può essere nulla"
  exit 1
fi
echo "Descrizione (Opzionale)"
read desc
echo dn: cn=$cn, $BASE >tempuser.ldif
echo objectclass: ipservice >>tempuser.ldif
echo cn: $cn >>tempuser.ldif
echo ipserviceprotocol: $proto >>tempuser.ldif
echo ipserviceport: $port >>tempuser.ldif
if [ -n "$desc" ]; then
        echo description: $desc >>tempuser.ldif
fi
echo "ldif======================"
cat tempuser.ldif
echo "=========================="
ldapmodify 1>>ldap.log 2>>ldap.log  -f tempuser.ldif -x -D
"cn=Manager, dc=aop, dc=int" -w 0penldaP -v -H
'ldaps://ldap.aop.int' -a
errore=$?
echo ldapmodify return code $errore
if [ "$errore" = "0" ]; then
  echo "OK: server inserito correttamente"
  exit 0
fi
if [ "$errore" = "68" ]; then
  echo "ERRORE: il server è già presente"
  exit 1
```

```
fi
```

Script per l'inserimento di un nuovo utente:

```
#!/bin/bash
BASE="ou=utenti, ou=develop, dc=aop, dc=int"
echo "Inserimento nuovi utenti"
echo "Digitare uid"
read uid
if [ "$uid" = "" ]; then
  echo "ERRORE: uid non può essere nullo"
  exit 1
fi
echo "Digitare il nome"
read nome
if [ "$nome" = "" ]; then
  echo "ERRORE: il nome non può essere nullo"
  exit 1
fi
echo Digitare il cognome
read cognome
if [ "$cognome" = "" ]; then
  echo "ERRORE: il cognome non può essere nullo"
  exit 1
fi
echo "Casella di posta?"
read mail

echo dn: uid=$uid, $BASE >tempuser.ldif
echo objectclass: inetorgperson >>tempuser.ldif
echo uid: $uid >>tempuser.ldif
echo cn: $nome >>tempuser.ldif
echo sn: $cognome >>tempuser.ldif
if [ -n "$mail" ]; then
        echo mail: $mail >>tempuser.ldif
fi
ldapmodify 1>>ldap.log 2>>ldap.log  -f tempuser.ldif -x -D \
"cn=Manager,dc=aop,dc=int" -w ** -v -H 'ldaps://ldap.aop.int'
-a
errore=$?
echo ldapmodify return code $errore
if [ "$errore" = "0" ]; then
  echo "OK: l'utente è stato inserito"
  exit 0
fi
if [ "$errore" = "68" ]; then
  echo "ERRORE: l'utente è già presente"
  exit 1
fi
```

Script per l'inserimento di account in uno specifico nodo:

```
#!/bin/bash
USER="ou=utenti, ou=develop, dc=aop, dc=int"
BASESMTP="cn=smtp.aop.int, ou=servizi, ou=develop, dc=aop,
dc=int"

function ldapsearchtest {
        ldapsearch -LLL -H ldaps://ldap.aop.int -b 'ou=utenti,
ou=develop, dc=aop, dc=int' "(&(objectclass=inetorgperson)
($1=$2))" cn sn uid > tmp.txt
        ris=$(awk '{if ($1 == "uid:") print $2}' tmp.txt)
        nc=$(awk -v cognome="" -v nome="" '{if ($1 == "sn:")
cognome=$2} {if ($1 == "cn:") nome=$2} END {print cognome,
nome}' tmp.txt)
        nome=$nc
        echo $nc
        if [ -z $ris ];
                then return 0
                else return 1
        fi
}

echo "Inserimento nuovi utenti"
echo "Digitare uid"
read uid
echo Verifica presenza utente

ldapsearchtest uid $uid $nome
ris=$?
if [ "$ris" = "0" ];
  then echo "L'$uid non è presente, inserire prima l'utente"
  exit 1
fi
echo "Digitare la password"
read password
if [ "$password" = "" ]; then
  echo "ERRORE: la password non può essere nulla"
  exit 1
fi

if [ -z "$mail" ]; then
  echo "Inserire l'email"
  read mail
  if [ -z "$mail" ]; then
    echo "l'email non può essere nulla"
    exit 1
  fi
fi
```

```
echo dn: uid=$uid, $BASESMTP >tempuser.ldif
echo objectclass: account >>tempuser.ldif
echo objectclass: qmailuser >>tempuser.ldif
echo uid: $uid >>tempuser.ldif
echo userpassword: $password >>tempuser.ldif
echo mail: $mail >>tempuser.ldif
echo mailmessagestore: $uid/ >>tempuser.ldif
ldapmodify 1>>ldap.log 2>>ldap.log  -f tempuser.ldif -x -D \
"cn=Manager,dc=aop,dc=int" -w *** -v -H 'ldaps://ldap.aop.int'
-a
errore=$?
erroremail=$?
echo ldapmodify return code $errore
if [ "$errore" = "0" ]; then
  echo "OK: l'utente è stato inserito"
  exit 0
fi
if [ "$errore" = "68" ]; then
  echo "ERRORE: l'utente è già presente"
  exit 1
fi
```

6.2.1 Programma Java per l'accesso ad LDAP

Semplice software per la verifica della funzionalità del Java nell'accesso ai server LDAP.

```
import javax.naming.Context;
import javax.naming.NamingEnumeration;
import javax.naming.directory.Attribute;
import javax.naming.directory.BasicAttribute;
import javax.naming.directory.InitialDirContext;
import javax.naming.directory.DirContext;
import javax.naming.directory.Attributes;
import javax.naming.directory.ModificationItem;
import javax.naming.NamingException;
import java.util.Hashtable;

class Getattr {
    public static void main(String[] args) {
        // Binding
        Hashtable env = new Hashtable(11);
        env.put(Context.INITIAL_CONTEXT_FACTORY,
            "com.sun.jndi.ldap.LdapCtxFactory");
        env.put(Context.PROVIDER_URL,
            "ldap://172.31.10.3:389/dc=aop,dc=int");
        env.put(Context.SECURITY_PRINCIPAL,
            "cn=Manager,dc=aop,dc=int");
        env.put(Context.SECURITY_CREDENTIALS, "secret");

        try {
```

```
            DirContext ctx = new InitialDirContext(env);
            // Richiede gli attributi di un oggetto
            Attributes answer = ctx.getAttributes("cn=java");
            // Cerca ("sn") e lo stampa
            System.out.println("sn: " + answer.get("sn").get
());
            //Stampa tutti gli attributi
            for (NamingEnumeration ae=answer.getAll();
ae.hasMore();)
            {
                Attribute attr = (Attribute)ae.next();
                System.out.println("attribute: "+attr.getID());
                for (NamingEnumeration e=attr.getAll();
                    e.hasMore();
                    System.out.println("value: " + e.next()));
            }
            //Modifica dei dati
            ModificationItem[] mods = new ModificationItem[1];
            answer = ctx.getAttributes("cn=java");
            Attribute desc = answer.get("description");
            desc.add(3,"desc5");
            mods[0] = new ModificationIte
(DirContext.ADD_ATTRIBUTE,
                new BasicAttribute("description", "desc_02"));
            ctx.modifyAttributes("cn=java", mods);
            ctx.close();
        } catch (NamingException e) {
            System.err.println("Problem getting attribute: " +
e);
        }
    }
}
```

Bibliografia

1: E. J. Thornton, J. Mundy, D. W. Chadwick, A comparative performance analysis of 7 ldap
2: Xin Wang, Henning Schulzrine, Dilip Kandlur, Dinesh Verma, Measurement and Analysis of LDAP Performance
draft-freier-ssl-version3-02: Alan O. Freier, Philip Karlton, Paul C. Kocher, The SSL Protocol, Novembre 1996
draft-iab-auth-mech-03: E. Rescorla, A Survey of Authentication Mechanisms, Marzo 2004
draft-ietf-ldapbis-authmeth-11: R. Harrison, LDAP: Authentication Me, Luglio 2004
FIPS PUB 180-1: ational Institute of Standards and Technology, SECURE HASH STANDARD, Aprile 1995
HOWTOLDAP: Giuseppe Lo Biondo, LDAP Implementation HOWTO, 2001
OpenSSLHOWTO: vari, OpenSSL HOWTO,
http://sapiens.wustl.edu/~sysmain/info/openssl/index.html
OSPKI: Symeon (Simos) Xenitellis, The Open source PKI Book, 2000
OW-001-tac_plus: SARA, Common Vulnerabilities and Exposures-TACACS Server, 10 luglio 2000
Password Management: Matt Bishop, Password Management, 1991
RFC1274: P. Barker, S. Kille, The COSINE and Internet X.500 Schema, Novembre 1991
RFC1321: R. Rivest, The MD5 Message-Digest Algorithm, Aprile 1992
RFC1411: D. Borman, Telnet Authentication: Kerberos Version 4, Gennaio 1993
RFC1510: J. Kohl, C. Neuman, The Kerberos Network Authentication Service (V5), Settembre 1993
RFC1704: N. Haller, R. Atkinson, On Internet Authentication, Ottobre 1994
RFC1760: N. Haller, The S/KEY One-Time Password System, 1995
RFC2222: J. Myers, Simple Authentication and Security Layer (SASL),
RFC2246: T. Dierks, C. Allen, The TLS Protocol, Gennaio 1999
RFC2251: M. Wahl, T. Howes, S. Kille, Lightweight Directory Access Protocol (v3), Dicembre 1997
RFC2252: M. Wahl, A. Coulbeck, T. Howes, S. Kille, Lightweight Directory Access Protocol (v3): Attribute Syntax Definit, Dicembre 1997
RFC2253: M. Wahl, S. Kille, T. Howes, Lightweight Directory Access Protoc, Dicembre 1997
RFC2254: T. Howes, The String Representation of LDAP Search Filt, Dicembre 1997
RFC2255: T. Howes, M. Smith, The LDAP URL Format, Dicembre 1997
RFC2256: M. Wahl, A Summary of the X.500(96) User Schema for use with LDAPv3 , Dicembre 1997
RFC2289: N. Haller, C. Metz, P. Nesser, M. Straw, A One-Time Password System, Febbraio 1998
RFC2307: L. Howard, An Approach for Using LDA as a Network Information Service, Marzo 1998
RFC2444: C. Newman, The One-Time-Password SASL Mechanism, Ottobre 1998
RFC2510: C. Adams, S. Farrell, Internet X.509 Public Key Infrastructure Certificate Management Protocols, Marzo 1999
RFC2518: Y. Goland, E. Whitehead, A. Faizi, S. Carter, D. Jensen, HTTP Extensions for Distributed Authoring -- WEBDAV , Febbraio 1999

RFC2585: R. Housley, P. Hoffman, Internet X.509 Public Key Infrastructure Operational Protocols: FTP and HTTP, Maggio 1999
RFC2712: A. Medvinsky, M. Hur October 1999, Addition of Kerberos Cipher Suites to Transport Layer Security (TLS), Ottobre 1999
RFC2743: J. Linn, Generic Security Service Application Program Interface, Gennaio 2000
RFC2798: M. Smith, Definition of the inetOrgPerson LDAP Object Class, Aprile 2000
RFC2829: M. Wahl, H. Alvestrand, J. Hodges, R. Morgan, Authentication Methods for LDAP, Maggio 2000
RFC2830: J. Hodges, R. Morgan, M. Wahl, Lightweight Directory Access Protocol (v3): Extension for Transport Layer Security, Maggio 2000
RFC2865: C. Rigney, S. Willens,A. Rubens, W. Simpson, Remote Authentication Dial In User Service (RADIUS, Giugno 2000
RFC2942: T. Ts'o, Telnet Authentication: Kerberos , Settembre 2000
RFC3253: G. Clemm, J. Amsden, T. Ellison, C. Kaler, J. Whitehead, Versioning Extensions to WebDAV (Web Distributed Authoring and Versioning) , Marzo 2002
RFC3377: J. Hodges, R. Morgan, Lightweight Directory Access Protocol (v3): Technical Specification, Settembre 2002
Web 1: Eric Glass, The NTLM Authentication Protocol ,
http://davenport.sourceforge.net/ntlm.html
Web 2: MIT, Kerberos: The Network Authentication Protocol ,
http://web.mit.edu/kerberos/www/
Web 3: Peter Larsen and Jason ZionsPeter Larsen, Jason Zions, Mixing It Up: Windows, UNIX, And Active Directory,
http://rad.microsoft.com/ADSAdClient31.dll?GetAd=&PG=CMSTN1&SC=F3&AP=1164
Web 4: SUN, NIS+ End-of-Feature (EOF) Announcement FAQ,
http://wwws.sun.com/software/solaris/faqs/nisplus.html