# Sicurezza come “identity management”

Prof. Francesco Bergadano

Dipartimento di Informatica
Università degli Studi di Torino

bergadano@di.unito.it

# Sommario

La sicurezza informatica: da problema a soluzione....3
Servizi e prodotti di sicurezza informatica....4
La centralità dell'utente autorizzato....5
"Che cos'è" un utente autorizzato....6
Controllo di accesso....7
Enrollment , autenticazione e autorizzazione....8
Autenticazione e identificazione....10
Autenticazione utente....11
Two-factor authentication....11
Autenticazione con token....12
Autenticazione biometrica....13
Autenticazione con password....14
Protocolli di Autenticazione....15
Password temporizzate....22
One time password (OTP)....23
Autorizzazione e identity management....32
Gestione dei client e della rete....**Errore. Il segnalibro non è definito.**

## La sicurezza informatica: da problema a soluzione

Come è cambiata la percezione della sicurezza informatica

La sicurezza come problema:

- possibilità di intrusioni informatiche e “hacker”
- possibilità di inserimento di programmi dannosi e “virus”
- incertezza sull’identità dell’utente
- perdita di dati
- blocco di sistema e rete

La sicurezza come soluzione:

- gestire un data base di utenti con le loro credenziali di accesso
- gestire il parco macchine dei propri utenti
- garantire la disponibilità di dati e servizi
- garantire l’integrità dei server e dei dati
- garantire la riservatezza di storage e comunicazioni

## *Servizi e prodotti di sicurezza informatica*

Se la sicurezza è un problema: prodotti stand-alone e consulenza

- Security "probe" *una tantum* (vulnerability assessment - VA)
- Antivirus, Firewall, IDS etc. come prodotti da scaffale
- Sistemi di autenticazione proprietaria "sicuri" ma non integrati
- Replicazione "ad hoc" di dati e servizi
- Consulenza orientata al "problema" della sicurezza

Se la sicurezza è una soluzione: security management

- User management
- Gestione centralizzata di client e dispositivi
- Gestione e monitoraggio centralizzati di dati, server e rete:
    - Intrusion Detection / Prevention
    - Probe continuo e corrispondenti azioni correttive
    - Ridondanza / Recovery / Business continuity
    - Security analytics / reporting
- Progettazione sicura di infrastrutture e applicazioni
- Consulenza nella progettazione e nella gestione

### *La centralità dell'utente autorizzato*

Un servizio informatico quasi sempre deve
- essere disponibile per l'utente autorizzato
- non essere accessibile ad altri

E' quindi centrale la figura dell'utente autorizzato e la procedura attraverso la quale il sistema lo riconosce, in locale o in rete.

Il mercato, nel passare dalla sicurezza come problema alla sicurezza come soluzione, ha spostato la sua attenzione dall'utente non autorizzato o "hacker" all'utente che vuole e può usare il servizio, ma che deve essere identificato e abilitato all'accesso.

### *“Che cos’è” un utente autorizzato*

Un utente autorizzato, dal punto di vista informatico, comprende due componenti:

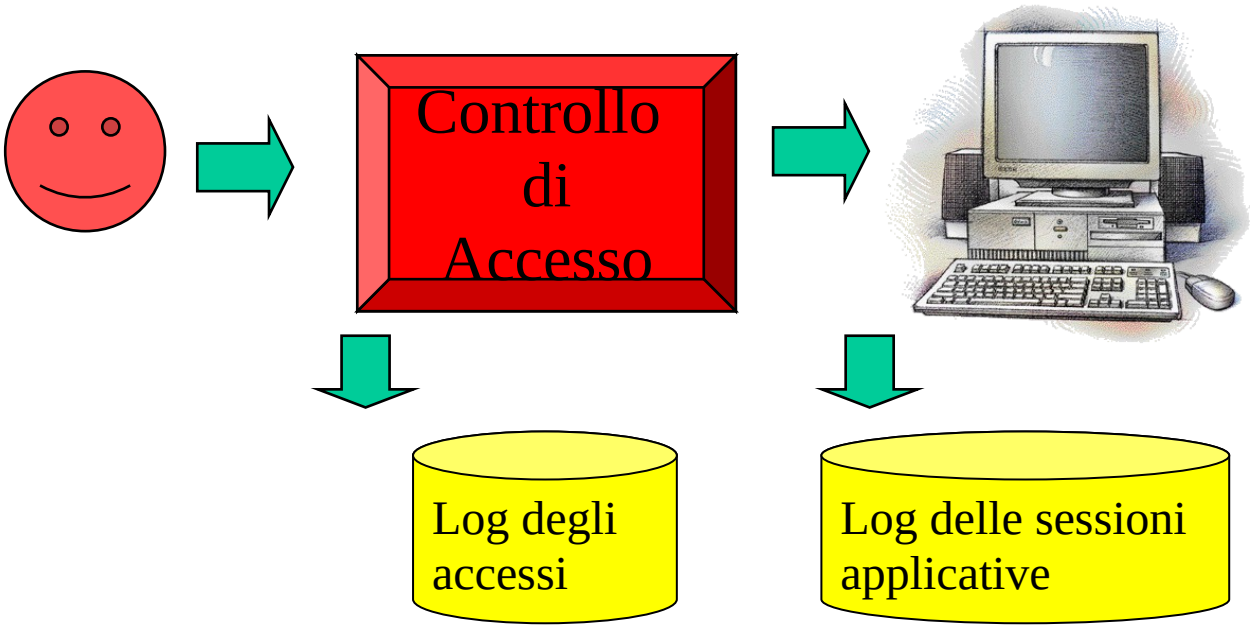
1. Un **identificativo**, che possiamo considerare il “nome” informatico dell’utente stesso. Insieme a questo identificativo spesso sono disponibili altre informazioni relative all’utente, di tipo anagrafico o necessarie per meglio qualificare l’utente nel contesto del particolare servizio al quale deve accedere. Uno stesso utente fisico potrà naturalmente corrispondere a più di un identificativo.

2. Un’informazione, che chiameremo “**credenziale** di accesso”, che permette di eseguire una procedura, che può avere due esiti:
   a. L’utente può accedere al servizio e si prosegue
   b. L’utente non può accedere al servizio e non potrà proseguire nel corrispondente utilizzo del sistema

   Può spesso accadere che vengano associate all’utente più credenziali di accesso, da utilizzarsi per accedere a diversi servizi o a diverse componenti dello stesso servizio. Può anche succedere che il sistema lato client non utilizzi direttamente una delle credenziali fornite, ma che la elabori per derivare l’informazione effettivamente necessaria per l’interazione con il lato server.

## Controllo di accesso

La procedura di verifica delle credenziali sopra citata viene detta "controllo di accesso" e permette quindi di creare una sorta di barriera tra l'utente e il servizio richiesto.




Scopo del controllo di accesso:

- Evitare utilizzi non autorizzati
- Evitare usi illeciti di utenti autorizzati
- Permettere 'logging' riferito all'utente

## *Enrollment , autenticazione e autorizzazione*

Il sistema, per concedere o meno l'accesso all'utente, procede spesso in tre fasi distinte:

*Enrollment*

Il sistema deve acquisire i dati dell'utente e il suo profilo (ruoli e autorizzazioni)

*Autenticazione*[♦]

Il sistema deve verificare che l'utente sia effettivamente chi sostiene di essere: ovvero che l'identificativo fornito al sistema nella fase di controllo di accesso corrisponda effettivamente all'utente che richiede il servizio.

*Autorizzazione*

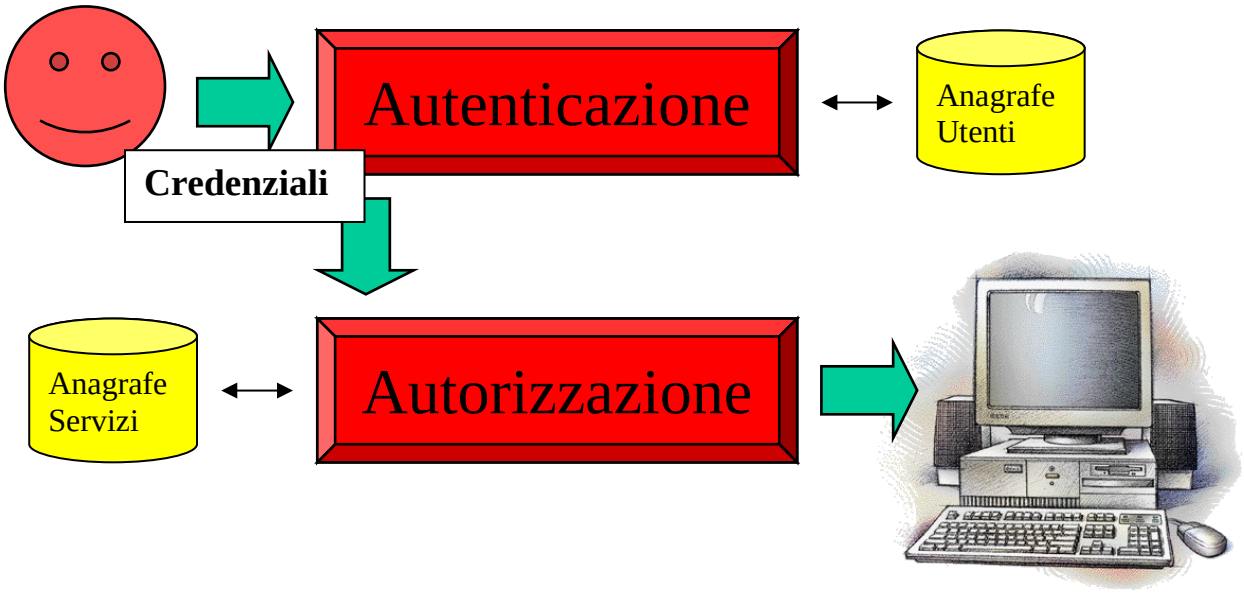
Il sistema, una volta conosciuta l'identità dell'utente, deve verificare che quell'utente sia abilitato o autorizzato a utilizzare il servizio richiesto.[1]

♦ L'autenticazione utente, trattata qui, non deve essere confusa con l'autenticazione di messaggio, che è invece finalizzata a garantire l'integrità e la provenienza di un messaggio trasmesso su una rete altrimenti insicura.

1

L'*autenticazione* verifica se l'utente è chi sostiene di essere, ovvero se corrisponde all'identificativo dichiarato in fase di accesso, richiedendo le menzionate **credenziali di autenticazione**.

L'*autorizzazione* verifica se l'utente, ovvero il suo identificativo, è **abilitato rispetto al servizio** o al sotto-servizio richiesto.



### *Autenticazione e identificazione*

La fase di autenticazione è a volte meglio definita come identificazione. Lo scopo è lo stesso, ma modalità e procedure sono diverse.

- *Autenticazione*
  - *Utente dice chi è (fornisce un identificativo)*
  - *Il sistema verifica*
  - *Il sistema concede o meno l'accesso*
- *Identificazione*
  - *Utente segue una procedura di autenticazione, ma non fornisce alcun identificativo in modo esplicito*
  - *Il sistema concede o meno l'accesso*

Nel seguito, per semplicità, faremo riferimento all'autenticazione utente, che è più diffusa. Le stesse osservazioni però, mutatis mutandis, valgono anche per l'identificazione.

## Autenticazione utente

L’autenticazione di un utente può avvenire in uno dei seguenti modi, o attraverso una combinazione di essi:

- ***Usando qualcosa che l’utente possiede (es. smartcard)***
- ***Usando delle caratteristiche fisiche dell’utente (biometria)***
- ***Usando qualcosa che l’utente sa (es. password, passphrase, pin)***

## Two-factor authentication

L’autenticazione di un utente deve avvenire in piu’ di uno dei precedenti modi (almeno due)

### *Autenticazione con token*

L'autenticazione di un utente può avvenire come detto usando qualcosa che l'utente possiede, come un token USB o una smartcard, o ancora un dispositivo personale, come un palmare o un PC. Questo oggetto viene detto "token di autenticazione".

Vantaggi - e' potenzialmente il metodo più sicuro, in quanto

- Solo con il token e' possibile autenticarsi
- Può essere reso impossibile il "replay" dell'autenticazione
- Limita gli effetti della manipolazione hardware del PC di accesso
- Tamper-resistance in alcuni casi

Svantaggi - principalmente di carattere pratico:

- L'utente perde o dimentica il token, che può anche essere rubato
- Maggiore scomodità rispetto ai sistemi basati su password
- Costi del token e dell'interfaccia hardware

Molto spesso il token di autenticazione viene abbinato a un PIN o a una password (qualcosa che l'utente "sa"), realizzando di fatto una integrazione con quest'altro metodo di autenticazione.

### *Autenticazione biometrica*

L'autenticazione biometrica è poco utilizzata perché:

- E' imprecisa, e di conseguenza poco sicura
  - FAR – false alarm rate
  - IPR – impostor pass rate
- E' poco accettata dall'utente perché
  - Può essere intrusiva
  - Rallenta l'utente nell'accesso
  - Dà una falsa impressione di impossibilità di disconoscere l'accesso
- E' ostacolata dalla normativa sulla privacy
- E' soggetta ad attacchi di "replay"

### *Autenticazione con password*

L'autenticazione con password è la più diffusa.

Vantaggi
- semplice da realizzare e semplice per l'utente
- adatta per applicazioni consumer / Web
- disponibile su tutti i sistemi e le applicazioni

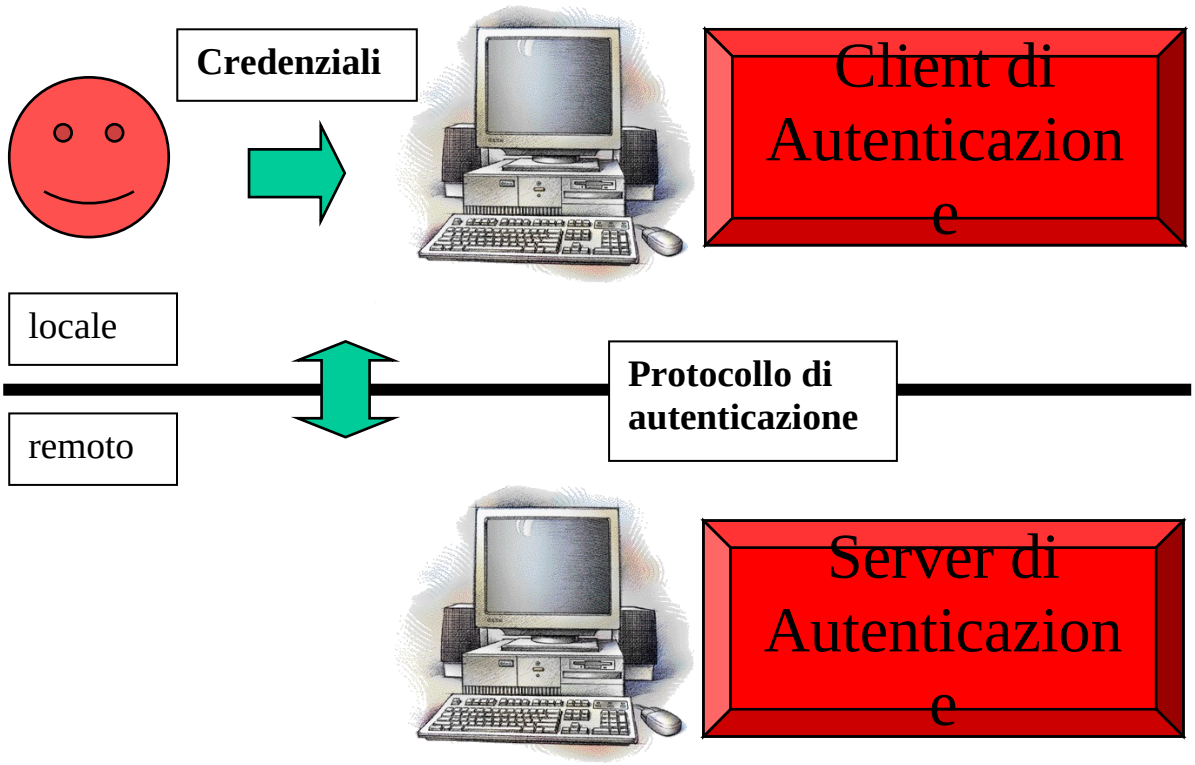
Svantaggi
- sniffing / replay
- guessing / cracking
- phishing e simili

## Protocolli di Autenticazione

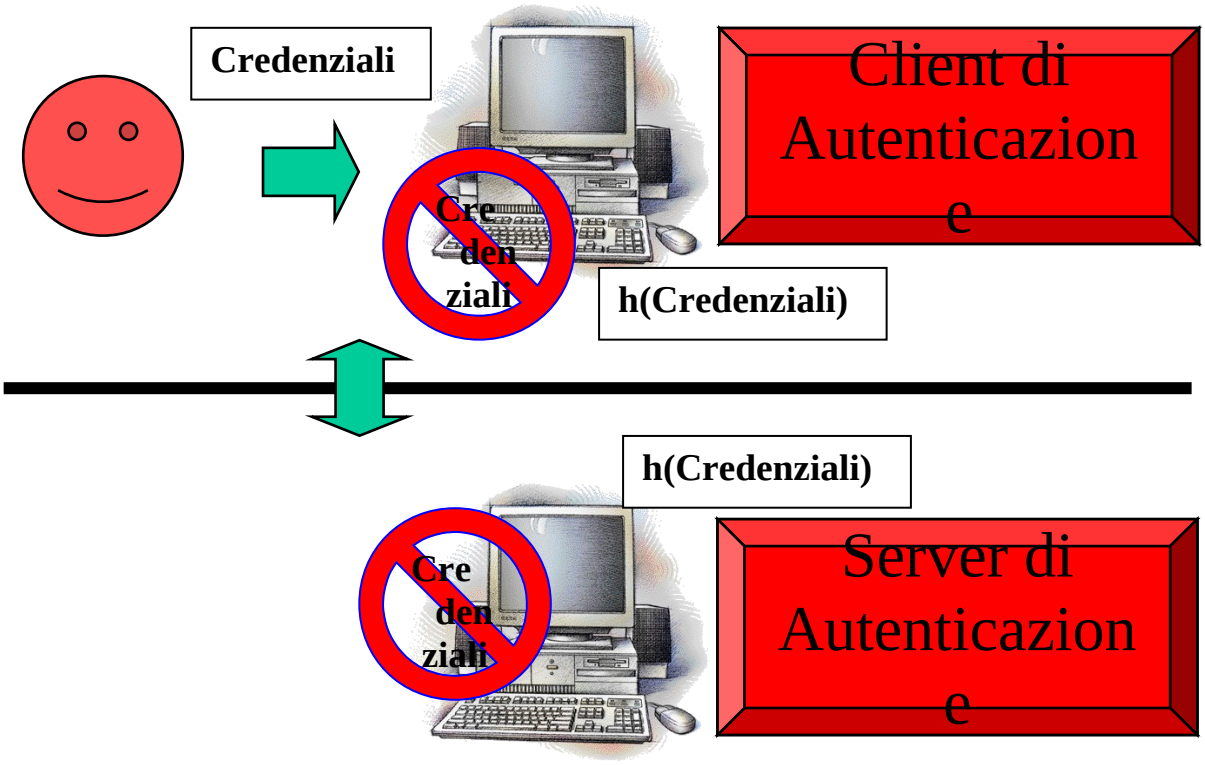
L'utente, per autenticarsi, utilizza normalmente un sistema (client di autenticazione / initiator) remoto rispetto al sistema che verifica la correttezza delle credenziali (server di autenticazione / responder).

Inoltre client e server devono comunicare utilizzando un protocollo di autenticazione, che garantisca la sicurezza del metodo, e in particolare che:

- l'utente venga riconosciuto se e solo se fornisce le credenziali corrette
- le credenziali non possano essere intercettate e utilizzate in un secondo tempo (replay delle credenziali)
- l'accesso prosegua in modo sicuro per tutta la sessione (evitando intercettazioni, modifiche dei messaggi, replay dei messaggi)



Le credenziali di autenticazione **non devono essere memorizzate** in chiaro né sul server né sul client, altrimenti potrebbero essere “rubate” e riutilizzate in un secondo tempo.



## Esempio: UNIX

| **Credenziali = password UNIX di 8 caratteri** | **h(Credenziali) = “encrypted” password di 11 caratteri** |
|---|---|

Come viene calcolata la “encrypted” password:

password (8 caratteri) → K di 64 bit
B = blocco di 64 ‘0’
ripeti 25 volte: cifra (con DES*) B con K → B
da B genera 11 caratteri stampabili → EP
memorizza EP nel file di sistema /etc/passwd

** modificato con il ‘salt’, una informazione diversa per ogni utente, anch’essa memorizzata in /etc/passwd, che influenza il comportamento delle SBOX del DES*

La modifica delle SBOX con il “salt” può indebolire il DES, in quanto a questo punto non più rispondente allo standard.

Tuttavia, l’uso del DES in questo contesto è comunque sicuro, in quanto il testo da cifrare è molto corto (un singolo blocco di 64 bit) e tecniche di analisi statistica non sono utilizzabili.

Debolezze dei cifrari simmetrici
(esempio con sostituzione monoalfabetica):

1) uso di conoscenza di parti di testo in chiaro
j k o o y o k v y p
h e l l o _ _ _ _ _
h e l l o l e _ o _
h e l l o l e r o y

2) analisi statistica
Regolarità nel testo cifrato corrispondo a regolarità nel testo in chiaro
Ad esempio per i cifrari a sostituzione monoalfabetica, la frequenza delle lettere o degli n-grammi nel testo cifrato corrisponde alla frequenza di lettere ed n-grammi nel linguaggio utilizzato nel testo in chiaro.

Entrambe queste tecniche richiedono una significativa lunghezza del testo. Inoltre la password può avere un valore qualunque e non contiene parti di testo fisso.

Nel file “delle password” di Unix (/etc/passwd) vengono memorizzati utenti, sale, e password codificate:

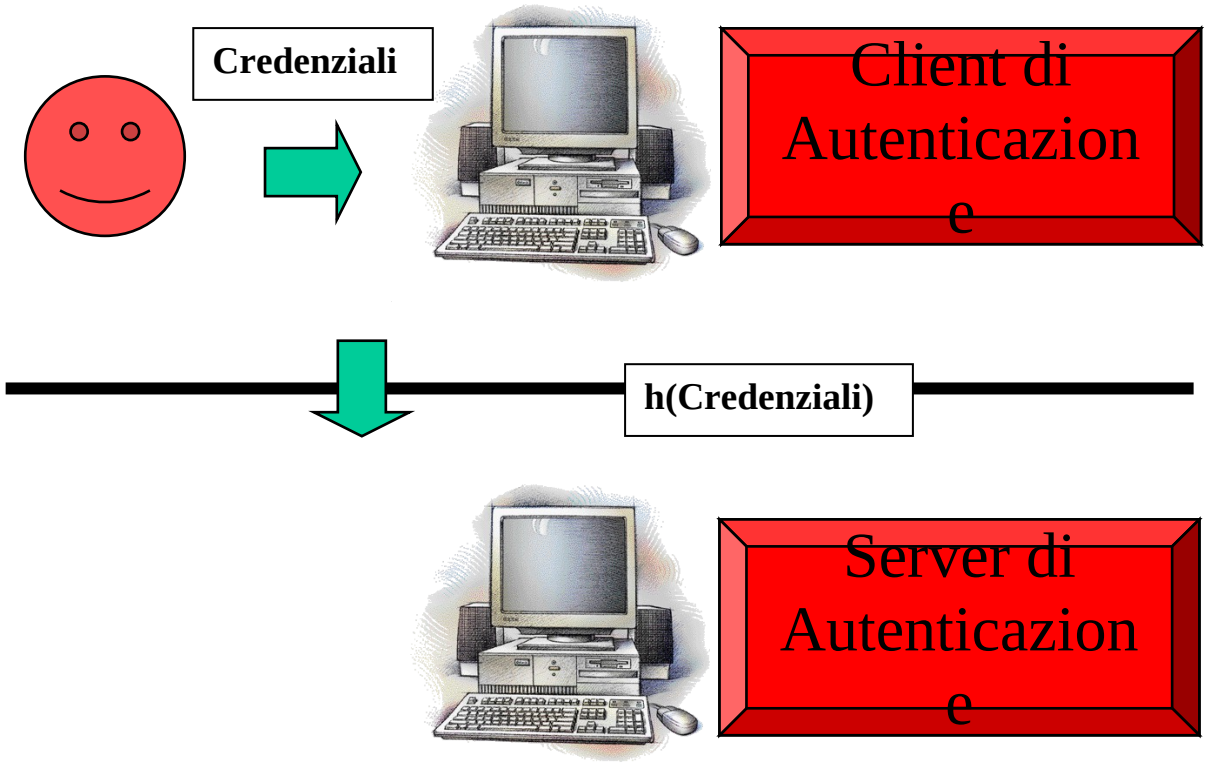
utente1, sale1, h(password1)
utente2, sale2, h(password2)
…
utenten, salen, h(passwordn)

Se qualcuno legge questo file, non riesce a risalire alle password se non provando tutte le password di 8 caratteri, per ogni utente.

Nelle prime versioni Unix, /etc/passwd era leggibile da tutti e con queste informazioni, rendendo possibile, anche se molto dispendioso, un attacco di tipo esaustivo (brute force).

Poi, la componente con le password codificate e’ stata spostata nel file protetto /etc/shadow.

Inoltre le credenziali, anche trasformate (es. h(credenziali)), non devono poter essere intercettate e riutilizzate (replay di autenticazione) in un secondo momento. Il seguente esempio illustra un protocollo di autenticazione debole da questo punto di vista.



## Protocolli di Autenticazione - esempi

Questi protocolli, ampiamente usati nei prodotti e sistemi di mercato, cercano di limitare o eliminare il problema dell'intercettazione e del replay delle credenziali di autenticazione:

- Password con validità limitata nel tempo
- Password generata da token come funzione di data e ora
- Autenticazione con one time passwords
- Needham/Schroeder
- Kerberos
- SSL – Secure Sockets Layer

## Password temporizzate

Il sistema di autenticazione richiede credenziali generate con password, che vengono trasmesse tra client e server di autenticazione, e quindi sono soggette a intercettazione e/o replay.

Le credenziali hanno tuttavia una **durata limitata nel tempo**. Nei sistemi più semplici l'utente è costretto a cambiare password dopo un certo periodo (es. tre mesi). Nei sistemi più sofisticati la password viene generata da un token in base a un segreto condiviso con il lato server e in funzione di data e ora. E' quindi possibile cambiare password molto più spesso (es. ogni minuto).

Tutti questi sistemi hanno un **livello di sicurezza basso**, in quanto nell'intervallo di tempo in cui la password è valida sono possibili gli stessi attacchi basati su intercettazione e replay possibili con le credenziali di durata illimitata. Hanno tuttavia avuto un notevole successo di mercato.

## One time password (OTP)

Come dice il nome in questo caso le credenziali vengono utilizzate una sola volta e quindi l'intercettazione è inutile – il sistema è sicuro.

P = password iniziale, scelta dall'utente
$P_1 = h(P)$
$P_2 = h(P1) = h(h(P)) = h^2(P)$
...
$P_n = h^n(P)$

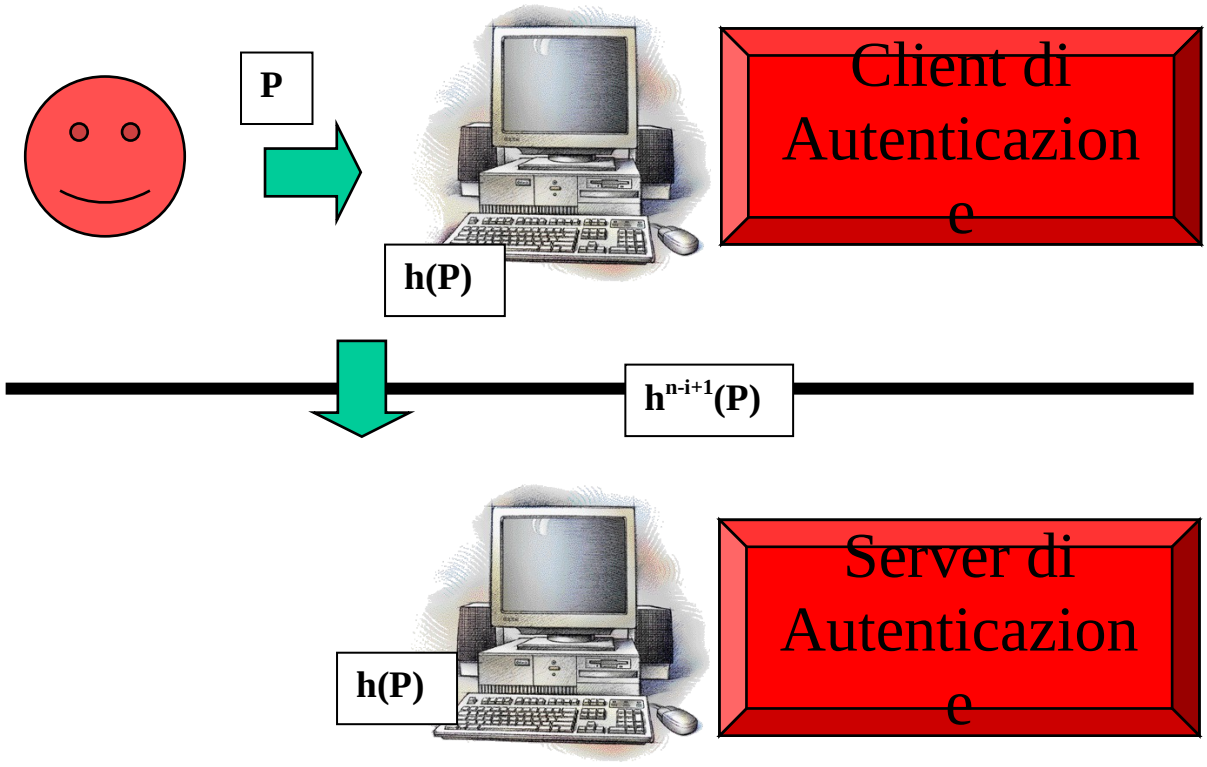
Uso

**$P_n$ alla prima sessione,**
**$P_{n-1}$ alla seconda sessione,**
**...,**
**$P_1$ per la n-esima sessione e**
**P alla (n+1)-esima e ultima sessione.**

A questo punto occorre reinizializzare il sistema. Usando una funzione di hash h non invertibile non sarà possibile usare una password intercettata per calcolare la successiva.

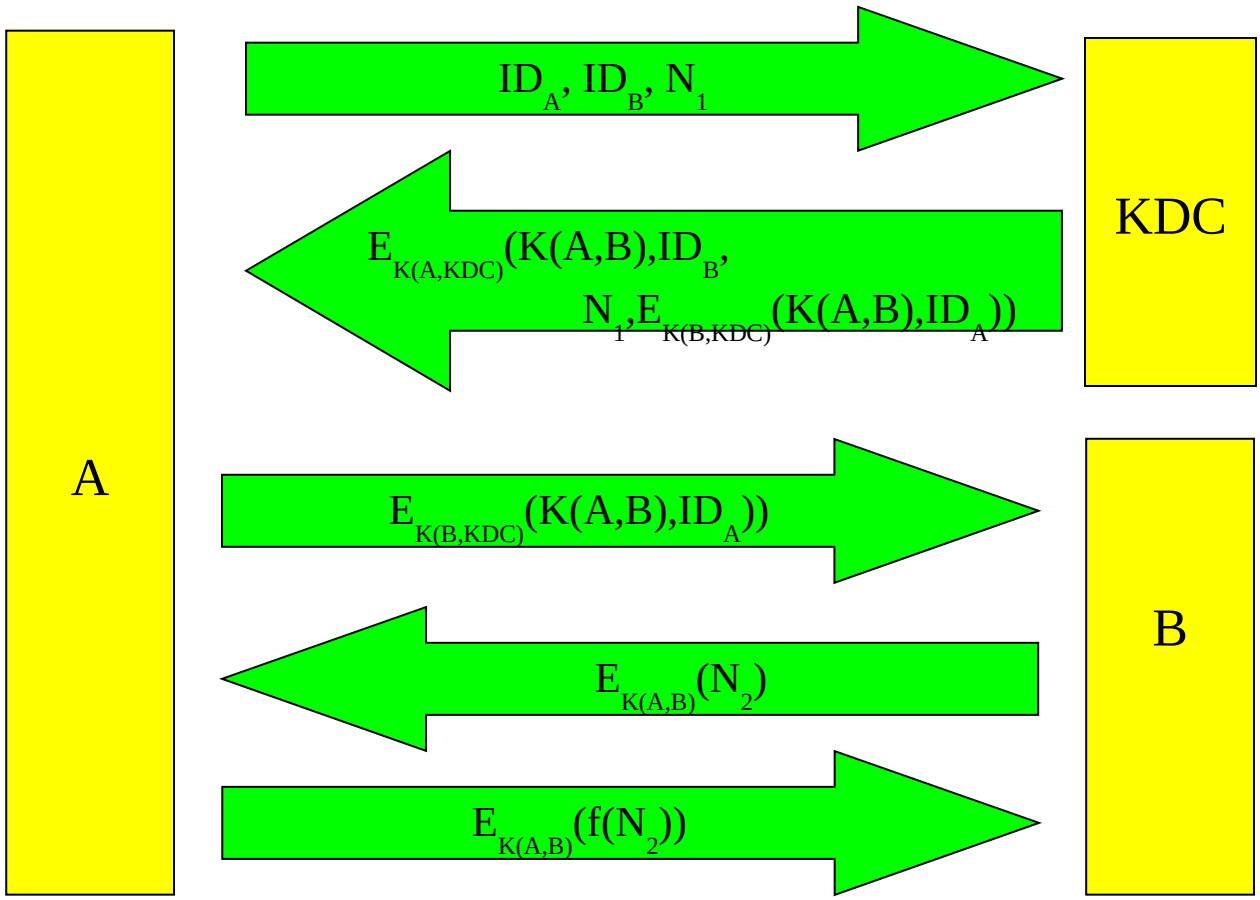
## One-time passwords – sessione i-esima

- Il Client, conoscendo h(P), può calcolare $h^{n-i+1}(P)$
- Il Server, conoscendo h(P), può verificare la correttezza di $h^{n-i+1}(P)$
- Chi intercetta $h^{n-i+1}(P)$
  non può calcolare la successiva password $h^{n-i}(P)$



### *Protocollo di autenticazione di Needham/Schroeder*

Il protocollo permette di condividere una chiave simmetrica K(A,B) tra due soggetti A, B che già condividono rispettive chiavi simmetriche K(A,KDC) e K(B,KDC) con una terza parte KDC (Key Distribution Center).



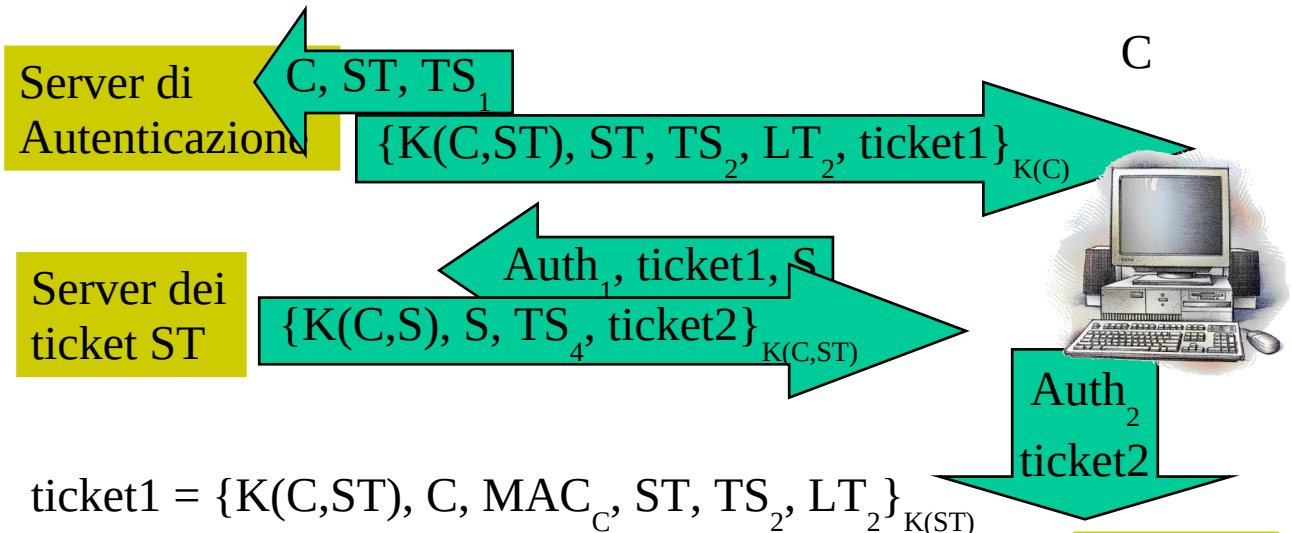

I valori N1 ed N2, detti “nonce” servono per evitare attacchi di tipo “replay” – in particolare i messaggi 4 e 5 servono a B per verificare che A possieda effettivamente la chiave K(A,B).

***

Si segua ora cap. 7.3 da pag. 214 a 219 (symm key distribution)
Poi 10.1 per public key distribution, quindi 13.1 e 13.2
Poi Kerberos, X.509 e PKI
Poi posta elettronica sicura con PGP, S/MIME, PEC
Infine SSL, Web security e pacchetto OpenSSL per laboratorio

## *Kerberos (versione 4)*

- Controllo di accesso centralizzato
- Applicabile in una LAN di grandi dimensioni
- Basato su crittografia convenzionale
- Utilizzato in Microsoft Active Directory




$ticket1 = \{K(C,ST), C, MAC_C, ST, TS_2, LT_2\}_{K(ST)}$

$Auth_1 = \{C, MAC_C, TS_3\}_{K(C,ST)}$

$ticket2 = \{K(C,S), C, MAC_C, S, TS_4, LT_4\}_{K(S)}$

$Auth_2 = \{C, MAC_C, TS_5\}_{K(C,S)}$

## *SSL*

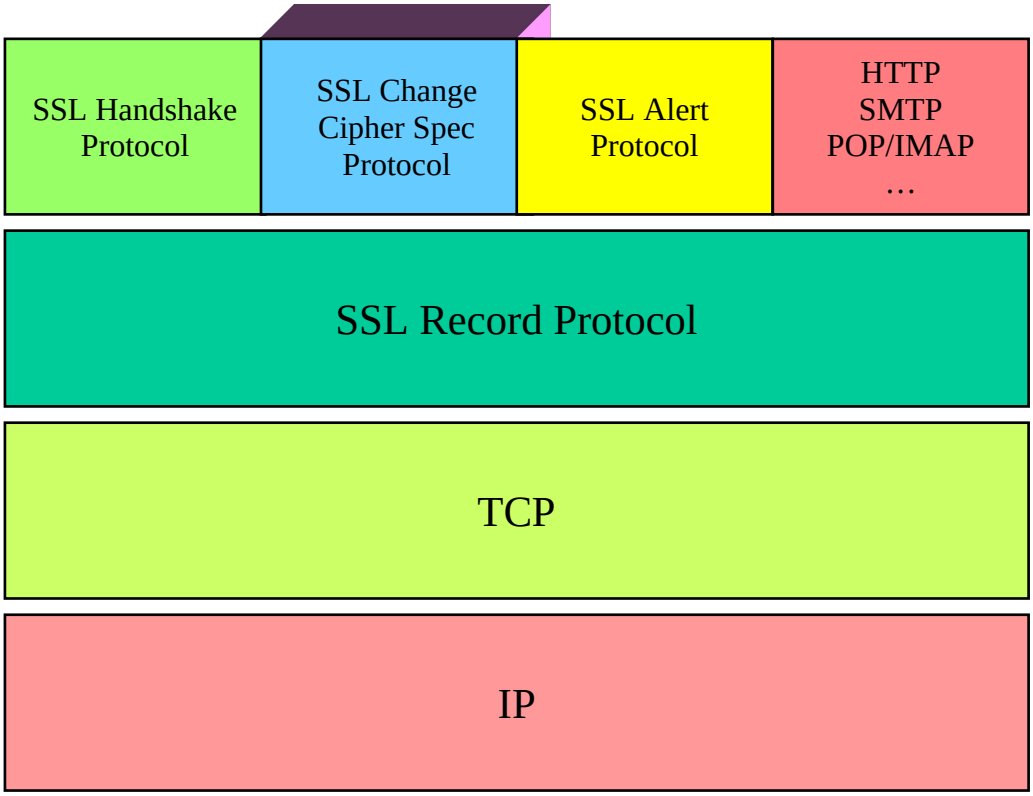
### *USO e SCOPI*

- Applicabile in WAN / Internet
- Basato su crittografia asimmetrica
- Integrato in Browser e MUA

### *STORIA E STANDARD*

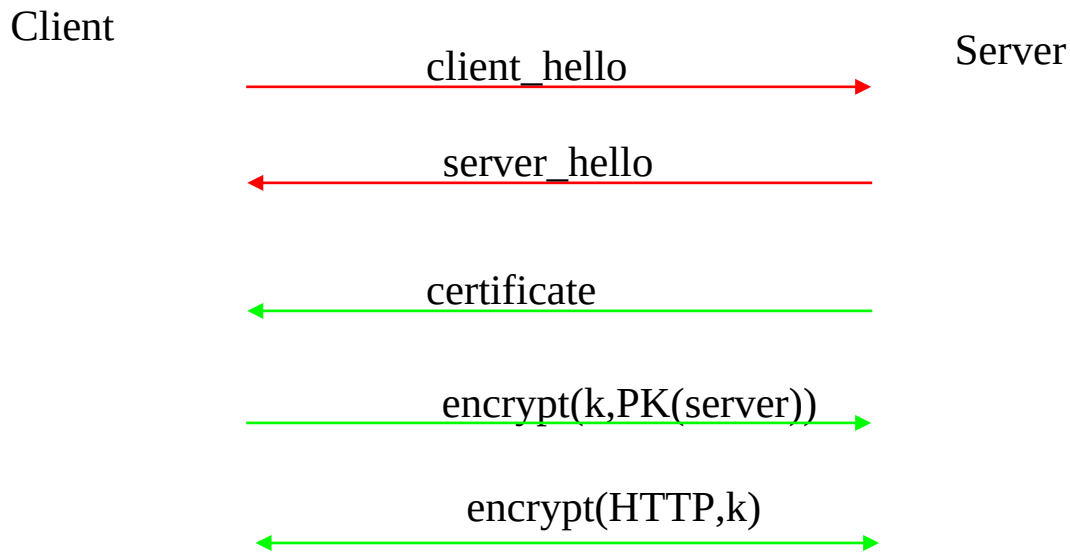
- SSL è stato proposto da Netscape
- SSLv3 ha avuto contributi anche da altre industrie
- È stato inizialmente proposto come un Internet Draft
- L'IETF lavora allo sviluppo di un Internet Standard (attualmente RFC) che può essere visto come SSLv3.1

# PILA DI PROTOCOLLI USATI CON SSL

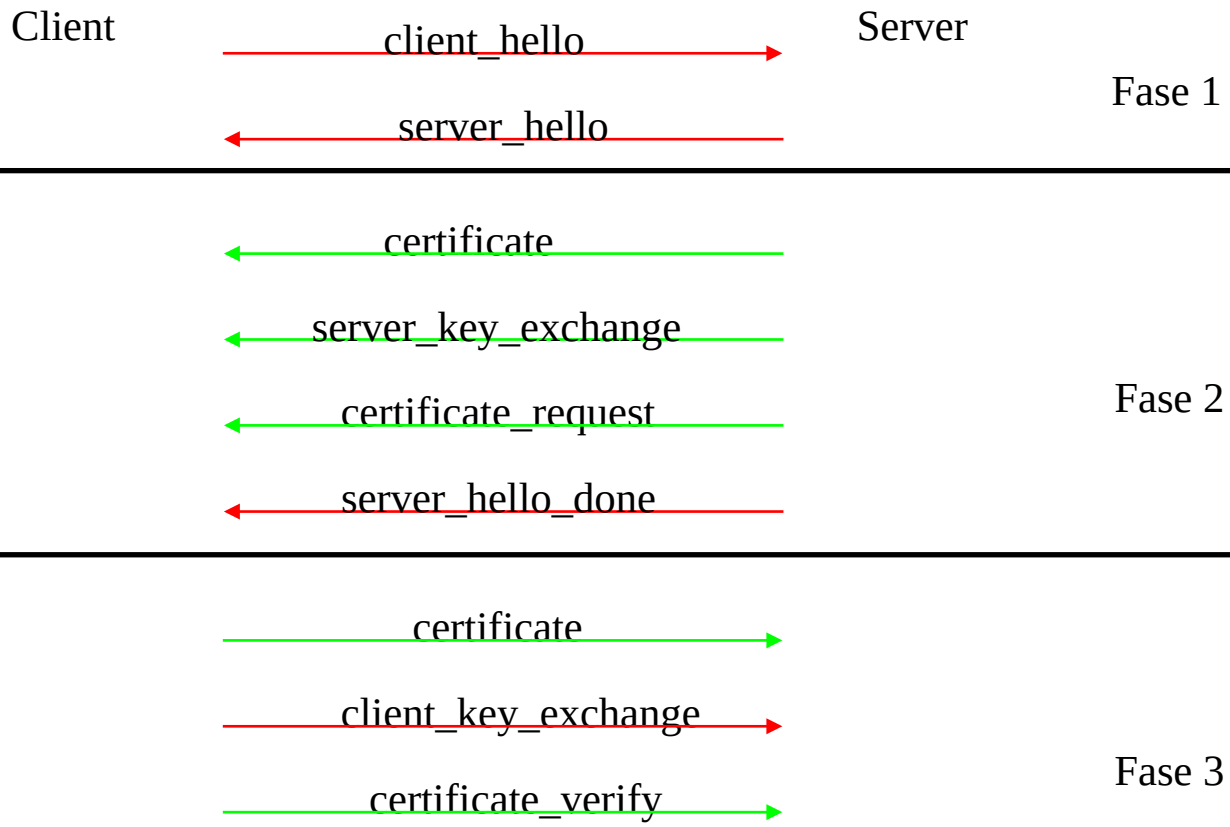


SSL Handshake Protocol

Versione semplificata (senza client authentication):



Versione completa (con client authentication):




Nella versione semplificata, la client authentication è sostituita da autenticazione con password su canale sicuro, accettabile in molte applicazioni e ampiamente utilizzata su Internet (es. home banking)

## Autorizzazione e identity management

- Integrazione con sistema operativo e DNS
  - Es. Active Directory utilizza i record SRV del DNS per rendere reperibili ai client i propri servizi di autenticazione, e sostituisce il precedente schema di autenticazione in rete con i domain controller
- Directory
  - Convergenza sullo standard aperto LDAP: qui vengono memorizzate informazioni sull'utente, informazioni relative all'autenticazione (derivati da credenziali di accesso), all'autorizzazioni (diritti di accesso per specifiche applicazioni), e certificati digitali.
- Meta-directory
  - Necessità di integrazione di diversi directory, aperti e proprietari, realizzati da vendor diversi, raccogliendo i dati in una nuova directory
- Virtual Directory
  - Integrazione di diverse directory, offrendo un front-end unico a dati comunque mantenuti sui directory nativi
- Provisioning
  - Necessità di propagare la definizione degli utenti con le loro autorizzazioni ai sistemi e agli applicativi finali
- Delegation
  - Necessità di far gestire i diritti di accesso a terze parti, appunto delegate per particolari sottodomini e applicazioni
  - Necessità di delegare all'utente stesso alcuni compiti (es. cambio password)
- Identità federate
  - Necessità di gestire autenticazione e autorizzazione attraverso domini e organizzazioni diverse