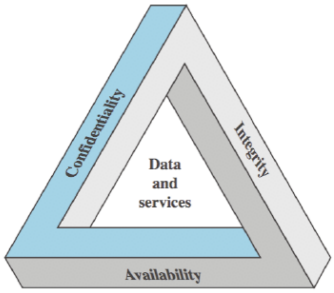
Protezione Delle Risorse

D: Quali risorse (*Asset*) vogliamo proteggere?

- **Hardware**: sicurezza "fisica"
- **Software**: sistema operativo e applicativi
- **Dati**: file e database
- **Reti**: collegamenti e apparati di rete



"**Proteggere**" vuol dire garantire le proprietà di *confidenzialità, integrità, disponibilità, autenticità e tracciabilità*.

Confidenzialità:
Nessun utente deve poter ottenere o dedurre dal sistema informazioni che non è tenuto a conoscere.

Integrità:
Bisogna impedire l'alterazione diretta o indiretta delle informazioni, sia dalla parte degli utenti, sia dalla parte dei processi non autorizzati, che a seguito di eventi accidentali.
(Se i dati vengono alterati è necessario fornire uno strumento per verificarlo)

Disponibilità:
Rendere disponibile a ciascun utente le informazioni alle quali ha diritto di accedere, nei tempi e nei modi previsti (include prestazione e sicurezza)

Autenticità:
Ciascun utente deve poter verificare l'autenticità di un'informazione e si può richiedere di verificare se un'informazione è stata manipolata.

Tracciabilità:
Ogni azione delle entità devono essere tracciate in un modo univoco in modo tale da supportare la non-ripudiabilità e l'isolamento delle responsabilità

	Confidenzialità	Integrità	Disponibilità
HW			Calcolatore rubato
SW	Copia non autorizzata	Eseguibile modificato	Eseguibili cancellati
Dati	Lettura non autorizzata	File modificati	File cancellati
Rete	Lettura messaggi inviati	Messaggi modificati / ritardati / duplicati	Messaggi distrutti Rete fuori uso

Minaccie e Attacchi

→ Una ***minaccia*** è una possibile violazione della sicurezza
→ Un ***attacco*** è una violazione effettiva del sistema

Gli attacchi possono essere di tipo:

Attivi: Tentativi di alterare le risorse o modificare il funzionamento dei sistemi
Passivi: Tentativi di capire le risorse e utilizzarle senza intaccare i sistemi e le risorse
Interni: Iniziati da un'entità all'interno del sistema
Esterni: Iniziati da un'entità esterna, tipicamente attraverso la rete

Classi di attacco:

Disclosure: Accesso non autorizzato alle informazioni
Deception: Accettazione di falsi dati
Disruption: Interruzione o prevenzione di operazioni corrette
Usurpation: Controllo non autorizzato di alcune parti del sistema

Principi Fondamentali

Aspetti Economici: la progettazione delle misure di sicurezza deve essere il più semplice possibile

Fail-Safe Default: i comportamenti non specificati devono prevedere un caso di default sicuro

Sorgente Pubblico: è preferito a un sorgente privato

Tracciabilità Delle Operazioni: qualsiasi operazione può essere riconosciuta in un sistema ripristinato

Seprazione dei Privilegi: bisogna differenziare gli accessi alle risorse e ai file

Separazione delle Funzionalità: distinzione dei ruoli nel sistema fisico e logico

Isolamento dei Sottosistemi: un sistema compromesso non ne deve compromettere altri

Una **politica di sicurezza** è un'indicazione di cosa è e cosa non è permesso. Le regole possono riguardare dati, utenti e operazioni possibili.

Un **meccanismo di sicurezza** è un metodo (procedura) per garantire una politica di sicurezza.

→ Data una politica, i meccanismi possono **prevenire, recuperare o scoprire** un attacco.

Crittografia

La crittografia è una scienza che si occupa di proteggere l'informazione rendendola sicura, in modo che se un utente ne entri in possesso non sia in grado di comprenderla.

Algoritmo Crittografico:

Un algoritmo crittografico è una funzione che prende in ingresso un messaggio e come parametro una **chiave**, e produce in uscita un messaggio trasformato.

- Cifratura: *plaintext --> ciphertext*
- Decifratura: *ciphertext --> plaintext*

→ Le chiavi possono essere **uguali**, quindi l'algoritmo è **simmetrico** (le chiavi devono essere segrete)
→ Le chiavi possono essere **diverse**, quindi l'algoritmo è **asimmetrico** (una chiave è pubblica e una è privata)

Robustezza Crittografica:

→ **NON** deve essere possibile dato un testo cifrato risalire (facilmente) al testo in chiaro
→ **NON** deve essere possibile dato un testo cifrato e decifrato, risalire alla chiave

In generale, nessun algoritmo crittografico è completamente sicuro, si dice quindi che un **algoritmo crittografico è computazionalmente sicuro se:**

- Il costo per decifrarlo è superiore al valore dell'informazione cifrata
- Il tempo di decifratura è superiore al tempo di vita dell'informazione cifrata

Crittoanalisi:

La crittoanalisi tenta di decifrare un testo senza conoscere la chiave di decifratura. L'attacco più banale è il *Brute-Force*.

Principio di Kerckoffs : la segretezza deve essere nella chiave e non nell'algoritmo!

Crittografia Simmetrica

La **crittografia simmetrica**, detta anche crittografia a chiave segreta, utilizza una chiave comune e il medesimo algoritmo crittografico per cifrare e decifrare la risorsa.

Due utenti che desiderano comunicare con la crittografia simmetrica devono concordarsi su che algoritmo e chiave usare (*la chiave deve essere scambiata su un canale sicuro*)

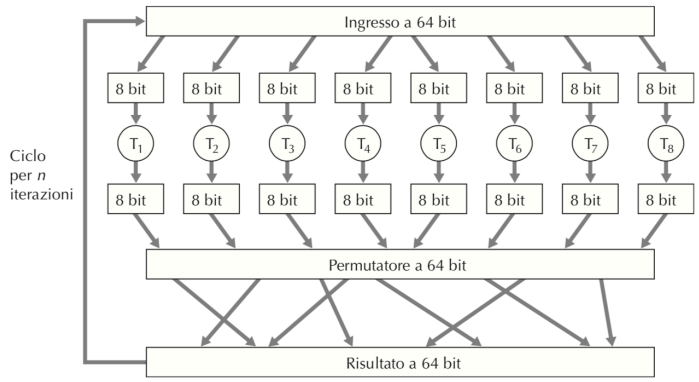
Alcuni Algoritmi Simmetrici:

- **Cifrario di Cesare**
- **Cifratura Monoalfabetica**

- **Cifrari a Blocchi:**

Sono usati in molti protocolli sicuri di Internet, compreso PGP (posta elettronica), SSL (connessione TCP) e IPSec (trasmissione a livello di rete). Sono anche chiamati cifrari a flusso.

→ Dati K bit, 2^K ingressi vengono permutati
→ Le permutazioni possono essere combinate per creare schemi più complessi



DES (*Data Encription Standard*) [56 bit] :

DES è il più noto algoritmo crittografico simmetrico. Utilizza chiavi a 56 bit ed è ormai considerato obsoleto.

TRIPLO-DES [112 – 168 bit] :

Per aumentare la sicurezza del DES, si triplica in cascata l'algoritmo con chiavi diverse. Esistono due varianti:

- Con chiave da 112 (*x 2*)
- Con chiave da 168 (*x 3*)

AES (*Advanced Encryption Standard*) [128 – 192 – 256]

Negli **algoritmi simmetrici** la chiave è la stessa in cifratura e decifratura. Deve quindi essere segreta e deve essere scambiata su un canale sicuro. Per questo **Diffie** e **Hellman** propongono uno schema che supera questa limitazione: crittografia a chiave pubblica (asimmetrica)

Crittografia Asimmetrica

Nella crittografia **asimmetrica** ogni utente ha una chiave pubblica e una privata: la chiave pubblica viene resa nota, mentre quella privata deve rimanere segreta.

> → La risorsa viene cifrata con la **chiave pubblica** del destinatario, il quale dovrà usare la propria **chiave privata** per decifrarlo

Vantaggi:

→ Non è più necessario incontrarsi per scambiare le chiavi
→ La stessa chiave pubblica può essere usata da più utenti

Requisiti:

→ La generazione della chiave privata e pubblica deve essere semplice
→ L'operazione cifratura e decifratura deve essere semplice quando è nota la chiave
→ Deve essere computazionalmente impossibile ricavare la chiave privata da quella pubblica
→ Deve essere computazionalmente impossibile ricavare il testo in chiaro avendo il testo cifrato e la chiave pubblica

Algoritmo RSA

RSA è un algoritmo che si basa sulla difficoltà di **scomporre un numero in fattori primi**. La chiave ha di solito dimensioni di 2^10 bit (*300 cifre decimali*).

→ Il bruteforce su RSA non prova tutte le chiavi possibili, ma prova solo a fattorizzare il prodotto di numeri primi.

Come funziona:

Scelti due numeri primi `p, q` si calcola:

- `n = p*q`
- `z = (p-1)*(q-1)`
- `un numero 1 < e < n relativamente primo a z`
- `un numero d tale che (e*d-1) sia multiplo di z`

[…]

Sintesi Algoritmi Asimmetrici:

- Richiedono molte risorse computazionali (dalle 100 alle 1000 volte più lenti dei simmetrici)
- Vengono utilizzati per scambiare le chiavi di sessioni (simmetrici)
- Con RSA ciò che viene cifrato con la chiave pubblica può essere decifrato con la chiave privata ma anche l'incontrario! In questo modo si può garantire l'autenticità

Integrità E Autenticazione

L'obbiettivo crittografia è garantire la privacy

L'integrità garantisce che un messaggio non sia stato modificato da terzi. Per fare ciò ci appoggiamo alle funzioni HASH.

Funzioni HASH:

Le funzioni HASH permettono di risolvere il problema di autenticità e integrità del messaggio. Se un utente mi invia un messaggio come faccio a sapere che è suo?

Una **funzione hash** trasforma un qualsiasi messaggio in una lunghezza predefinita (*digest*)

Per stabilire la sicurezza, le condizioni gli algoritmi che eseguono hashing devono avere:
- **Coerenti**: Input uguali devono corrispondere ovviamente ad output uguali
- **Casuali** (o apparire tali): bisogna impedire l'interpretazione del messaggio originale
- **Univoci**: due messaggi non dovrebbero generare lo stesso *digest*
- **Non Invertibili**: non deve essere possibile risalire al messaggio originale

Gli HASH **non invertibili** vengono di solito utilizzati per assegnare un'**impronta digitale** alle risorse.

Le funzioni HASH più comuni sono MD5 e SHA. *E' possibile offrire Autenticità senza scambiare la chiave?* Si grazie agli algoritmi asimmetrici: **firma digitale**

Autenticità: Firma Digitale

E' l'equivalente informatico di una firma convenzionale, generalmente non ripudiabile

Nella crittografia asimmetrica on si cifra l'intero messaggio con gli algoritmi asimmetrici, ma si usano per scambiare la chiave di sessione (di un alg. simmetrico) e poi utilizzo la chiave appena creata per andare a cifrare il messaggio. Questo però non permette di confermare l'autenticità. Per essere sicuri che il mittente sia quello pensato si utilizza la firma digitale.

Si utilizza la cifratura asimmetrica nel senso inverso:

Ad esempio, in RSA viene utilizzato nel seguente modo:
- L'algoritmo di cifratura diventa l'algoritmo di verifica
- L'algoritmo di decifratura diventa l'algoritmo di firma

→ Firmare l'intero documento diventa oneroso, quindi si firma l'HASH del documento

Non è garantito che l'utente che si dichiara bob, sia realmente bob. Per risolvere si utilizza il **certificato elettronico**: associa la chiave pubblica e privata univocamente a un'identità. Il certificato viene rilasciato dalla **CA** (*Certification Authority*)

Autenticazione

E' un processo che avviene tra due interlocutori

Gli interlocutori possono essere utenti o computer dove la proprietà primaria è la richiesta di un **corretto controllo di accesso**.

Tipologie di autenticazione:

- **Locale**: l'utente accede in locale al servizio che effettua l'autenticazione. (*i.e. acceso allo smartphone)*

- **Diretta**: l'utente accede da remoto al servizio che effettua l'autenticazione. (*i.e. accesso alle macchine del laboratorio Delta*)

- **Indiretta**: il servizio di autenticazione è separato

- **Offline**: i certificati elettronici

Fattori di autenticazione:

- **Conosce** (*PIN, password, …*)
- **Possiede** (*carta, tessera, …*)
- **E'** (*impronta digitale, iride, …*)

#Autenticazione: Qualcosa che si conosce: **username** e **password**

Vantaggi	Svantaggi
- Semplice - Economico - Non richiede di salvare dati (client)	- Password deboli scelte dall'utente - Metodi di autenticazione deboli

Gli attacchi possibile alle password possono essere:
- Intercettazione (*MITM/Sniff*)
- Craking (*Brute Force*)
- *Social Engeniring, Trojan, Keylogger, …*

#Autenticazione: Qualcosa che si possiede: **One Time Password**

One Time Password è un sistema in cui viene generata una password ogni volta per evitare il problema dell'intercettazione. Le password "monouso" vengono generate seguendo un contatore o l'istante temporale.

Altri esempi:
- Smart Card
- Carte Magnetiche

Spesso queste si combinano con dei PIN.

#Autenticazione: Qualcosa che si è:

E' il possesso di caratteristiche univoche che forniscono l'autenticità:

- **Fisiche**: impronta digitale, iride, …
- **Comportamentali**: firma, timbro della voce, scrittura, …

Punto debole:

- Misure imprecise basate su dei template
- Possibilità di falsi positivi e falsi negativi

Autenticazione **DIRETTA**:

Le modalità di autenticazione viste fin ora si possono usare in **locale**, se l'autenticazione viene da **remoto** (*autenticazione diretta*) ci possono essere altri problemi: un intruso potrebbe intercettare e registrare le informazioni (*MITM*)

Soluzione: l'autenticazione viene su un canale sicuro, oppure si aggiunge una "*sfida"*.

Autenticazione **INDIRETTA**:

Le informazioni degli utenti vengono salvati su in sistema unico, e viene offerto anche ad altri servizi per autenticare. Due esempi di autenticazione indiretta sono:

- **RADIUS** (*Remote Authentication Dial in User Service*) nato per accesso remote dial-up
- **Kerberos** utilizzato per l'autenticazione *Single Sign-On (SSO)* all'interno di un dominio amministrativo.

Autenticazione **OFF-LINE**:

E' basata sui certificati: emessi da un'**autorità di certificazione** (***CA***) e distribuiti da un'infrastruttura a chiave pubblica (***PKI***).

Autorità di Certificazione (CA)

I 10 compiti di una CA

1) Identificare con certezza la persona fisica che fa richiesta della chiave pubblica
2) Rilasciare e rendere pubblico il certificato
3) Garantire l'accesso telematico al registro delle chiavi pubbliche
4) Informare i richiedenti sulle tecniche di accesso e di registrazione
5) Dichiarare la propria politica di sicurezza
6) Attenersi alle norme sul trattamento dei dati personali
7) Non rendersi depositario delle chiavi private
8) Procedere alla revoca in caso di problemi con l'intestatario
9) Rendere pubblica la revoca e sospensione delle chiavi
10) Assicurare una corretta manutenzione delle chiavi pubbliche

Come ottenere un certificato digitale:

1) L'utente genera una **coppia di chiavi**
2) L'utente **invia alla *CA* una richiesta di certificato** insieme alla chiave generata (a meno che non sia la *CA* che ha generato le chiavi)
3) **La *CA* autentica l'utente** chiedendoli di recarsi allo sportello *LVP* (*Local Validation Point*) collegato con la *CA*
4) Quando è verificata l'autenticità **la *CA* emette il certificato** e lo invia al richiedente tramite posta elettronica e inserisce la chiave nel registro delle chiavi pubbliche

L'intera procedura accade nell'ambito di una **PKI (*Public Key Infrastructure*)**

PKI

Public Key Infrastructure

La struttura mini ma è ***CA+LVP***. Si può avere una struttura gerarchica di chiavi pubbliche:

- E' una struttura ad albero
- La root certifica le CA di primo livello
- Le primo livello certificano le CA di secondo livello
- Le CA di ultimo livello certificano l'utente

Utilizzo dei certificati:

- Bob invia ad Alice il certificato firmato dalla CA
- Alice controlla la firma ed estrapola la chiave pubblica di Bob
- Alice ha la chiave pubblica di bob garantita dalla CA

Problemi dei Certificati:

- E' necessario ottenere in modo sicuro il certificato dalla CA per controllare la firma
- Un certificato può essere revocato, ma il match della firma può avere comunque successo
- Il sistema implica una fiducia nella CA, ma chi lo garantisce?

Problemi comuni a tutti gli schemi:

Alice vuole inviare una coppia di chiavi pubblica e privata, ma:
- Come la **genera**?
- Come la **conserva**?
- Come la **trasporta**?

Autorizzazione: Controllo degli Accessi

Garantisce che il servizio sia limitato a solo a chi ne ha diritto

- Le **politiche** di accesso definiscono l'attribuzione dei privilegi sugli oggetti
- I **meccanismi** di accesso definiscono come le relazioni tra i soggetti e gli oggetti sono rappresentati

Due principi utili:
- **Privilegio minimo**: Ad un soggetto dovrebbero essere concessi solo i privilegi minimi necessari a compiere l'azione che deve compiere
- **Separazione dei compiti**: Nessun soggetto dovrebbe avere abbastanza potere per sovvertire il sistema

ACL (Acces Control List):

Sono adatte in un contesto in cui la protezione è **orientata ai dati**: è semplice gestire gli accessi a un oggetto

Politiche di controllo dell'accesso:

Si distinguono in:
- **DAC**: Discretionary Access Control
- **MAC**: Mandatory Acccess Control

Gli accessi possono essere combinati con una suddivisione di utenti e ruoli:
- Un gruppo è una lista di soggetti
- Un ruolo è un insieme prefissato di permessi di accesso

DAC

In un modello *DAC* gli utenti possono a loro discrezione concedere e revocare i permessi sugli oggetti sotto il loro controllo
- Ogni oggetto ha un proprietario e il soggetto ne definisce i permessi di accesso
- Il DAC è flessibile e usato in diversi contesti
- Il DAC non permette di controllare la diffusione delle informazioni

MAC

In un modello *MAC* la politica di accesso è determinata centralmente dal sistema, utilizzato ad esempio in ambito militare

i.e. nel modello Bell-LaPadula, il sistema segue le seguenti regole:

- **No read-up**: non è possibile leggere informazioni dei livelli più alti rispetto al proprio
- **No write-down**: non è possibile scrivere informazioni nei livelli più bassi al proprio

Il MAC è meno flessibile, ma più robusto del DAC.

Firewall

I **firewall** di rete sono apparecchiature o sistemi che controllano il flusso del traffico tra due reti con diversi livelli di sicurezza:

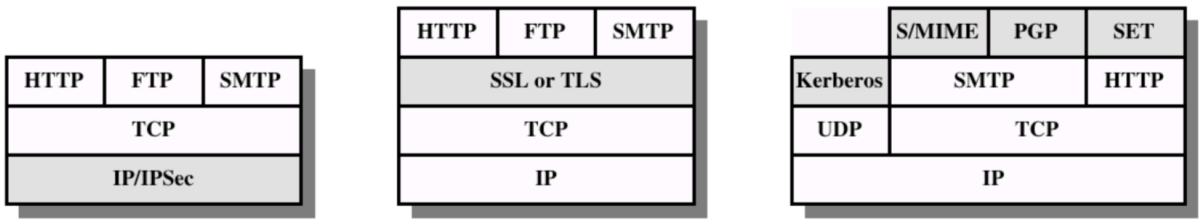
- Per prevenire accessi non autorizzati a una rete privata
- Per prevenire l'esportazione di dati dall'interno all'esterno
- Schermare alcune reti interne da nascondere ad altri
- Bloccare alcuni accessi e/o servizi a degli utenti
- Monitorare (e logging)

Problemi del firewall:

- Gli attacchi possono arrivare dall'esterno ma anche dall'interno di una rete
- Non difende da bug non aggiornati nei protocolli
- I filtri sono difficili da impostare e mantenere, sono un difficile compromesso tra libertà e sicurezza
- Può degradare le performance della rete

Default Deny	**Default Permit**
Tutto quello che non è espressamente ammesso è proibito: - I servizi sono abilitati caso per caso dopo un'analisi - Gli utenti sono molto ristretti	Tutto quello che non è espressamente proibito è ammesso: - Il sys admin deve procedere velocemente ogni volta che appare un bug nel protocollo - Vengono aggiunti/rimossi servizi quando vengono scoperti pericolosi - Gli utenti sono meno ristretti

Scurezza nello stack protocollare TCP/IP



A livello applicativo:

Ogni applicazione implementa già all'interno della sua logica la sicurezza. I messaggi di livello applicativo che vengono generati sono già cifrati. Il TCP quindi quando riceve i dati, non gli interessa la semantica (significato) e non ne capisce il contenuto perché è già cifrato.

Chi sviluppa l'applicazione si prende in carico di sviluppare anche la parte legata alla cifratura (può essere oneroso e inutile inglobare un algoritmo di cifratura già essitente in ogni applicazione differente)

A livello di trasporto:

A livello di trasporto il messaggio è già cifrato con SSL o TSL, quindi TCP non vede il contenuto (e non gli interessa), ma questo strato è utilizzato da più di un'applicazione per rendere più sicura la connessione (i.e. HTTPS); quindi il protocollo a liv. Applicativo non cambia, è uno in più strato che cifra i dati in mezzo.

A livello di rete:

Fa la cifratura avviene direttamente a livello del pacchetto IP. S la cifratura la faccio a livello di trasporto, significa che quando viaggia sulla rete, oltre all'*header IP* chi intercetta il pacchetto riesce a leggere anche parte del pacchetto del contenuto *payload* del pacchetto IP e quindi vede parte del pacchetto TCP.
Per ovviare a ciò si cifra a livello IP (ci sarà l'*header IP* perché è necessario), ma c'è subito dopo un secondo header che è dell'*IPSEC* che contiene le informazioni utilizzate per la cifratura e da li in poi è cifrato (es quindi non vedo le porte usate a liv. di trasporto)

A livello di collegamento (i.e. WLAN):

Viene cifrato il pacchetto usato nelle Wireless Lan, si cifra solo il tratto condiviso dall'host all'AP, (si fa solo in casi particolari come questo).. si possono anche combinare le cifrature dei liv. Superiori.

Sicurezza delle e-Mail

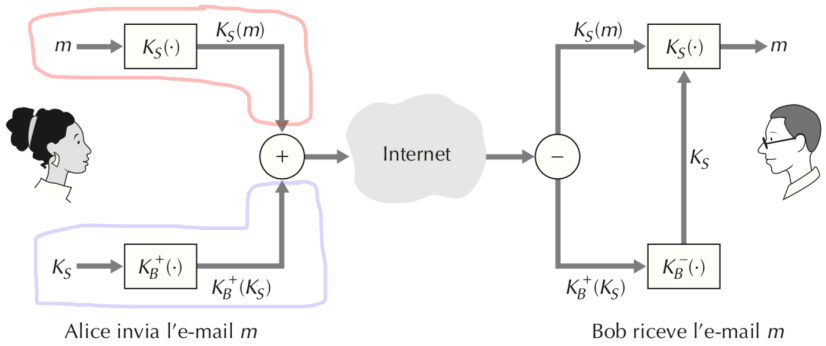
Livello Applicativo

Perché ci interessa?
Perché la cifratura può garantire l'autenticazione del mittente, del destinatario, farla leggere solo al destinatario o garantirne l'integrità. Per poter ottenere queste proprietà si applicano i meccanismi seguenti:

Esempio Cifratura Delle Email Garantendo Le (4) Proprietà

Step 1: Alice utilizza una chiave simmetrica, *KS*, per inviare una e-Mail a bob:

***Ks** è la simmetrica chiave di sessione (usata per quello specifico messaggio)*
***Kb** è chiave asimmetrica pubblica del destinatario (viene usato per cifrare la Ks)*



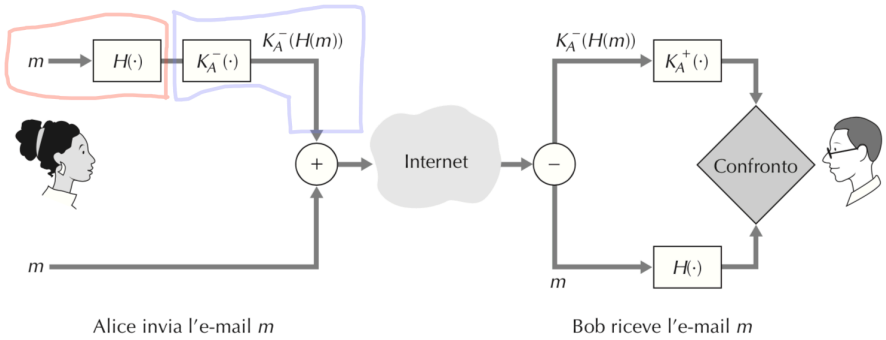
- (*rosso*) Alice cifra il messaggio con una chiave *Ks*, in questo modo ottiene il messaggio cifrato
- (*blu*) Alice cifra la sua chiave di sessione *Ks* con la chiave pubblica di bob *Kb*, in questo modo cifra la sua chiave di sessione e solo Bob con la chiave privata potrà decifrare la chiave di sessione *Ks* per decifrare poi il contenuto del messaggio.

Il tutto viene poi unito e inviato a Bob

- Non ho integrità del messaggio e autenticazione del mittente. Garantisco solo la sicurezza e l'autenticazione destinatario

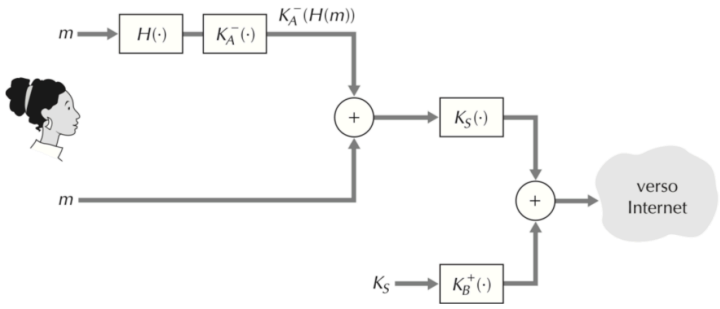
Step 2: Alice utilizza una funzione *HASH* e firma digital per garantire l'autenticità del messaggio:

***H** è la funzione di HASH per garantire l'integrità*
***Ka** è la chiave privata di Alice*



- (*rosso*) Alice utilizza una funzione di HASH per garantire l'integrità del messaggio.
- (*blu*) Alice cifra il risultato della funzione di HASH con la sua chiave privata *Ka* per garantire l'autenticità.

L'unione di questi due step permette ad Alice di inviare una e-Mail a Bob garantendo le (4) proprietà della sicurezza:



PGP

Pretty Good Privacy

PGP stato implementato fine anni ’90 e poi reso disponibile come software libero ed è lo schema citato sopra.

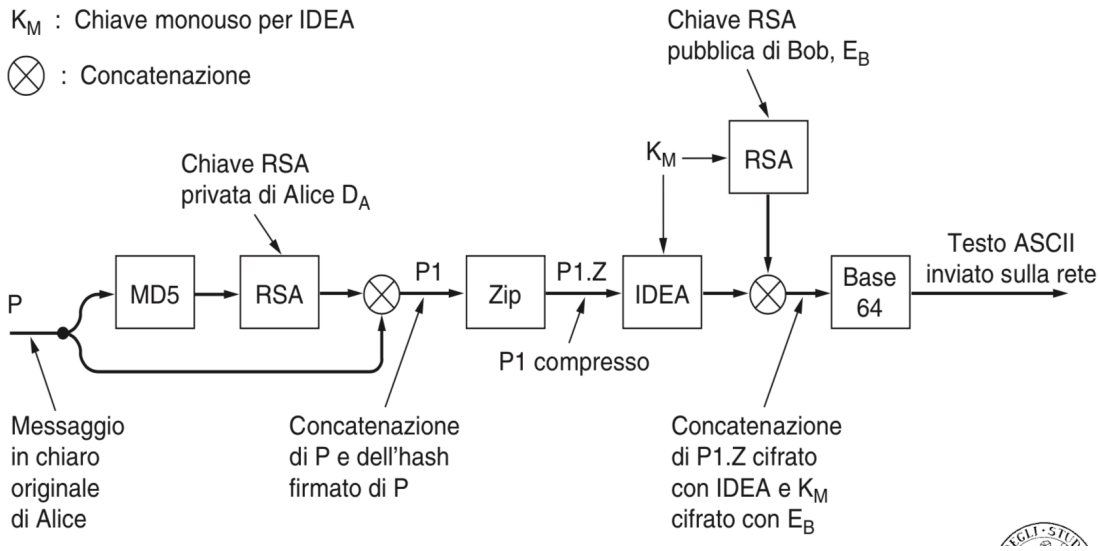
Un problema riscontrato in *PGP* è che *SMTP* è un protocollo testuale, quindi non posso inviare i binari che ottengo con le funzioni di cifratura e devo trasformare il contenuto di *PGP* in testo. (Risolvo ciò con conversione a base64)

[base64]
Ho 64 simboli: a-zA-Z0-9+/… Dato un file binario come input lo divido in gruppi da 6 bit e gli traduco in caratteri

I <u>servizi</u> che PGP offre sono:

- Autenticazione (*SHA-1, RSA, firme*)
- Confidenzialità (*CAST, IDEA, triplo-DES*)
- Compressione
- Codifica per compatibilità (*Radix-64*)
- Segmentazione

Esempio funzionamento PGP:



PGP si basa anche l’<u>uso della fiducia</u>:

- L’utente assegna un livello di fiducia ad ogni utente e intermediario
- PGP assegna un livello di fiducia nell’abbinamento chiave-utente

SSL / TLS

Sicurezza livello di trasporto

SSL & TLS offrono confidenzialità e integrità. SSL è stato principalmente stati progettati per rendere sicure le transazioni nella rete, poi è diventato uno standard con TLS.

Handshake SSL/TLS:

La fase di handshake prevede ad
- Autenticare (*se richiesto*) solo il server o anche il client
- Accordare le parti su che algoritmi cifrari usare
- Generare le chiavi di cifratura da usare

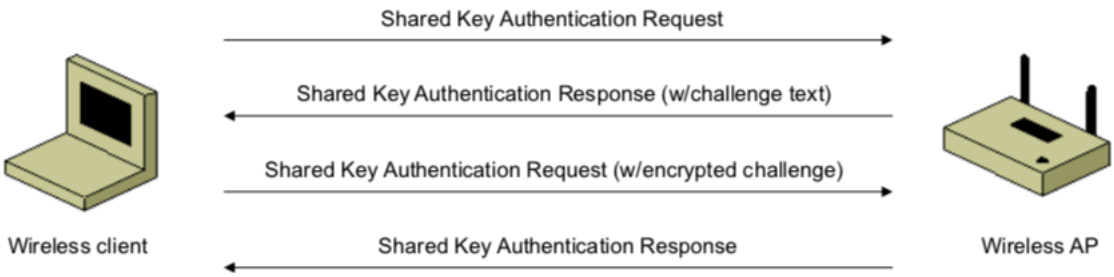
Wireless LAN

Sicurezza livello di collegamento

WEP: Wired Equivalent Privacy (802.11)

È un protocollo utilizzato per proteggere le reti wireless e fornisce una minima autenticazione per evitare che chiunque si collegasse alla WLAN.
Serve per cifrare i dati tra utente e AP: la chiave usata è comune a tutti gli utenti. Ogni utente ha la stessa chiave, se intercetto la chiave catturo il traffico di tutti gli utenti.

WEP - Autenticazione:



Sebbene usi una *secret key*, e un check-sum per garantire l'autenticità, WEP offre una protezione minima.

WPA / WPA2: Wifi Protected Access

WPA e WPA2 forniscono le seguenti caratteristiche crittografiche:
- Integrità e cifratura dei dati con gli standard 802.1x
- Protezione da attacchi "replay"
- Operano a livello Media Access Control