Università degli Studi di Verona

DIPARTIMENTO DI INFORMATICA

Ultima modifica: September 22, 2017-

Sicurezza di Rete

Riassunto dei principali argomenti

Candidati:
Davide Bianchi
Matteo Danzi

Anno Accademico 2016/2017

Indice

1 Introduzione **2**

2 Cenni di crittografia **2**
2.1 Crittografia a chiave simmetrica 3
2.2 Crittografia a chiave asimmetrica 3

3 Integrità **3**
3.1 Funzioni Hash 3

4 Autenticità **4**

5 Autenticazione **4**
5.1 Autenticazione diretta 5
5.2 Autenticazione indiretta 5
5.3 Autenticazione offline 5

6 Autorizzazione **5**
6.1 Meccanismi di controllo dell'accesso 5
6.1.1 Matrice di controllo dell'accesso 5
6.1.2 Access Control List (ACL) 5
6.1.3 Capabilities 5
6.2 Politiche per il controllo dell'accesso 6
6.2.1 Discretionary Access Control - DAC 6
6.2.2 Mandatory Access Control - MAC 6
6.3 Gruppi 6

7 Intrusion detection systems **6**

8 Firewall **6**

9 Sicurezza delle mail **6**

10 Sicurezza a livello di trasporto **6**

11 Sicurezza di wireless LAN **6**

12 Credits **6**

Sommario

Questa dispensa è scritta per la parte teorica del corso di Programmazione e sicurezza di Rete. Non sono inclusi gli argomenti: SDN e la parte riguardante le tecnologie di comunicazione in rete (fibra, cavi ETH ecc.) Il codice LaTeXè disponibile a `https://github.com/alx79/dispense-info-univr.git`

1 Introduzione

Il fatto di garantire una protezione a determinati assets implica il garantire di alcune proprietà:

1. Confidenzialità: un utente non dovrebbe venire a conoscenza di cose che non è autorizzato a conoscere (riservatezza dei dati, privacy);
2. Disponibilità: rendere disponibili ad un utente autorizzato le informazioni che può avere e che richiede;
3. Integrità: impedire l'alterazione di dati e informazioni in maniera diretta o indiretta (anche in seguito a incidenti);
4. Autenticità: ad un utente deve essere garantita l'autenticità delle informazioni che riceve;
5. Tracciabilità: le azioni di un utente devono essere tracciate in modo univoco, in modo evitare eventuali casi di ripudiablità.

Ciò che può compromette le caratteristiche sopra elencate sono le minacce e gli attacchi. Viene definita *minaccia* una possibile violazione della sicurezza, mentre invece un *attacco* è una violazione effettiva della sicurezza.

Gli attacchi possono essere sostanzialmente di 4 tipologie:

- *attivi*: tentativi di alterare il funzionamento di un sistema;
- *passivi:* tentativi di carpire informazioni senza intaccare i meccanismi del sistema;
- *interni:* effettuati da un'entità interna al sistema;
- *esterni:* effettuati da un'entità esterna al sistema.

Gli attacchi (o le minacce) sono suddivisi in classi:

- *disclosure:* accesso non autorizzato alle informazioni;
- *deception:* accettazione di dati falsi;
- *disruption:* interruzione o prevenzione di informazioni corrette;
- *usurpation:* controllo non autorizzato di alcune parti del sistema.

2 Cenni di crittografia

La crittografia è una scienza che si occupa di nascondere un'informazione rendendola sicura in modo che un terzo utente non autorizzato possa avervi accesso.

Un algoritmo crittografico è una funzione che prende in ingresso un messaggio in chiaro (*plaintext*) e produce un testo cifrato (*ciphertext*). Gli algoritmi crittografici sono di due categorie:

- a chiave *simmetrica*: le chiavi di cifratura e decifratura sono uguali;
- a chiave *asimmetrica*: vengono usate due chiavi differenti, una chiave è pubblica, l'altra è privata.

Ogni algoritmo crittografico deve essere robusto, vale a dire:

- deve essere difficile ottenere il testo in chiaro senza chiave da quello cifrato;
- dato un testo cifrato e uno in chiaro ottenere la chiave di cifratura.

Bisogna tenere sempre a mente che **nessun algoritmo crittografico è assolutamente sicuro**, quindi un algoritmo di dice *computazionalmente sicuro* se il costo necessario a violarlo è superiore a quello dell'informazione contenuta, oppure il tempo necessario a violarlo è superiore al tempo di vita dell'informazione.

In ogni caso, per analizzare un algoritmo crittografico, bisogna mantenere presente che la segretezza deve risiedere nella chiave, non nella struttura dell'algoritmo.

2.1 Crittografia a chiave simmetrica

La crittografia a chiave simmetrica utilizza una chiave condivisa e gli stessi algoritmi per cifrare e decifrare le informazioni (ovviamente la chiave deve essere scambiata su canali sicuri). Un classico esempio di cifrario a chiave simmetrica è il **cifrario di Cesare**, che usa come chiave un alfabeto non ordinato. In questo caso è facile ottenere la chiave perchè si può procedere con l'analisi delle frequenze, ovvero il l'analisi di quanto spesso in un testo si presenta una stessa sillaba.

Un altro esempio di cifrario a chiave simmetrica è costituito dai cifrari a blocchi, che permutano k bit. Le permutazioni possono poi venire combinate per ottenere schemi più complessi.

Alcuni esempi di agloritmi a chiave simmetrica sono:

- DES: chiavi a 56 bit, ormai obsoleto;
- Triplo-DES: un DES applicato 3 volte con chiavi diverse (di lunghezza 112 o 168 bit);
- AES: usa chiavi a 128, 192 o 256 bit.

Tutti questi algoritmi sono soggetti al problema della distribuzione delle chiavi. Nel 1976 due crittologi (Diffie e Hellman) propongono un sistema che supera questo problema, perchè **non condivide chiavi**.

2.2 Crittografia a chiave asimmetrica

In questo tipo di crittografia ogni utente ha una coppia di chiavi, una **pubblica**, che viene resa nota, e una **privata**, che viene mantenuta segreta. L'idea è che il messaggio venga cifrato con la chiave pubblica del destinatario, che potrà poi decifrarlo con la sua chiave privata.

L'uso di algoritmi a chiave asimmetrica comporta i seguenti vantaggi:

- non è più necessario scambiarsi chiavi;
- la stessa chiave pubblica può essere usata da più utenti.

I requisiti principali di un sistema crittografico di questo tipo sono:

- deve essere semplice la generazione delle chiavi;
- devono essere semplici le operazioni di cifratura/decifratura quando si ha la chiave;
- deve essere difficile ricavare la chiave privata da quella pubblica;
- deve essere difficile ricavare il testo in chiaro avendo il testo cifrato e la chiave pubblica.

Un esempio di algoritmo simmetrico è RSA, che si basa sulla difficoltà di scomporre un numero in fattori primi. Le chiavi usate da RSA hanno le dimensioni di 2^{10} bit.

Gli algoritmi asimmetrici richiedono molte più risorse degli algoritmi a chiave simmetrica (sono infatti anche molto più lenti), e vengono usati per scambiarsi una chiave di sessione che verrà poi usata da un algoritmo simmetrico sicuro, computazionalmente più efficiente.

3 Integrità

Lo scopo storico della crittografia è quello di garantire la privacy, ossia come garantire che un'informazione ricevuta provenga effettivamente dall'utente che ci si aspetta l'abbia mandata.

3.1 Funzioni Hash

Una funzione hash è una funzione che trasforma un messaggio di lunghezza arbitraria in uno di lunghezza fissa (viene chiamato *hash* o *digest* del messaggio originale). Le funzioni hash attualmente più utilizzate sono MD5 e SHA.

Per soddisfare le condizioni di sicurezza, gli algoritmi che gestiscono le funzioni hash dovrebbero avere le seguenti caratteristiche:

- *coerenti*: a input uguali corrispondono output uguali;
- *casuali*: per impedire l'interpretazione del messaggio originale;
- *univoci*: la probabilità che due messaggi generino due hash uguali deve essere remota;
- *non invertibili*: deve essere impossibile (o computazionalmente complesso) risalire dal digest al messaggio originale.

Le funzioni hash vengono anche usate come fingerprint per verificare che nessuno sia intervenuto sul messaggio originale (altrimenti i due digest sarebbero diversi, vedi esempio).

Ora daremo un esempio di come possa avvenire una comunicazione sfruttando le funzioni hash. Alcune definizioni:

- m è il messaggio in chiaro;
- $H(m)$ è l'hash del messaggio;
- $c(x)$ è la funzione di cifratura;
- A e B sono due utenti.

Indichiamo inoltre come $H_A(m)$ l'hash del messaggio scritto da A.

Esempio. A scrive un messaggio e ne utilizza il testo come input di una funzione di hash, che genera il digest $H_A(m)$. A poi manda $c(m + H_A(m))$ a B.

B decifra e separa il contenuto del messaggio cifrato che ha ricevuto, e calcola con la funzione di hash un hash denominato $H_B(m)$. Se vale

$$H_B(m) = H_A(m)$$

il messaggio è autentico.

Se i due utenti non sono interessati a mantenere occultato il messaggio, viene utilizzato un *MAC* (Message Authentication Code), un segreto condiviso conosciuto da entrambi gli utenti. In questo caso viene mandato al destinatario il pacchetto con

$$m + H_A(m + s)$$

Usando un MAC si ha anche garanzia di autenticità, grazie al segreto condiviso. Qui sorge un nuovo problema: come poter scambiare con l'altro utente un segreto condiviso su un canale protetto? Per ovviare a questo problema è stato proposto un meccanismo di *firma digitale*, che **non usa chiavi segrete**.

4 Autenticità

Il meccanismo di firma digitale è applicato con un algoritmo a chiave asimmetrica combinato con una funzione hash.

La firma di un documento procede nel seguente modo:

1. Viene calcolato l'hash del messaggio;
2. Viene firmato l'hash del messaggio con la chiave privata del mittente;
3. Si manda al destinatario la hash firmata del messaggio e il messaggio.

Per la verifica dell'autenticità della firma digitale si procede così:

1. Si decifra l'hash firmato ricevuto usando la chiave pubblica del mittente;
2. Si calcola l'hash del messaggio ricevuto;
3. Si comparano l'hash calcolato con quello ottenuto decifrando l'hash firmato: se combaciano, la firma è autentica.

Con un meccanismo così descritto, sorge questo problema: come è possibile verificare che la chiave che si sta usando per decifrare il messaggio ricevuto sia proprio quella del mittente? Potrebbe essere quella di un malintenzionato che si spaccia per il mittente.

Per ovviare a questo problema sono nati i certificati, che appunto certificano che una chiave è proprio quella dell'utente con cui si sta comunicando. La certificazione si ottiene mediante una CA (*certification authority*). I certificati sono emessi secondo uno standard, e contengono come campi i dati relativi all'utente, un numero seriale e un periodo di validità (vedi slide per completezza). I certificati vengono gestiti da una PKI (*public key infrastructure*) gerarchica.

5 Autenticazione

L'autenticazione è un servizio di sicurezza usato per garantire l'identità degli interlocutori.

Si possono distinguere 4 modalità di autenticazione:

- *locale*: l'utente accede in locale al servizio, che effettua l'autenticazione;
- *diretta*: l'utente accede da remoto al servizio, che effettua direttamente l'autenticazione;
- *indiretta*: l'utente accede da remoto ai servizi, che si appoggiano su un sistema di autenticazione separato;
- *off-line*: i servizi possono prendere decisioni autonome, senza contattare ogni volta l'autorità di autenticazione.

5.1 Autenticazione diretta

L'autenticazione diretta prevede uno scambio diretto della password. Per evitare registrazioni della password da parte di terzi, si usano delle "sfide" a cui l'utente che si autentica deve rispondere per dare prova della sua identità.

5.2 Autenticazione indiretta

L'autenticazione indiretta si basa su un sistema di autenticazione esterno, e gli altri sistemi si appoggiano su di esso. Esempi di questo tipo di servizi sono **Radius** e **Kerberos**. Spesso sono implementati come sistemi SSO, per garantire con un solo login l'accesso a tutti i servizi di un certo dominio amministrativo.

L'autenticazione basata su sistemi di questo tipo funziona nel seguente modo:

1. l'utente effettua il login sul server SSO;
2. l'utente riceve dal server SSO un token di autenticazione;
3. il token viene mandato dall'utente al server dove il servizio che necessita di autenticazione risiede;
4. il server con il servizio confronta il token ricevuto confrontandolo con il server SSO;
5. il server SSO conferma il token;
6. il server con il servizio manda il contenuto necessario al client che ha a questo punto eseguito il login.

5.3 Autenticazione offline

L'autenticazione offline è basata sull'uso di certificati digitali distribuiti da una CA, che associano l'identità di un utente ad una chiave pubblica.

Per ottenere un certificato digitale, l'utente deve generare sul proprio pc, tramite un opportuno browser, una coppia di chiavi. L'utente deve poi mandare alla CA la chiave pubblica insieme ad una richiesta di certificato, che lo manda di persona ad un *Local Validation Point* (LVP). Completato questo passaggio, la CA manda il certificato all'utente per mail e inserisce la chiave nell'elenco delle chiavi pubbliche certificate.

L'intera sequenza di operazioni viene svolta all'interno di una PKI (*Public Key Infrastructure*), ovvero una gerarchia di CA che si certificano a cascata.

6 Autorizzazione

L'autorizzazione è il servizio di controllo dell'accesso volto a garantire che solo gli utenti autorizzati possano avere accesso ai relativi file.

Per attuare un controllo degli accessi efficiente ci si serve di politiche (che definiscono l'attribuzione dei privilegi ai soggetti) e meccanismi (che specificano le relazioni soggetto-oggetto). Per assegnare in maniera più affidabile possibile i privilegi, ci si affida a due principi:

- **privilegio minimo**: ogni utente dovrebbe avere solo i privilegi minimi necessari a compiere le azioni che deve svolgere;
- **separazione dei compiti**: ogni utente dovrebbe avere privilegi tali da impedirgli di sovvertire il sistema.

6.1 Meccanismi di controllo dell'accesso

6.1.1 Matrice di controllo dell'accesso

È una matrice che contiene i soggetti come righe e gli oggetti come colonne. In ogni cella della matrice sono inseriti i privilegi per quella data copia soggetto/oggetto. A lungo andare, se i soggetti e gli oggetti sono numerosi, possono sorgere problemi di scalabilità.

6.1.2 Access Control List (ACL)

È una variante della matrice sopra descritta, in cui si memorizza la matrice per colonne, e in ogni cella si mettono i soggetti che possono interagire con l'oggetto nella colonna relativa e con i permessi concessi.

6.1.3 Capabilities

È il duale della ACL: per ogni soggetto si salvano la lista degli oggetti con i relativi permessi, cosa che permette di gestire in maniera efficiente i permessi associati al singolo utente.

6.2 Politiche per il controllo dell'accesso

6.2.1 Discretionary Access Control - DAC

È un approccio in cui i singoli utenti possono modificare i permessi dei file che sono sotto il loro controllo (ci si basa sul concetto di *ownership*). Il modello DAC è flessibile ma non permette di controllare la diffusione dell'informazione (infatti un utente con permessi su un oggetto potrebbe mandarlo ad uno che non li ha).

6.2.2 Mandatory Access Control - MAC

In questo modello la politica di controllo dell'accesso è regolata dal sistema in maniera centralizzata, non dai singoli utenti. Il sistema decide quindi quali permessi dare a quali utenti. È un sistema più robusto ma meno flessibile.

6.3 Gruppi

Per rendere più efficiente l'organizzazione dei permessi si possono raggruppare gli utenti di un sistema in gruppi di utenti, per semplificare l'associazione di determinati oggetti a determinate categorie di utenti (che sono appunto raggruppati).

7 Intrusion detection systems

8 Firewall

9 Sicurezza delle mail

10 Sicurezza a livello di trasporto

11 Sicurezza di wireless LAN

12 Credits

Davide Bianchi (mail: davideb1912@gmail.com)
Matteo Danzi (mail: matteodanziguitarman@hotmail.it)