


# Sicurezza


Corso di "Sistemi per Elaborazione dell'Informazione"  
 Prof. Carpentieri Bruno  
 A.A. 2004/2005

Di Cerbo Raffaella  
 Fasulo Giuseppina




## Introduzione

- Un sistema è **sicuro** se le sue risorse si adoperano e vi si accede soltanto nei modi previsti.
- Ottenere una sicurezza totale è impossibile; ma esistono meccanismi che rendono la violazione della sicurezza un caso raro.
- Le violazioni della sicurezza del sistema si dividono in **dolose** o **accidentali**.




## Introduzione

- Tra le forme di accessi dolosi ci sono le seguenti:
  - Letture non autorizzate di dati;
  - Alterazione non autorizzata dei dati;
  - Distruzione non autorizzata dei dati;
  - Impedimento del legittimo uso del sistema (rifiuto del servizio);
- Ad esempio, i grandi sistemi commerciali, contenenti dati relativi agli stipendi o ad altre attività finanziarie, sono obiettivi invitanti per i ladri.




## Protezione del sistema

- Per proteggere il sistema è necessario prendere misure di sicurezza a 4 livelli:
  1. **Fisico**: i siti che ospitano i sistemi di calcolo devono essere protetti fisicamente contro gli accessi furtivi da parte di intrusi.
  2. **Umano**: occorre vagliare gli utenti per ridurre la possibilità di concedere autorizzazioni a utenti che forniscano l'accesso a intrusi.
  3. **Rete**: molti dati informatici nei sistemi moderni viaggiano in linee a noleggio private, linee condivise o linee telefoniche. L'intercettazione di questi può essere dannosa. Un attacco per rifiuto del servizio potrebbe causare l'interruzione delle comunicazioni.
  4. **Sistema operativo**: il sistema deve proteggere se stesso dalle violazioni della sicurezza dolose o accidentali.



## Autenticazione degli utenti

- Per i sistemi operativi il principale problema di sicurezza è l'**autenticazione**.
- Ogni utente identifica se stesso; si tratta di stabilire se l'identità di un utente è autentica.
- L'autenticazione è basata su uno o più dei seguenti tre elementi:
  - Oggetti posseduti dall'utente (Chiave o scheda);
  - Conoscenze dell'utente (identificatore o parola d'ordine);
  - Attributo dell'utente (impronta digitale, della retina o firma).



## Parole d'ordine (password)

- Quando un utente si identifica con il proprio identificatore, riceve la richiesta d'immissione della parola d'ordine.
- Se la parola d'ordine immessa corrisponde a quella memorizzata nel sistema, il sistema presume che l'utente sia legittimo.
- Ad esempio, si potrebbe associare una parola d'ordine ad ogni risorsa, come un file; in questo modo ogni volta che si chiede l'uso della risorsa si deve fornire la relativa parola d'ordine, se questa è giusta si permette l'accesso.

## Vulnerabilità delle parole d'ordine

- Le parole d'ordine sono assai comuni poiché sono facili da capire e da usare. Sfortunatamente sono anche facili da indovinare.
- Ci sono due metodi per indovinare una parola d'ordine:
  1. basato sulla conoscenza dell'utente;
  2. Uso della forza bruta.
    - **Esempio:** una parola d'ordine di 4 cifre decimali permette solo 10000 combinazioni quindi occorrono solo 5000 tentativi per indovinare quella giusta. Un programma che fa un tentativo ogni millisecondo impiega solo 5 secondi ad indovinare una parola d'ordine di 4 cifre.
- In generale, è più difficile indovinare parole d'ordine lunghe o che distinguono lettere maiuscole dalle minuscole.

## Vulnerabilità delle parole d'ordine

- Il fallimento della sicurezza può derivare da:
  1. **Shoulder surfing:** un intruso può sbirciare sopra la spalla di un utente mentre questo inizia una sessione di lavoro e capire la sua parola d'ordine.
  2. **Sniffing:** chiunque abbia accesso alla rete in cui risiede un calcolatore può inserire un monitor di rete che gli consenta di osservare tutti i dati trasferiti nella rete, comprese le parole d'ordine. La cifratura dei flussi di dati contenenti le parole d'ordine risolve questo problema.

## Vulnerabilità delle parole d'ordine

- Le parole d'ordine possono essere:
  - **Generate dal sistema:** difficili da ricordare e quindi l'utente deve trascriverle;
  - **Scelte dall'utente:** più facili da indovinare.
- Lo schema delle parole d'ordine ha delle varianti:
  - **Invecchiamento:** si costringe l'utente a modificare la parola d'ordine a intervalli regolari. Poiché un utente può alternare sempre le stesse parole d'ordine, il sistema registra l'elenco delle ultime  $n$  parole d'ordine per ogni utente per impedirne il riutilizzo;
  - **Alla fine di ogni sessione di lavoro:** il sistema o l'utente sceglie una nuova parola d'ordine per ogni sessione di lavoro successiva. In questo modo, in caso di abuso, la parola d'ordine può essere usata una sola volta.

## Parole d'ordine cifrate

- Il sistema UNIX si serve della **cifratura** per evitare di mantenere segreto il proprio elenco di parole d'ordine.
- Il sistema contiene una funzione difficile da invertire, in cui dato un valore  $x$  è facile calcolare  $f(x)$ , ma dato il valore di  $f(x)$  è "impossibile" calcolare  $x$ .
- Quando un utente immette una parola d'ordine questa viene cifrata e confrontata con quella già cifrata e memorizzata. Quindi anche se la parola d'ordine cifrata viene vista, non potendo essere decifrata è impossibile stabilire quella originale.

## Parole d'ordine cifrate

**Pecca** del metodo di cifratura UNIX: Chiunque disponga di una copia del file che contiene le parole d'ordine cifrate, può servirsi di efficienti procedure, ad esempio cifrando ciascuna parola di un dizionario e confrontando il risultato con le parole d'ordine. Se l'utente ha scelto come parola d'ordine una parola contenuta nel dizionario, la parola d'ordine viene scoperta.

- Poiché i sistemi UNIX fanno uso di un algoritmo di cifratura noto, un pirata informatico (cracker) potrebbe conservare un insieme di coppie <parola d'ordine, parola d'ordine cifrata> per individuare rapidamente le parole d'ordine già violate.
- Le nuove versioni dello UNIX registrano le parole d'ordine cifrate in un file che può essere letto solo dall'amministratore del sistema.
- Per evitare il metodo di cifratura del dizionario alcuni sistemi non consentono l'uso di parole del dizionario come parola d'ordine.

## Parole d'ordine monouso

- **Parole d'ordine accoppiate:** All'inizio di una sessione di lavoro, il sistema sceglie a caso una coppia di parole d'ordine dall'insieme delle parole d'ordine accoppiate e presenta all'utente un elemento della coppia selezionata; l'utente deve fornire l'altro elemento. Il sistema *esige* dall'utente la risposta corretta.
- L'utente e il sistema condividono un'informazione segreta che non è mai trasmessa, ma si usa un insieme con un seme, condiviso, come argomento di una funzione.
- In questo sistema la parola d'ordine cambia ad ogni richiesta di accesso.

## Parole d'ordine monouso

- Il **seme** (seed): numero casuale, sequenza alfanumerica che costituisce la richiesta d'identificazione da parte del sistema.
- L'informazione segreta e il seme si usano come argomenti della funzione  $f(\text{informazione}, \text{seme})$ , il valore di questa funzione è la parola d'ordine che si comunica al calcolatore.
- Il calcolatore è a conoscenza sia del seme che dell'informazione segreta, quindi calcola il valore della funzione: se i due risultati coincidono, l'identità dell'utente è autentica.

## Parole d'ordine monouso

- Varianti:
  - **Autenticazione secondo due fattori**: uso di un generatore di parole d'ordine monouso e di un PIN. L'utente immette l'informazione segreta PIN (numero d'identificazione personale) per mezzo di una tastiera, il visore mostra la parola d'ordine monouso.
  - **Libro dei codici**: elenco di parole d'ordine usa e getta, ogni parola d'ordine dell'elenco si usa, nell'ordine, solo una volta e poi si cancella dalla lista.

## Tecniche biometriche

- Per garantire l'accesso sicuro ad ambienti fisici spesso si usano strumenti per la rilevazione dell'impronta digitale.
- I dispositivi leggono il motivo formato dalle increspature della pelle sulle dita e lo convertono in una sequenza di numeri.
- Uno specifico programma esegue la scansione del dito e confronta i dati ottenuti con quelli memorizzati, determinando se il dito sulla tavoletta corrisponde a quello relativo ai dati memorizzati.

## Minacce a programmi: Cavalli di troia

- Segmento di codice che abusa del suo ambiente.
- Programma apparentemente utile che contiene codice nascosto, che quando invocato, esegue qualche funzione non voluta o dannosa.
- Esempio: un programma che sembra stia eseguendo una funzione utile, come la calcolatrice, in realtà sta cancellando in maniera silenziosa i file dell'utente.

## Minacce a programmi: Cavalli di troia

- Variante del cavallo di troia: programma che emula una procedura di inizio di una sessione di lavoro.
- Esempio: un utente, ignaro, nella fase di accesso a un terminale crede di aver scritto erroneamente la propria parola d'ordine; prova ancora e, questa volta, ha successo.

Realmente è accaduto che un emulatore della procedura d'accesso alla sessione di lavoro, sottrae il nome dell'utente e la parola d'ordine. L'emulatore registra la parola d'ordine e mostra un messaggio di errore, interrompe la propria esecuzione e lascia l'utente alla vera procedura d'accesso.

## Minacce a programmi: Trabocchetti

- **Trabocchetto** (trap door): il progettista di un programma o di un sistema può lasciare nel programma un "buco" segreto.
- Risulta essere difficile individuarli poiché è necessario analizzare tutto il codice sorgente dei componenti del sistema, e dato che i sistemi possono essere composti da milioni di righe di codice, queste analisi non si fanno spesso.
- Esempio: alcuni programmatori sono stati arrestati per aver truffato banche inserendo errori di arrotondamento per ottenere l'accredito nei propri conti.

## Minacce a programmi: Alterazione della pila

- Modo più diffuso col quale un utente esterno al sistema può ottenere un accesso non autorizzato a esso.
- L'attacco sfrutta un errore nel programma, ad esempio trascurare di controllare che si rispettino i limiti di un elemento da riempire con i dati.
- L'aggressore determina i punti vulnerabili e scrive un programma che permette di compiere le seguenti azioni:
  1. Superare(overflow)i limiti di un campo da riempire immettendo dati, ad esempio di un vettore che riceve dati in ingresso, fino a invadere il buffer.

## Minacce a programmi: Alterazione della pila

2. l'invasione della pila si deve compiere in modo da sovrascrivere il corrente indirizzo di ritorno annotato nella pila con l'indirizzo del segmento di codice che compie l'attacco;
  3. scrivere il segmento di codice, per l'area successiva nella pila, che include i comandi che l'aggressore desidera eseguire, ad esempio attivare un interprete di comandi;
- Attacco particolarmente dannoso poiché si esegue all'interno di un sistema e si può trasmettere nei canali di comunicazione.

## Minacce a programmi: Alterazione della pila

- Questo attacco avviene attraverso protocolli che si usano per comunicare con il calcolatore, quindi può essere difficile da individuare o prevenire.
- Una possibile soluzione consiste nel dotare la CPU di una funzione che le consente d'impedire l'esecuzione di codice presente in una sezione della memoria assegnata alla pila.

## Minacce ai sistemi: worm

- Un worm è un processo che sfrutta gli effetti della proliferazione per minare le prestazioni del sistema.
- Genera continuamente copie di se stesso logorando le risorse del sistema, fino a renderlo inutilizzabile da tutti gli altri processi.
- Diventano efficaci nelle reti di trasmissione, poiché possono riprodursi in diversi sistemi a esse collegati e far crollare l'intera rete.

## Worm di R. T. Morris

- Uno studente inserì un programma worm in uno o più calcolatori connessi alla rete Internet.
- Il programma si propagò rapidamente per grandi distanze, entro poche ore dalla sua immissione consumò le risorse dei sistemi infetti fino a causarne il crollo.
- Il worm era composto da due programmi:
  - **Rampino d'arrembaggio:** Costituito da 99 linee di codice in linguaggio C da compilare ed eseguire in ogni calcolatore raggiunto, una volta stabilitosi si connetteva al calcolatore che lo aveva generato e caricava una copia del programma principale nel sistema agganciato.
  - **Programma Principale:** Iniziava la ricerca di altri calcolatori cui il sistema infettato poteva connettersi con facilità.

## Worm di R. T. Morris

- Morris sfruttò il comando **rsh**: comando di rete del sistema operativo UNIX che facilita le esecuzioni remote.
- Il worm analizzava file speciali, contenenti un elenco di coppie di nomi <calcolatore, nome d'utente>, alla ricerca dei nomi dei siti che avrebbero consentito l'esecuzione remota senza chiedere l'immissione della parola d'ordine, attivava in questi sistemi un interprete dei comandi remoto e vi caricava il programma principale e ricominciava l'esecuzione.
- Esistono altri due metodi d'infezione che sfruttavano banchi del sistema operativo presenti nei programmi *finger* e *sendmail* di UNIX

## Morris: programma finger

- **Finger:** funziona come una guida telefonica elettronica, il comando `finger nome_d'utente@nome_del_calcolatore` riporta il nome reale per l'accesso al sistema di una persona, e altre informazioni.
  - Viene eseguito come processo in sottofondo in ciascun sito e risponde alle richieste inviategli da qualsiasi sito della rete Internet.
  - Il punto vulnerabile era costituito dal fatto che la lettura dei dati ricevuti in ingresso era eseguita senza controllare i limiti del vettore di memoria che avrebbe dovuto contenerli, quindi il codice eseguiva un attacco per alterazione della pila.

## Morris: programma finger

- Il programma di Morris inviava al comando `finger` una richiesta costituita di una sequenza di 536 byte affinché oltrepassasse i limiti del vettore.
- Anziché restituire il controllo alla funzione `main` veniva deviato a una procedura contenuta nella sequenza di 536 byte ora residente nella pila.
- Questa procedura eseguiva il comando `/bin/sh`, che in caso di successo dava al worm un interprete dei comandi remoto nella macchina presa di mira.

## Morris: programma sendmail

- **Sendmail:** instrada i messaggi di posta elettronica in un ambiente di rete.
  - Morris incluse una chiamata al comando `debug`, che invece di specificare un indirizzo d'utente come in normali casi di verifica, inviava una serie di comandi che spedivano ed eseguivano una copia del programma avente la funzione di rampino d'arrembaggio.
  - Giunto sul posto, il programma principale del worm faceva una serie di tentativi per scoprire le parole d'ordine degli utenti secondo un algoritmo di tre fasi:
    - Ricercava utenti senza parola d'ordine;
    - Ricercava in un dizionario interno di 432 lemmi;
    - Ricercava nel dizionario in linea dello UNIX.

## Morris: programma sendmail

- Il worm cercava in ogni utenza violata nuovi file di dati di `rsh`, facendo dei tentativi su tutti gli elementi di `rsh` per ottenere l'accesso a utenze in sistemi remoti.
- Ad ogni nuovo accesso il worm cercava copie di se stesso già attive. Se ne trovava una, la nuova copia terminava la propria esecuzione, a esclusione di ogni settima istanza.
- Consentendo ad ogni settimana copia di proseguire, si portò alla totale infestazione di sistemi Sun e VAX connessi alla rete Internet.

## Worm di Morris: il suo arresto

- Le caratteristiche di rete dello UNIX che favorirono la propagazione del worm contribuirono anche al suo arresto.
- Nel giro di pochi giorni erano disponibili specifici programmi per la correzione provvisoria dei difetti della sicurezza sfruttati dal worm.

## Minacce ai sistemi:virus

- Frammento di codice che s'inserisce in un programma legittimo.
- Progettati per diffondersi e seminare distruzione in un sistema, modificando o distruggendo file e causando crolli dei sistemi e malfunzionamento dei programmi.
- Di solito i virus sono diffusi da utenti che prelevano programmi infetti dalle banche elettroniche pubbliche(BBS) o a causa dello scambio di dischi infetti.
- Una comune forma di trasmissione dei virus è lo scambio di file della serie di programmi Microsoft Office.

## Minacce ai sistemi:virus

- Alcuni documenti prodotti col Microsoft Word possono contenere le *macro* che i programmi inclusi nel pacchetto Microsoft Office eseguono automaticamente.
- La miglior protezione contro i virus informatici è la prevenzione o la pratica dell'elaborazione sicura, ad esempio acquistare programmi sigillati dai rivenditori autorizzati
- Una misura di difesa potrebbe essere quella di evitare di aprire ogni allegato a un messaggio di posta elettronica proveniente da indirizzi sconosciuti.

## Minacce ai sistemi: Attacchi per rifiuto del servizio

- Questo tipo di attacco non prevede l'acquisizione d'informazioni o la sottrazione di risorse, piuttosto il blocco dell'utilizzo legittimo di un sistema o servizio.
- Sono attacchi che si compiono tramite la rete e si dividono due categorie:
  1. Attacchi che impiegano talmente tante risorse del sistema attaccato da non lasciarne libere per svolgere elaborazioni utili. Ad esempio, la pressione di un pulsante in un sito Web potrebbe portare al caricamento di un'applet Java che quando è in esecuzione usa tutto il tempo di CPU disponibile.

## Minacce ai sistemi: Attacchi per rifiuto del servizio

2. Attacchi che mirano a mettere in crisi la rete dell'organizzazione obiettivo dell'attacco.

- Abusano di alcune funzioni fondamentali dei protocolli TCP/IP. Ad esempio, se l'aggressore invia "voglio stabilire una connessione", ma a questa non fa mai seguire la parte che dall'altro capo ci si aspetta "la connessione è ora completa", ci si trova con molte sessioni TCP parzialmente avviate che possono assorbire tutte le risorse di rete del sistema, impedendo connessioni TCP legittime.
- Poiché si servono degli stessi meccanismi delle normali richieste di servizio, questi attacchi non si possono impedire.

## Migliorare la sicurezza dei sistemi

- Un metodo per migliorare la sicurezza di un sistema consiste in una sua scansione periodica, alla ricerca di varchi nella sicurezza, controllando vari aspetti del sistema come:
  - Parole d'ordine brevi o facili da indovinare;
  - Programmi privilegiati non autorizzati;
  - Programmi non autorizzati nelle directory del sistema;
  - Processi dall'esecuzione inaspettatamente lunga;
  - Improprie protezioni delle directory sia degli utenti che di sistema;
  - Improprie protezioni dei file dei dati del sistema, come il file delle parole d'ordine;

## Migliorare la sicurezza dei sistemi

- Elementi pericolosi nel percorso di ricerca dei programmi;
- Modifiche ai programmi di sistema individuate con somme di controllo;
- Demoni di rete inattesi o nascosti;
- Nasce il problema di come calcolatori fidati si possano collegare in modo sicuro a una rete non fidata, ad esempio Internet.
- Soluzione: uso di una barriera di sicurezza(*firewall*), che separa i sistemi fidati da quelli non fidati:un calcolatore o instradatore situato tra le due categorie di sistemi.
- Limita l'accesso di rete tra due domini di sicurezza, controlla e registra tutte le connessioni.

## Migliorare la sicurezza dei sistemi

- Internet è considerato dominio non fidato;come secondo dominio si prevede una zona parzialmente fidata: zona smilitarizzata(DMZ), e un terzo dominio che comprende i calcolatori aziendali.
- Connessioni consentite: da Internet ai calcolatori DMZ, dai calcolatori aziendali a quelli DMZ e a Internet.
- Connessioni non consentite: da Internet ai calcolatori aziendali.
- Il sistema che costituisce la barriera di sicurezza deve essere sicuro e resistente agli attacchi.

## Migliorare la sicurezza dei sistemi

- Vulnerabilità della barriera di sicurezza:
  - Non riescono a prevenire gli attacchi che si trasmettono all'interno di protocolli e connessioni che le stesse barriere permettono.
  - Spoofing: si sfruttano le tecniche di contraffazione dell'identità, tramite le quali un sistema non autorizzato finge di esserlo, riuscendo a soddisfare alcuni criteri di autorizzazione.

## Rilevamento delle intrusioni

- Attività volta a scoprire tentativi d'intrusione, intrusioni già avvenute ed attivare azioni appropriate in risposta alle intrusioni.
- Esistono molte tecniche per il rilevamento di intrusioni che si differenziano secondo vari fattori:
  - Fase in cui è avvenuto il rilevamento dell'intrusione: mentre si stava verificando o solo successivamente.
  - Le informazioni esaminate per scoprire l'attività intrusiva: comandi per l'interprete da parte dell'utente, chiamate del sistema da parte dei processi.
  - L'ampiezza della capacità di risposta: un sistema potrebbe sviare l'attività dell'intruso portandolo verso una trappola, risorsa fittizia.

## Elementi che caratterizzano un'intrusione

- I sistemi di rilevamento delle intrusioni IDS seguono uno dei due metodi per identificare un'intrusione:
  1. **Rilevamento basato sulle tracce:** esamina i dati d'ingresso al sistema o il traffico della rete alla ricerca di particolari schemi o sequenze di azioni, chiamate tracce che si ritengono indizi di attacchi.
    - Esempio: programma antivirus che analizza file binari alla ricerca di virus conosciuti.
  2. **Rilevamento di anomalie:** tecniche che cercano di identificare comportamenti anomali in un sistema.
    - Esempio: riconoscimento di una procedura d'accesso che si svolge in un orario anomalo per un certo utente.

## Elementi che caratterizzano un'intrusione

- Il rilevamento di anomalie può riuscire a scoprire metodi d'intrusione che in precedenza erano sconosciuti.
- Il rilevamento basato su tracce identifica solo attacchi conosciuti.
- Un IDS deve offrire una frequenza di falsi allarmi bassa.
- Consideriamo la probabilità di occorrenza di annotazione d'intrusione  $P(I)$ ;
- $I$ : evento che rappresenta l'occorrenza di un'annotazione che riflette un reale comportamento intrusivo.

## Elementi che caratterizzano un'intrusione

- Si supponga che  $A$  denoti l'evento dell'IDS che attiva un allarme.
- IDS dovrebbe massimizzare sia  $P(I|A)$  sia  $P(\neg I|\neg A)$ , sia la probabilità che un allarme indichi un'intrusione sia la probabilità che l'assenza di un allarme indichi che non vi sono intrusioni.
- Considerando  $P(I|A)$ , si calcola tale valore con l'uso della **formula di Bayes:**
$$P(I|A) = \frac{P(I) * P(A|I)}{P(I) * P(A|I) + P(\neg I) * P(A|\neg I)}$$
- Meno uno su sette allarmi indica una vera intrusione.

## Verifica e registrazione

- **Elaborazione dei tracciati di verifica:** metodo per il rilevamento delle intrusioni in cui gli eventi rilevanti per la sicurezza si memorizzano in un file di sistema formando tracciati di verifica da confrontare con tracce d'attacchi.
- Nei sistemi UNIX ci sono due programmi per creare ed analizzare tracciati di verifica e generare risposte:
  - **Syslog:** crea tracciati di verifica e fornisce una funzione d'invio di messaggi rilevanti per la sicurezza.
  - **Swatch:** applica tecniche di rilevamento, basate su tracce, ai tracciati di verifica generati da syslog mentre questi vengono generati e produce opportuni risultati nel caso in cui si rilevino intrusioni.

## Verifica e registrazione

- **Syslogd**: processo demone creato quando syslog viene installato in un sistema UNIX. Attende messaggi da varie possibili sorgenti e li dispone come prescritto nel file di configurazione syslog.conf.
- Il file syslog.conf contiene un insieme di coppie, con un campo selettore e uno di azione. Il campo selettore identifica i messaggi sui quali si deve compiere l'azione (es. invio del messaggio a un file).

## tripwire

- Strumento per il rilevamento di anomalie, per il controllo dell'integrità del file system del sistema UNIX.
- Opera seguendo l'ipotesi che una vasta classe d'intrusioni generi modifiche anomale nei file system e nelle directory di sistema.
- **tw.config**: file di configurazione che controlla il comportamento del tripwire, che elenca le directory e file per i quali si vogliono tenere sotto controllo modifiche, cancellazioni e aggiunte.
- tripwire offre diverse funzioni di hash resistenti alle collisioni cioè data  $f(x)$ , ottenuta applicando  $f$  ad un file  $x$ , è impossibile il calcolo di un file diverso  $x'$  t.c.  $f(x)=f(x')$ .

## tripwire

- Nella prima attivazione tripwire prende in ingresso il file tw.config e calcola una traccia per ciascun file o directory, contenente gli attributi che si devono controllare, le tracce si memorizzano in una base di dati.
- Le volte successive, tripwire considera come ingresso sia tw.config sia la base di dati precedentemente memorizzata, ricalcola la traccia per ogni file o directory elencato in tw.config e confronta questa traccia con quella memorizzata nella base di dati.

## tripwire

- Limiti:
  - Necessità di proteggere da modifiche non autorizzate i file di programma dello stesso tripwire e i file di dati associati, inclusa la base di dati. Ciò rende scomodo l'aggiornamento della base di dati dopo modifiche autorizzate.
  - Alcuni file rilevanti per la sicurezza, ad esempio file di registrazione di sistema, devono cambiare nel tempo e tripwire non fornisce un metodo per distinguere tra modifiche autorizzate e non.

## Controllo delle chiamate del sistema

- Metodo più raffinato e recente per il rilevamento di anomalie.
- Si controllano le chiamate del sistema per rilevare in tempo reale se un processo mostra un comportamento diverso da quello atteso.
- Sfrutta il fatto che un programma definisce implicitamente le sequenze di chiamate del sistema che può eseguire, secondo i possibili cammini d'esecuzione all'interno del programma stesso.

## Controllo delle chiamate del sistema

- Esempio: metodo proposto alla University of New Mexico per il sistema UNIX. In questo metodo, la sequenza usuale di chiamate per un programma viene generata eseguendo il programma su diversi dati in ingresso. Le sequenze generate vengono memorizzate e utilizzate per riempire una tabella che indica, per ogni chiamata del sistema, quali chiamate la possono seguire a distanza 1,2,..k. Durante successive esecuzioni del programma la sequenza di chiamate viene confrontata con la tabella per rilevare anomalie.



## Controllo delle chiamate del sistema

- I processi *root* sono tra i candidati ideali per questa tecnica, poiché hanno un insieme limitato di comportamenti e non cambiano spesso.
- Limite:
  - Un aggressore potrebbe costruire un attacco che non modifichi la sequenza delle chiamate del sistema tanto da allarme l'IDS.
- Il sistema IDS potrebbe decidere se un comportamento differente sia causa di un'intrusione o no, dal numero di discrepanze osservate.

## Crittografia

- È impossibile costruire una rete in cui è possibile fidarsi del fatto che gli indirizzi di sorgente e destinazione di un pacchetto identifichino con certezza chi lo ha inviato o chi lo riceve.
- **Crittografia:** usata per imporre i potenziali mittenti e destinatari di un messaggio.
- **Chiave:** codici segreti su cui si basa la crittografia. Vengono distribuite selettivamente ai calcolatori di una rete e si usano per gestire messaggi. Progettate in modo che sia computazionalmente impossibile derivarle a partire dai messaggi generati tramite esse.

## Crittografia

- La crittografia permette:
  - al destinatario di un messaggio di verificare che il messaggio sia stato creato da un calcolatore che possiede una certa chiave;
  - Al mittente di codificare il suo messaggio in modo che solo un calcolatore in possesso di una certa chiave possa decifrare il suo messaggio.

## Autenticazione

- Un algoritmo di autenticazione permette al destinatario di un messaggio di verificare che questo sia stato creato da un calcolatore che possiede una certa chiave. Comprende i seguenti elementi:
  - Un insieme  $K$  di chiavi;
  - Un insieme  $M$  di messaggi;
  - Un insieme  $A$  di autenticatori;
  - Una funzione  $S: K \rightarrow (M \rightarrow A)$ . Per ogni  $k \in K$ ,  $S(k)$  è una funzione che genera autenticatori sui messaggi. Sia  $S$  sia  $S(k)$  devono essere funzioni calcolabili in modo efficiente per ogni  $k$ .
  - Una funzione  $V: K \rightarrow (M \rightarrow \{true, false\})$ . Per ogni  $k \in K$ ,  $V(k)$  è una funzione che verifica gli autenticatori sui messaggi. Sia  $V$  sia  $V(k)$  devono essere funzioni calcolabili in modo efficiente per ogni  $k$ .

## Algoritmi di Autenticazione

- Proprietà fondamentale: dato un messaggio  $m$ , un calcolatore può generare un autenticatore  $a \in A$  t.c.  $V(k)(m, a) = true$  soltanto se esso possiede  $S(k)$ .
- Si dividono in due tipi:
  - **Codice di autenticazione di messaggi (MAC):** la conoscenza di  $V(k)$  o di  $S(k)$  è equivalente, uno si deriva dall'altro. È, quindi, importante proteggere  $V(k)$  quanto  $S(k)$ . Definisce  $S(k)(m) = f(k, m)$  dove  $f$  è una funzione non invertibile sul primo argomento (non derivabile dal valore della funzione) e resistente alle collisioni sul secondo argomento (è impossibile trovare  $m' \neq m$  t.c.  $f(k, m) = f(k, m')$ ). Un algoritmo di verifica è dato da  $V(k)(m, a) = (f(k, m) = a)$ .

## Algoritmi di Autenticazione

- **Firma digitale:** gli autenticatori prodotti sono chiamati firme digitali. È impossibile dal punto di vista computazionale derivare  $S(k)$  da  $V(k)$  dove  $V$  è una funzione non invertibile. Non è necessario che  $V(k)$  sia tenuta segreta e può essere liberamente distribuita.  $V(k)$  è detta *chiave pubblica*,  $S(k)$  *chiave privata*.
  - **RSA:** algoritmo basato sulla firma digitale. La chiave  $k$  è una coppia  $\langle d, N \rangle$ , dove  $N$  è il prodotto di due grandi numeri primi  $p$  e  $q$  scelti casualmente. L'algoritmo di firma è  $S((d, N))(m) = f(m)^e \bmod N$ , dove  $f$  è una funzione resistente alle collisioni. L'algoritmo di verifica è  $V((d, N))(m, a) = (a^d \bmod N = f(m))$  dove  $e$  soddisfa  $ed \bmod (p-1)(q-1) = 1$ . È impossibile, dal punto di vista computazionale, per un calcolatore che possiede  $V(\langle d, N \rangle)$  (cioè che possiede  $\langle e, N \rangle$  ma non  $d$ ) generare  $S(\langle d, N \rangle)$  cioè  $d$ .

## Cifratura

- Un algoritmo di cifratura permette al mittente di un messaggio di imporre che solo il calcolatore che possiede una certa chiave possa leggere il messaggio. Comprende i seguenti elementi:
  - Un insieme  $K$  di chiavi;
  - Un insieme  $M$  di messaggi;
  - Un insieme  $C$  di testi cifrati;
  - Una funzione  $E: K \rightarrow (M \rightarrow C)$ . Per ogni  $k \in K$ ,  $E(k)$  è una funzione che genera testi a partire da messaggi. Sia  $E$  sia  $E(k)$  devono essere funzioni calcolabili in modo efficiente per ogni  $k$ .
  - Una funzione  $D: K \rightarrow (C \rightarrow M)$ . Per ogni  $k \in K$ ,  $D(k)$  è una funzione che genera messaggi a partire da testi cifrati. Sia  $D$  sia  $D(k)$  devono essere funzioni calcolabili in modo efficiente per ogni  $k$ .

## Algoritmi di Cifratura

- Proprietà fondamentale: dato un testo cifrato  $c$ , un calcolatore può calcolare  $m$  in modo che  $E(k)(m)=c$  solo se possiede  $D(k)$ .
- Si dividono in due tipi:
  - **Algoritmo di cifratura simmetrica:**  $D(k)$  si può derivare da  $E(k)$  e viceversa. La segretezza di  $D(k)$  deve essere protetta tanto quanto quella di  $E(k)$ . Il più usato è DES(data encryption standard), non considerato sicuro per molte applicazioni, poiché le chiavi impiegate di 56 bit di lunghezza si possono sottoporre a una ricerca esaustiva usando risorse di calcolo moderatamente potenti. Si sta procedendo all'adozione di un nuovo algoritmo AES(advanced encryption standard).

## Algoritmi di Cifratura

- **Algoritmo di cifratura asimmetrica:** risulta impossibile derivare  $D(k)$  da  $E(k)$  e quindi  $E(k)$  non ha ragione d'essere tenuto segreto e si può distribuire;  $E(k)$  è detta *chiave pubblica* e  $D(k)$  *chiave privata*.  $k$  è una coppia  $\langle d, N \rangle$  dove  $N$  è il prodotto di due grandi numeri primi  $p$  e  $q$  scelti casualmente. L'algoritmo di cifratura è dato da  $E(\langle d, N \rangle)(m) = m^d \bmod N$  dove  $e, d$  ed  $N$  sono definiti come in precedenza. L'algoritmo di decifrazione è dato da  $D(\langle d, N \rangle)(c) = c^e \bmod N$ .

## Un esempio:SSL

- Protocollo crittografico che permette a due calcolatori di comunicare in modo sicuro, entrambi devono poter determinare mittente e destinatario dei messaggi diretti all'altro.
- Il più usato nella rete Internet.
- Avviato da un *client* allo scopo di comunicare in modo sicuro con un *server*.
- Prima dell'avvio, s'ipotizza che il server  $s$  abbia ottenuto un **certificato**,  $cert_s$ , da una terza entità: **autorità di certificazione**(certification authority\_CA).

## Un esempio:SSL

- **Certificato:** struttura di dati che contiene:
  - Vari attributi *attrs* del server, come il nome unico che lo contraddistingue e il suo nome comune DNS;
  - Un algoritmo di cifratura pubblico  $E(k_s)$  per il server;
  - Un intervallo di validità *interval* durante il quale il certificato si può considerare valido;
  - Una firma digitale a sulle informazioni da parte della CA,  $a = S(k_{ca})(\langle attrs, E(k_s), interval \rangle)$ .
- Prima dell'attivazione del protocollo, si presume che il client abbia ottenuto l'algoritmo pubblico di verifica  $V(k_{ca})$  per la specifica CA.

## Un esempio:SSL

- Quando un client  $c$  si connette al server  $s$ , gli invia un valore casuale  $n_c$  di 28 byte.
- Il server  $s$  risponde inviando al client un proprio valore casuale  $n_s$  oltre al suo certificato  $cert_s$ .
- Il client verifica che  $V(k_{ca})(\langle attrs, E(k_s), interval \rangle, a) = true$  e che la data corrente sia compresa nell'intervallo di validità *interval* del certificato.
- Se entrambe le verifiche sono soddisfatte:
  - Il client genera un valore casuale di 46 byte, *premaster secret* denotato con  $pms$ , invia al server il valore  $cpms = E(k_s)(pms)$ .
  - Il server ricostruisce  $pms = D(k_s)(cpms)$ .

## Un esempio:SSL

- Client e server sono in possesso di  $n_c$ ,  $n_s$  e pms, e possono autonomamente calcolare un valore condiviso di 48 byte *master secret* denotato da ms, con  $ms=f(n_c, n_s, pms)$ , dove f è una funzione non invertibile e resistente alle collisioni.
- Solo il server e client possono calcolare ms, poiché solo loro sono in possesso di pms.
- La dipendenza di ms da  $n_c$  e  $n_s$  assicura che ms sia un valore nuovo.

## Un esempio:SSL

- A partire dal master secret ms client e server calcolano le seguenti chiavi:
  - Una chiave di cifratura simmetrica  $k_{cs}^{crypt}$  per cifrare i messaggi dal client al server;
  - Una chiave di cifratura simmetrica  $k_{sc}^{crypt}$  per cifrare i messaggi dal server al client;
  - Una chiave di generazione di MAC  $k_{cs}^{MAC}$  per generare autenticatori sui messaggi dal client al server;
  - Una chiave di generazione di MAC  $k_{sc}^{MAC}$  per generare autenticatori sui messaggi dal server al client.

## Un esempio:SSL

- Un client per inviare un messaggio m al server:
  - Invia  $c = E(k_{cs}^{crypt})(m, S(k_{cs}^{MAC})(m))$
  - Il server riceve c e ricostruisce  $(m, a) = D(k_{cs}^{crypt})(c)$ , e accetta m se  $V(k_{cs}^{MAC})(m, a) = true$
- Un server per inviare un messaggio m al client:
  - Invia  $c = E(k_{sc}^{crypt})(m, S(k_{sc}^{MAC})(m))$
  - Il client ricostruisce  $(m, a) = D(k_{sc}^{crypt})(c)$ , e accetta m se  $V(k_{sc}^{MAC})(m, a) = true$

## Un esempio:SSL

- Questo protocollo permette:
  - Al server di limitare i ricevitori dei suoi messaggi al client che ha generato pms;
  - Al client di limitare i destinatari dei messaggi che invia e il mittente dei messaggi che accetta alla parte che conosce  $S(k_s)$ .
- Scopo del certificato *cert*: il campo *attrs* contiene informazioni che il client può usare per determinare l'identità del server con il quale sta comunicando.
- Per applicazioni nelle quali anche il server deve avere informazioni sul client, l'SSL ha un'operazione tramite la quale un client può inviare un certificato al server.

## Uso della crittografia

- I protocolli di rete sono organizzati in **strati**.
- Ciascun strato agisce come client rispetto allo strato sottostante.
- Quando un protocollo genera un messaggio per il protocollo analogo nel calcolatore di destinazione:
  - Consegna il suo messaggio al protocollo sottostante nella pila dei protocolli di rete;
  - Questo lo consegna al rispettivo protocollo in quella macchina

## Uso della crittografia

- Esempio: in una rete IP, TCP(protocollo dello strato di trasporto) agisce come un client dell'IP(protocollo dello strato di rete),
  - I pacchetti del TCP sono passati all'IP, lo strato sottostante;
  - In modo analogo IP li passa allo strato di collegamento dei dati sottostante affinché siano trasmessi attraverso la rete al corrispondente livello IP nel calcolatore di destinazione.
  - Questo livello IP, a sua volta, consegnerà i pacchetti TCP su quella macchina.

## Uso della crittografia

- La crittografia si può includere quasi in ogni livello.
- Il protocollo SSL fornisce sicurezza a livello trasporto.
- La sicurezza a livello rete, standardizzata IPSec, definisce formati dei pacchetti IP che permettono l'inserimento di autenticatori e la cifratura del contenuto dei pacchetti.

## Classificazione della sicurezza dei sistemi di calcolo

- Secondo una pubblicazione del 1985 del dipartimento della difesa degli USA esistono 4 livelli di sicurezza: A, B, C, D.
- **Piattaforma di calcolo fidata** (trusted computer base\_TCB): insieme dei sistemi di protezione di un sistema di calcolo che assicura l'impostazione dei criteri di sicurezza. Assicura solo che il sistema possa garantire il rispetto dei vincoli previsti dai criteri di sicurezza, ma non specifica il contenuto dei criteri stessi.

## Categoria di sicurezza: D

- Categoria di livello più basso, corrispondente ad una protezione minimale.
- Non ha sottocategorie e si assegna a sistemi che non soddisfano i criteri di nessuna delle altre categorie.
- Ad esempio, ad essa appartengono i sistemi operativi MS-DOS e Windows 3.1.

## Categoria di sicurezza: C

- Offre protezione discreta e permette di identificare e controllare gli utenti e le loro azioni per mezzo di strumenti di verifica. Suddivisa in:
  - **C1**: permette agli utenti di proteggere le loro informazioni private. È un ambiente in cui più utenti che collaborano accedono ai dati allo stesso livello di riservatezza. La maggior parte delle versioni UNIX appartiene a questa classe. La TCB protegge i dati d'autenticazione rendendoli inaccessibili agli utenti non autorizzati.
  - **C2**: insieme alle caratteristiche di C1 c'è la capacità di controllare gli accessi a un livello individuale. Alcune versioni UNIX particolarmente sicure appartengono a questa classe.

## Categoria di sicurezza: B

- Ha le caratteristiche di C2 ed ad ogni oggetto viene applicata un'etichetta che indica il livello di riservatezza. Suddiviso in:
  - **B1**: usa le etichette, divise in livelli, per prendere decisioni riguardanti il controllo degli accessi. Questi livelli sono gerarchici, un utente può avere accesso ad un oggetto con un'etichetta di riservatezza di livello pari o inferiore alla sua autorizzazione.
  - **B2**: estende l'uso di etichette di riservatezza a ogni risorsa del sistema. C'è protezione a livello di comunicazione, controlla eventuali sfruttamenti illegittimi di un canale protetto.
  - **B3**: permette la creazione di liste di controllo degli accessi che identificano gruppi o utenti ai quali non è consentito l'accesso ad un oggetto specificato.

## Categoria di sicurezza: A

- Grado più alto della classificazione. Suddivisa in:
  - **A1**: equivalente ad un sistema di classe B3, per funzioni e architettura ma si deve progettare e verificare impiegando metodi formali di definizioni e verifica, garantendo con alta probabilità che la TCB sia stata correttamente utilizzata.

## Un esempio:Windows NT

- Sistema operativo progettato per fornire garanzie di sicurezza.
- Il livello di sicurezza impiegato, normalmente è quello minimale ma l'amministratore del sistema può portare le garanzie di sicurezza al livello desiderato.
- Il modello di sicurezza si basa sul concetto di utente accreditato del sistema, *user account*.
- Permette la creazione di un numero arbitrario di utenti del sistema che si possono raggruppare in qualunque modo.
- L'accesso agli oggetti del sistema si può consentire o negare secondo i modi desiderati.

## Un esempio:Windows NT

- Il sistema identifica gli utenti con un identificatore di sicurezza unico.
- Quando un utente accede al sistema, si crea un **contrassegno d'accesso di sicurezza** che include:
  - Identificatore di sicurezza dell'utente;
  - Identificatori di sicurezza per tutti i gruppi dei quali l'utente è membro;
  - Elenco di tutti i permessi di cui l'utente gode.
- Il sistema usa l'idea di soggetto per impedire che i programmi di un utente ottengano modi d'accesso al sistema meno restrittivi di quelli dell'utente stesso.

## Un esempio:Windows NT

- **Soggetto**(subject):usato per identificare e gestire i permessi relativi ad ogni programma eseguito per conto di un utente, composto dal contrassegno d'accesso e dal programma che agisce per conto dell'utente.
- Dato che il sistema Windows NT si basa sul modello client-server, per controllare gli accessi si usano 2 classi di soggetti:
  - **Soggetto semplice**:programma d'applicazione che un utente esegue dopo aver ottenuto l'accesso al sistema. Si assegna in contesto di sicurezza definito secondo il contrassegno d'accesso dell'utente.
  - **Soggetto server**:processo realizzato come un server protetto che usa il contesto di sicurezza del client quando agisce per suo conto.

## Un esempio:Windows NT

- Gli attributi di sicurezza di un oggetto di questo sistema sono descritti da un **descrittore di sicurezza** che contiene:
  - Identificatore di sicurezza del proprietario dell'oggetto;
  - Lista di controllo discrezionale degli accessi che stabilisce quali utenti o gruppi abbiano o no possibilità d'accesso;
  - Lista di sistema di controllo degli accessi che controlla quali messaggi di verifica saranno generati.

## Un esempio:Windows NT

- Questo sistema classifica gli oggetti come:
  - **Contenitori**: esempio le directory, possono contenere in senso logico altri oggetti. Quando si crea un oggetto all'interno di un oggetto contenitore, il nuovo oggetto eredita i permessi dell'oggetto genitore;
  - **Non contenitori**: non ereditano nessun altro permesso.