Sicurezza dei Sistemi e delle Reti

File di Nali

Indice

Capitolo 1

Politiche di Sicurezza

1.1 Definizione Politica di sicurezza

La *gestione della sicurezza* è un *processo formale* per rispondere alle domande:

- quali sono i beni da proteggere
- quali sono le possibili minacce
- come si possono contrastare le minacce

Questo processo ha natura iterativa, ed è contenuto nella ISO 31000; in questa norma viene descritto un *modello per la gestione della sicurezza delle informazioni* che comprende le seguenti fasi:

- **Plan:**
  - stabilire politiche, processi e procedure di sicurezza
  - eseguire la valutazione del rischio
  - sviluppare un piano di trattamento del rischio
- **Do:**
  - implementare il piano di trattamento del rischio
- **Check:**
  - monitorare e mantenere il piano di trattamento del rischio
- **Act:**
  - mantenere e migliorare la gestione dei rischi
  - risposta ad incidenti, analisi di vulnerabilità e riprendere il ciclo iterativamente

1.1.1 Definizioni

- **Rischio:** esprime la possibilità che un attacco causi danni ad una organizzazione
- **Risorsa:** tutto ciò che necessita di essere protetto
  - *hw*
  - *sw*
  - *reputazione*

  La valutazione di una risorsa viene fatta in base:
  - ai costi da sostenere per sostituire la risorsa nel caso non sia più disponibile
  - perdita di incassi in caso di attacco
- **Vulnerabilità:** punti deboli che possono essere sfruttati per causare danni al sistema; possono essere classificate come:
  - critico
  - moderato
  - basso

Definizione per domanda esame: Una politica di sicurezza dei sistemi e delle reti è l'insieme strutturato di regole, requisiti e comportamenti che un'organizzazione definisce affinché i propri sistemi informatici e le proprie infrastrutture di rete applichino in modo coerente misure di protezione contro accessi non autorizzati, abusi, perdite di dati e altri rischi. Essa rappresenta ciò che il sistema deve far rispettare, automatizzando e facendo valere controlli come l'autenticazione, l'autorizzazione, il monitoraggio delle attività e la gestione delle vulnerabilità, allo scopo di garantire la riservatezza, l'integrità e la disponibilità delle informazioni, secondo gli obiettivi stabiliti dall'organizzazione stessa.

1.2 Domanda MAC, DAC e RBAC

definire l'utilizzo delle politiche di sicurezza basate su MAC, DAC e RBAC

Gli approcci DAC,MAC e RBAC fanno parte delle politiche di controllo degli accessi e forniscono approcci diversi a seconda del contesto di applicazione

1.2.1 DAC (Discretionary Access Control):

Il controllo dell'accesso viene fatto sull' **identità del soggetto richiedente** e delle **regole di accesso**. Definito *discrezionale* poichè un'entità potrebbe avere i privilegi di accessi che le permettono, a sua volta, di concedere l'accesso ad un'altra entità.
Si può rappresentare mediante una matrice, dove:

- le colonne rappresentano i soggetti
- le righe gli oggetti
- ogni cella specifica i diritti di accesso di quel soggetto a quel determinato oggetto

1.2.2 MAC (Mandatory Access Control):

La politica di controllo degli accesssi MAC, o Mandatory Access Controll si basa sul confronto tra **etichette di sicurezza** che indicano quanto sono sensibili le risorse e **autorizzazioni di sicurezza** che indicano quali entità del sistema sono idonee ad accedere a quali risorse.
Questa politica è definita *mandatoria* poichè un'entità che possiede l'accesso a una risorsa non può estendere il permesso di acesso a un'altra entità, può farlo solo l'amministratore di sistema.
I sistemi MAC si dividono in:

- **Multilevel security systems:** consiste in una struttura verticale di sistemi di sicurezza, agli utent viene assegnato un livello e possono accedere solo a risorse con un livello uguale o inferiore
- **Multilateral security systems:** l'accesso viene assegnato in base a segmenti che formano gruppi costituiti da livelli di sicurezza e parole in codice
  - si ottiene una struttura orizzontale, che contiene livelli di sicurezza verticali aggiuntivi

Vantaggi:

- molto sicuro, a prova di manomissione
- gli utenti non possono fare modifiche
- controllo automatizzato
- i dati non possono essere modificati senza apposita autorizzazione

Svantaggi:

- richiede una pianificazione dettagliata e un lavoro amministrativo
- controllo e aggiornamento dei dati di accesso
- manutenzione per aggiunta di nuovi dati o utenti e relative modifiche (elevato carico di lavoro per l'amministratore)

1.2.3 RBAC (Role Based Access Control):

Introduce il concetto di **ruolo**, ovvero una funzione che può essere assiciata a uno o più utenti (gli utenti possono avere più ruoli) e una **sessione**, ovvero una mappatura tra utente e un sottoinsieme di ruoli a cui è assegnato.
I ruoli di un utente possono fornire o meno accesso a determinate risorse.
Quattro tipi di entità:

- **Utente:** una persona che ha accesso al sistema, ogni individuo ha un ID associato
- **Ruolo:** funzione lavorativa all'interno dell'organizzazione
- **Autorizzazione:** approvazione di una modalità di accesso ad uno o più oggetti
- **Sessione:** mappatura tra utente e un sottoinsieme dei ruoli a cui è assegnato

1.3 Fare cenni sull'utilizzo di tali politiche nei Sistemi Operativi moderni

1.3.1 Unix security model

In Linux ci sono tre entità da considerare:

- **Soggetto:** può essere un utente o un processo
- **Oggetto:** file, cartelle, ...
- **Operazioni consentite:** lettura, scrittura, esecuzione

In Unix, ogni utente ha associato un id univoco, detto **UID**; può appartenere a gruppi di utenti, anch'essi identificati da un id univoco detto **GID**. Tutti gli utenti appartenenti ad un gruppo possono condividere tra loro oggetti.
Ad ogni file è assegnato un unico utente proprietario e un unico gruppo proprietario. L'autorizzazione viene concessa mediante una ACL che identifica le operazioni che i soggetti possono fare.

Processi in Linux

Ogni processo è isolato dagli altri e non possono accedere alla memoria altrui. Ogni processo viene eseguito con le autorizzazione dell'UID dell'utente che lo sta eseguendo.
Nel momento della creazione, ad ogni processo sono assegnati tre ID (inizialmente tutti uguali all'UID):

- **Effective UID:** determina le autorizzazioni per il processo

- **Real UID:** determina l'utente che ha avviato il processo
- **Saved UID:** EUID prima di eventuali modifiche

L'utente *root* può cambiare EUID/RUID/SUID a valori arbitrari; utenti non privilegiati possono cambiare EUID solo a RUID o SUID

Unix file access control

Le modifiche agli ID sono apportate mediante i comandi *setUID* e *setGID*; questa modifica permette ai programmi non privilegiati di accedere a risorse generalmente non accessibili.
Le directory possono aver impostato uno ***sticky bit***: specifica che solo il proprietario di un file nella cartella può apportare una modifica a quel file
Il ***superuser*** è esente dalle consuete restrizioni di controllo degli accessi, ha accesso a tutto il sistema.

1.3.2 Windows security architecture

L'architettura di sicurezza di Windows è basata su più entità:

- ***Security Reference Model (SRM):*** componente che in modalità kernel esegue controlli delle autorizzazioni e manipola i privilegi degli utenti
- ***Local Security Authority (LSA):*** risiede in un processo utente, è responsabile dell'applicazione della politica di sicurezza locale, tra cui:
  - criteri per le password, come complessità e tempi di scadenza
  - politica di controllo → specifica quali operazioni su quali oggetti vadano controllate
  - impostazioni dei privilegi
- ***Security Account Manager (SAM):*** è un database che archivia i dati degli account e le informazioni di sicurezza su entità locali e gruppi
- ***Active Directory (AD):*** implementa il protocollo LDAP (*Lightweight Directory Access Protocol*)

Windows security model

Windows ha un complesso sistema di controllo dell'accesso; ogni oggetto ha ACL per permettere autorizzazioni granulari ad utenti e/o gruppi di utenti.

Security descriptor

Ogni oggetto ha un *security descriptor* che contiene:

- ***security identifier* (SID)** per il possessore e il gruppo primario dell'oggetto (SID è associato univocamente ad ogni utente)

- ***discretionary ACL* (DACL):** diritti di accesso per gli utenti e i gruppi
- ***system ACL* (SACL):** tipi di accesso che generano log

Ad ogni processo viene inoltre associato un insieme di **token**, che prende il nome di ***security context***. Nel momento in cui un processo vuole accedere ad un oggetto, presenta il suo insieme di token e il sistema controlla se il security context abbia o meno accesso a tale risorsa in base al *security descriptor* dell'oggetto.

Capitolo 2

Set-UID

2.1 Si definisca il funzionamento e possibili implicazioni per la sicurezza

Il meccanismo SETUID in Unix/Linux consente a un programma di essere eseguito con i privilegi dell'utente proprietario del file, anziché con quelli dell'utente che lo esegue. È spesso usato per permettere a utenti comuni di svolgere operazioni che richiedono privilegi elevati, come accedere a risorse di sistema. Tuttavia, questa funzionalità ha implicazioni di sicurezza importanti: se un programma con SETUID è vulnerabile o mal scritto, può essere sfruttato per ottenere privilegi superiori, fino a diventare root. Per questo motivo, l'uso di SETUID è considerato delicato e va limitato a programmi essenziali e attentamente controllati.

2.1.1 Esempio di utilizzo di SETUID

Un esempio semplice di utilizzo del SetUID è un programma che consente a un utente normale di scrivere in un file accessibile solo a root. Normalmente, un utente senza privilegi non potrebbe modificare file di sistema, ma se il programma è di proprietà di root e ha il bit SetUID attivo, verrà eseguito con i privilegi di root anche se l'utente che lo lancia non li possiede. In questo modo, il programma può svolgere un'operazione riservata, come scrivere in un file protetto, senza dover dare all'utente l'accesso diretto a quei privilegi. Questo dimostra come SetUID possa essere usato per concedere temporaneamente diritti elevati in modo controllato.

2.2 Ogni processo Unix process è associato con un real user ID (RUID) e un effective user ID (EUID). Spiegare la logica ed importanza del setuid.

Il **Real User ID** (RUID) , determina l'utente che ha avviato il processo, mentre l'Effective User ID (EUID), determina le autorizzazioni per il processo. Il **set user ID** (setuid) ha due funzioni principali: permette a un utente di eseguire un file o un processo con i privilegi dell'utente proprietario, oltre che ai suoi. Inoltre consente a programmi privilegiati di accedere a risorse generalmente non accessibili. Questo può comportare problematiche su sistemi, se impostato in maniera non corretta, oltre che essere una vulnerabilità che può essere sfruttata per attacchi.

2.2.1 Quale tra RUID e EUID viene utilizzato dal sistema operativo per determinare se un processo ha il diritto di accedere a una risorsa o meno?

Il sistema operativo verifica il EUID per determinare se un processo ha diritto ad accedere a una risorsa o meno.

2.2.2 Nella maggior parte delle versioni di Unix, la famiglia di funzioni setuid consente di impostare l'EUID di un processo sul suo RUID. Riesci a pensare a un motivo per farlo in un processo in esecuzione come root (EUID=0)?

Quando si deve andare a modificare la password di un utente, si deve andare a modificare un file chiamato shadow: questo file è modificabile solo dall'utente root, perciò lanciando il comando passwd, il processo setta come ID quello di root, per permettere la modifica di questo file.

2.3 Esercizi con codice

2.3.1 Charlie ha trovato nel computer un file con i seguenti permessi:

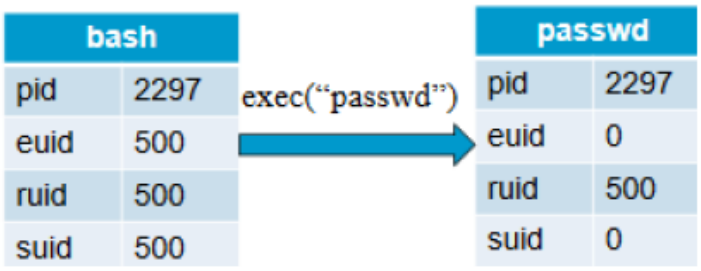
-rwsrwxrwx 1 root root 186 Oct 31 23:42 mioexe
Spiegare la pericolosità del file mioexe.
Il file mioexe ha tutti i permessi settati, quindi sia l'owner, che il gruppo a cui appartiene, che tutti gli altri utenti, possono leggere, scrivere ed eseguire il file. Inoltre è impostato il SUID, un permesso speciale che indica al kernel di lanciare

i comandi con i privilegi dell'owner del file: in questo caso l'owner è root, perciò si potrebbe modificare il file mioexe per lanciare dei comandi dannosi come root.

2.3.2 Come esempio si consideri /usr/bin/passwd : spiegare se tale comando ha il setuid settato e perché

Il comando '/usr/bin/passwd' ha il bit SETUID attivo perché permette a un utente normale di modificare la propria password, operazione che richiede l'accesso al file di sistema '/etc/shadow', il quale è leggibile e scrivibile solo dall'utente root.
Grazie al SETUID, quando un utente esegue 'passwd', il processo assume temporaneamente i privilegi di root (proprietario del file), così da poter aggiornare in sicurezza la password nel file protetto. Questo è un classico esempio di uso legittimo e necessario del SETUID, ma proprio per la sua sensibilità, il comando 'passwd' deve essere scritto in modo estremamente sicuro per evitare rischi di escalation dei privilegi.

2.3.3 Commentare l'esecuzione di passwd dal seguente processo



Lanciando il comando passwd, si indica al sistema operativo di voler cambiare la propria password utente. Questa password è presente in un file chiamato shadow, che è modificabile solo dall'utente root. Lanciando quindi il comando passwd, si va a indicare che il EUID è uguale a 0, cioè l'utente root, fornendo temporaneamente all'utente con RUID 500 di modificare la propria password.

2.3.4 Si consideri l'utente bob che appartiene solo al gruppo users. Per ognuno dei seguenti file, discutere se bob è capace di eseguire il file, se no spiegare perchè, se sì evidenziare i bit EUID e RUID dei corrispondenti processi.

-rwsr–r– 1 root root 213 Oct 12 11:10 file1.bin

Si, può eseguire il file perché, anche se potrebbe solo leggerlo, è impostato il setUID (s) e quindi può eseguirlo con i privilegi del owner (cioè di root). RUID è quello di Bob, mentre EUID è 0, cioè quello di root

-rwxr-xr– 1 alice users 134 Oct 12 11:11 file2.bin

Si, può eseguire il file perché fa parte del gruppo del owner del file (alice, che è del gruppo users). Non potrà però scriverci, in quanto non dispone del privilegio (r-x). In questo caso RUID e EUID corrispondono e sono quelli di Bob.

-rwsr-xr– 1 alice users 186 Oct 12 11:12 file3.bin

Si, può eseguire il file perché fa parte del gruppo del owner del file (alice, che è del gruppo users). A differenza di sopra però, è impostato il SETUID, perciò Bob lancia il comando come se fosse Alice. RUID = Bob, EUID = Alice

-r–rwxr– 1 bob users 113 Oct 12 11:13 file4.bin

No, in questo caso l'owner del file è Bob, ma non ha i privilegi per poter eseguire il file. I membri del suo gruppo, users invece possono leggere, scrivere ed eseguirlo, il che è un po' un controsenso.

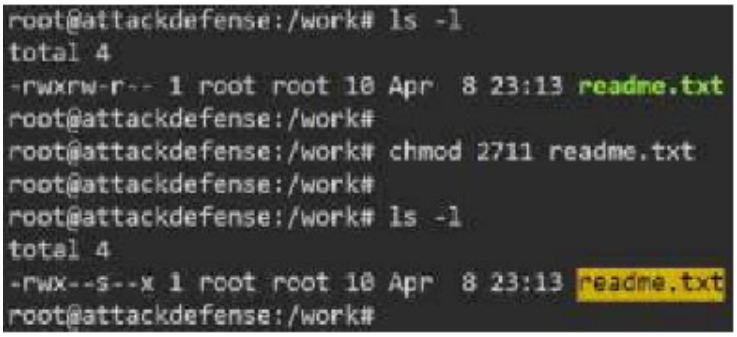
2.3.5 Si immagini che un attaccante trovi una shell di root su un terminale e digiti le seguenti righe di codice, quali conseguenze potrebbe avere?

% cp / bin/sh /tmp/ break-acct

% chmod 4755 /tmp/break-acct

L'attaccante, con il primo comando, copia la shell in una cartella temporanea. Con il secondo comando assegna il bit SetUID (il 4 davanti a 755), in modo che, quando un altro utente esegue quel file, la shell venga avviata con i privilegi del proprietario, cioè root. Questo permette a qualsiasi utente di ottenere una shell come root, con la possibilità di fare gravi danni al sistema. La parte 755 dei permessi indica che il proprietario può leggere, scrivere ed eseguire il file, mentre gli altri utenti (gruppo e altri) possono leggere ed eseguire (ma non scrivere).

2.3.6 Descrivere cosa succede al file readme.txt dopo l'esecuzione dei comandi visualizzati in figura



Il file readme.txt ha subito un comando per modificarne i permessi (chmod) e gli è stato settato il bit "set Group ID" (2), mentre l'owner può leggere, scrivere ed eseguire il file (7). I membri del suo gruppo e tutti gli altri utenti invece possono solo eseguire il file. Readme.txt diventa quindi un file eseguibile chi lo esegue, lo fa come se fosse nel gruppo del owner, piuttosto che nel suo.

2.3.7 Programma catall.c

b. Si consideri il seguente codice **catall.c**:

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>

int main(int argc, char *argv[]){
   char *pCatStr = "/bin/cat";

   char *pCmd = malloc(strlen(pCatStr) + strlen(argv[1]) + 2);
   sprintf(pCmd, "%s %s", pCatStr, argv[1]);
   system(pCmd);

   return 0;
}
```

Si spieghi in cosa consiste l'attacco effettuato con le seguenti istruzioni:

```
$ gcc catall.c -o catall
$ sudo chown root catall
$ sudo chmod 4755 catall
$ ./catall "aa;/bin/sh"
# whoami
root
```

In questo caso, il programma 'catall' viene compilato e poi configurato con il bit SetUID, rendendolo eseguibile con i privilegi di root. Il programma prende un argomento da linea di comando e lo concatena al comando '/bin/cat', eseguendolo tramite 'system()'. Il problema di sicurezza nasce dal fatto che l'argomento

dell'utente viene inserito direttamente in una stringa di comando senza alcun controllo.

Quando "attaccante lancia './catall "aa;/bin/sh"', il programma esegue effettivamente due comandi: prima 'cat aa', poi, a causa del punto e virgola, viene eseguito anche '/bin/sh'. Poiché il programma ha i privilegi di root, anche la shell ottenuta è con privilegi di root. Questo permette all'attaccante di accedere a una shell di amministratore, aggirando le normali restrizioni. L'istruzione 'whoami' confermerà infatti che l'utente è root. Questo è un esempio di escalation dei privilegi dovuto a un uso non sicuro della funzione 'system()' in un programma SetUID.

2.3.8 Si consideri il codice seguente uid.c compilato dallo user con id 1000, Completare sotto indicando ID corretto al posto di ???

b. Si consideri il codice seguente uid.c compilato dallo user con id 1000:

```
1.#include <stdio.h>
2. #include <unistd.h>
3. #include <stdlib.h>
4.
5. int main(void){
6.
7.     int val;
8.     printf("The real user ID is %d\n", getuid());
9.     printf("The effective user ID is %d\n", geteuid());    return 0;}
```

Completare sotto indicando ID corretto al posto di ???

```
1. gcc uid.c -o uid
2. sudo chown root.root uid
3. ls -la uid
4. -rwxr-xr-x 1 root root 16712 Jul 10 11:59 uid
5. ./uid
6. The real user ID is ??
7. The effective user ID is ??
8. sudo chmod 4755 uid
9. ls -la uid
10.  -rwsr-xr-x 1 root root 16712 Jul 10 11:59 uid
11.  ./uid
12.  The real user ID is ??
13.  The effective user ID is ??
```

- **riga 6:** 1000
- **riga 7:** 1000
- **riga 12:** 1000
- **riga 13:** 0 (viene impostato il bit SetUID con chmod 4755, ossia privilegi del proprietario)

2.3.9 File di bob

Si consideri il file `vedi` che contiene il codice eseguibile di un programma che lista il contenuto di file testuali (il comando è quindi `$vedi <file>`).
Il file `vedi` è dell'utente Bob, con UID=1700 e GID=5000, ed ha le protezioni 550. Per vedere le caratteristiche del file `vedi`, l'utente Bob effettua il comando: `$ls -l > dati`. Facendo `$ls -l dati`, Bob vede che il file dati è ovviamente di Bob ed ha le protezioni 604.

Argomentare le seguenti risposte:

i. Può Bob vedere il contenuto del file dati con il comando `$vedi dati`?
ii. Cosa succede se Charlie del gruppo 5000 scrive il comando `$vedi dati`?
iii. Cosa succede se Charlie del gruppo 5000 scrive il comando `$vedi dati` dopo che Bob setta il bit SetUID del file `vedi`?

1. Bob può usare vedi dati perché è il proprietario di entrambi i file e ha permessi di lettura sul file dati. Anche se vedi ha permessi 550 (r-x per owner e gruppo), Bob può eseguirlo e il programma, eseguito con i suoi privilegi, può aprire e leggere dati senza problemi.

2. Charlie, pur essendo nel gruppo 5000, non può leggere dati con vedi dati perché il file dati ha permessi 604, quindi il gruppo non ha permessi di lettura. Il programma vedi verrà eseguito con i privilegi di Charlie (poiché il bit SetUID non è attivo), quindi l'accesso al file sarà negato.

3. Se Bob imposta il bit SetUID su vedi, il programma eseguito da Charlie gira con l'effective UID di Bob (1700). In questo modo, anche se Charlie non ha permessi diretti su dati, può leggerlo perché il processo ha i privilegi di Bob.

Capitolo 3

Malware

Una definizione informale per malware potrebbe essere quella di *programma malevolo*, solitamente inserito di nascosto in un sistema, che ha lo scopo di compromettere la **riservatezza**, l'**integrità** o la **disponibilità** del sistema stesso.
I malware sono classificati in base a:

- **Propagazione:** software, rete, social engeneering
- **Azioni sui dati colpiti:** corruzione, furto, crittografia
- **Attack kit:** strumenti già pronti per attaccare
- **Attori e/o motivazioni dell'attacco**

3.1 Differenza tra due tipologie di malware (virus e worm) + esempio

Sono stati chiesti questi due ma per sicurezza aggiungo anche gli altri

3.1.1 Trojan

È un programma che ha un effetto evidente e atteso dall'utente, che ha però anche un effetto **nascosto** che viola le politiche di sicurezza e che viene condotto senza l'autorizzazione dell'utente

3.1.2 Virus

È un codice che può replicarsi modificando altri file o programmi per inserire codice in grado di replicarsi a sua volta; questa **proprietà di replicazione** è ciò distingue i virus dagli altri tipi di malware. Non svolge nessuna azione evidente, ma cerca di rimanere nell'ombra.

La replica richiede un certo tipo di assistenza da parte dell'utente, come ad esempio cliccare su un allegato.
Un virus è composto da tre parti:

- **Meccanismo di infezione**
- **Trigger:** evento che determina quando il payload viene attivato
- **Payload:** cosa fa il virus (oltre a diffondersi)

I virus attraversano quattro fasi:

1. **Dormiente:** il virus è inattivo in attesa di essere attivato
2. **Scatenante:** il virus viene attivato
3. **Propagazione:** il virus inserisce una copia di sé stesso in certe parti del sistema; ogni programma infetto conterrà ora un altro virus che entrerà a sua volta in fase di propagazione
4. **Esecutiva:** la funzione viene eseguita

Vettori di infezione

I principali vettori di infezione sono:

- **Boot sector** di dispositivi esterni; il codice è inserito nel boot sector e viene eseguito in fase di avvio
- **Eseguibili**
- **File macro:** il virus si attacca ai documenti per propagarsi

Esempio noto in letteratura:

Un esempio di virus può essere considerato il compression virus, che va a comprimere lo spazio occupato da un programma, per inserire un codice malevolo: così facendo la dimensione di un file è la stessa, andando a baypassare i controlli di un antivirus.

3.1.3 Worm

I worm sono programmi *stand alone* (a differenza dei virus che devono essere attivati da un qualche evento) in grado di replicarsi.
Le fasi di esecuzione sono:

- **Probing:** cerca informazioni sulla macchina
- **Expoloitation:** sfrutta le informazioni raccolte per trovare vulnerabilità
- **Replicazione**
- **Attacco** (payload)

Esempio noto in litteratura:

Esempio di worm della famiglia Nimda, che prese di mira i sistemi operativi Microsoft Windows.

3.1.4 Drive-by-download

Sfruttano **vulnerabilità del browser** per installare codice malevolo ad insaputa dell'utente nel momento in cui visita la pagina web dell'attaccante.

3.1.5 Clickjacking

L'attaccante intercetta un *click* dell'utente per costringerlo a fare delle cose contro la sua volontà.

3.1.6 Zombie e botnet

Lo *zombie* è una singola macchina, mentre la *botnet* è un insieme di macchine zombie controllate da una singola entità; vengono usate per fare DDoS, phishing, spamming, ...

3.1.7 Rootkit

È un insieme di programmi installati su un sistema per mantenere l'accesso ad un sistema, ad esempio, con privilegi di amministratore, nascondendo le prove della sua presenza e aggirando i meccanismi di controllo.
Permettono di fare attacchi anche con scarse conoscenze tecniche.

3.1.8 Scareware

Software che hanno lo scopo di diffondere shock, ansia e/o la percezione di una minaccia; sono un attacco di *social engeneering*.

3.1.9 Ransomware

Software che tiene in ostaggio il sistema per richiedere un riscatto all'utente, spesso tramite cifratura.

3.1.10 Vulnerabilità zero-day

Si intende un'exploit non ancora nota e che non ha quindi una contromisura.

3.1.11 Spear phishing

Viene **studiato nel dettaglio il bersaglio**, in modo tale da fare del phishing più mirato ed efficace.

3.1.12 Spyware

Malware che raccoglie piccole informazioni alla volta sugli utenti a loro insaputa, come ad esempio un *keylogger*.

3.1.13 APT - Advanced Persistent Threats

- ***Advanced:*** è un'applicazione con un ampia varietà di tecnologie di intrusione e malware
- ***Persistent:*** attacchi per un periodo di tempo prolungato verso il target
- ***Target:*** target selezionati in modo accurato

Le fasi principali di un attacco tramite APT sono:

- **Ricognizione:** si sceglie una vittima e la si studia
- **Weaponization:** si mette un trojan che permette accesso remoto in un payload consegnabile (email, USB, web)
- **Sfruttamento:** il codice malevolo viene attivato per portare a termine il suo scopo

3.2 Differenza tra virus metamorfico e polimorfico

3.2.1 Classificazione dei virus

È possibile classificare i virus in base alle **tecniche usate per superare i controlli di sistema:**

- **Cifratura del virus:** crea una chiave per *"crittografarsi"*; quando viene chiamato un programma infetto, con tale chiave viene decifrato il virus. Per evitare pattern di bit, durante la propagazione la chiave viene cambiata
- **Stealth virus:** si nasconde dal rilevamento da parte dell'antivirus, tramite mutazione o compressione del codice
- **Polymorphic virus:** durante la replica crea copie che svolgono la stessa funzione ma che hanno pattern di bit diversi
- **Metamorphic virus:** si riscrive completamente ad ogni iterazione per aumentare la difficoltà di rilevamento
- **Compression virus:** comprimono il file eseguibile in modo che la versione infetta abbia la stessa dimensione di quella originale

3.3 Discutere se le seguenti tecniche sono meccanismi di rilevamento utili rispettivamente per i virus polimorfici e metamorfici:

3.3.1 Static pattern matching

Utile per **virus polimorfici** poichè cambiano solo la parte cifrata del loro codice a ogni infezione, ma mantengono invariata la struttura del decryptor, che può essere riconosciuto con tecniche di pattern matching statico, perché rimane simile o identico tra le varianti.
Al contrario dei virus metamorfici che modificano completamente il proprio codice a ogni infezione, incluso il decryptor o qualsiasi parte riconoscibile.

3.3.2 Pattern matching during emulation

Utile per **virus metamorfici** poichè nonostante essi cambino completamente aspetto a ogni infezione, durante l'emulazione (cioè durante l'esecuzione simulata del codice in un ambiente controllato), il virus esegue le stesse azioni fondamentali, indipendentemente dalla sua forma.
La tecnica di pattern matching during emulation permette quindi di osservare il comportamento reale del codice eseguito (come accessi a file di sistema, chiamate di sistema sospette, o autoriproduzione).
Al contrario, emulare un virus polimorfico è inefficiente, perché richiede tempo e risorse per ottenere un'informazione che è già ricavabile staticamente.

3.3.3 Suspicious behaviour detection

Utile per **virus polimorfici** poichè il loro comportamento rimane molto prevedibile:

- Decriptano il corpo
- Si copiano in altri file
- Cercano persistenza

Per quanto riguarda i virus metamorfici possono cercare di mascherare il loro comportamento, soprattutto quelli più avanzati possono simulare comportamenti legittimi.
Questo li rende più difficili da intercettare solo con behaviour detection

Capitolo 4

Autenticazione

4.1 Discutere le problematiche di autenticazione su Web e in dettaglio uno schema challenge-response

L'autenticazione in generale, e in particolare quella web, consiste principalmente in un client che fa una richiesta a un server e un server che fornisce una risposta. Per fare in modo però che il server non inivii informazioni a client malevoli, che possono attaccare tramite attacchi di spoofing, fingendosi un client legittimo (o anche fingendosi un server legittimo), oppure con replay attack, ciò inivio di pacchetti già inviati da un client legittimo(tramite sniffing sulla rete), vengono introdotte misure di sicurezza più avanzate.
Client e server condividono informazioni segrete, che possono andare da una password a una chiave di crittografia(secret): un client che vuole accedere a un servizio web su un server, deve dimostrare di essere chi dichiara di essere. Il server presenta quindi una stringa (**challenge**) e il client, tramite il secret, può fornire la prova di identificazione richiesta e riesce ad accedere (**response**).
Questo schema fornisce segretezza, tramite uso di password o chiavi, e anche freschezza, nella misura in cui la challenge viene modificata a ogni richiesta, così che non si possa sfruttare una risposta già fornita con un replay attack.

Capitolo 5

Attacchi TCP

5.1 ARP spoofing attack, quali sono le possibili conseguenze ed eventuali contromisure

Il protocollo ARP (Address Resolution Protocol) si occupa di connettere un indirizzo fisico (MAC address) con un indirizzo logico(IP address). Per farlo ogni nodo all'interno della rete invia dei messaggi in broadcast per segnalare la sua presenza, così che gli altri nodi possano andare a popolare e/o modificare una tabella interna chiamata *ARP cache table*
Viene fatta un'assunzione di trust nella LAN, dato che le richieste non vengono tracciate, gli annunci non sono autenticati, le macchine si fidano l'una dell'altra. In questo modo un possibile attaccante può utilizzare il meccanismo automatico di aggiornamento della ARP cache table (che si aggiorna anche se non ha inviato alcuna richiesta) per fingersi un altro dispositivo.
Conseguenze: L'attaccante si impone tra due dispositivi permettendogli di intercettare il traffico (sniffing), modificarlo (per esempio per iniettare un malware) o bloccarlo (DoS).
Contromisure: Alcune possibili contromisure possono essere:

- ARP statici: si configurano manualmente le associazione degli indirizzi IP e MAC, ma non è pratico in reti dinamiche o di grandi dimensioni
- Crittografia del traffico: anche se l'attaccante intercetta il traffico, non può leggerne il contenuto.
- Strumenti di monitoraggio

5.2 Attacco TCP SYN flood e le sue contromisure + SYN-Cookie

L'attacco TCP SYN Flood è una forma di attacco DoS (Denial of Service) che mira a sovraccaricare un server esaurendone le risorse disponibili per le con-

nessioni, impedendo così agli utenti legittimi di accedere ai servizi. L'idea è di sfruttare le vulnerabilità legate alla negoziazione di una comunicazione tra client e server durante il three-way handshake, l'attaccante **continua a iniviare richieste SYN al server** senza poi rispondere ai SYN/ACK ricevuti in risposta, il server dovrà quindi attendere andando a saturare il TCB (Trasmission Control Block), impedendo le connessioni legittime.
Contromisure: ampliare la memoria del TCB in modo che possa ricevere più richieste, ridurre i timer prima che una richiesta SYN venga cancellata dalla memoria o utilizzo dei SYN-Cookie.
SYN-Cookie: Tenica per contrastare l'attacco SYN Flood, l'utilizzo di cookie permette a una sessione di restare attiva anche se la coda SYN è stata saturata da un attacco. Quando viene attivata una sessione con un three-way handshake, il server risponde a al pacchetto SYN del client con un SYN-ACK più il valore del cookie. Quando il client risponde con un ACK, il server decodifica il cookie per verificare la validità della richiesta e, solo allora, alloca le risorse necessarie per stabilire la connessione.
Questo approccio consente al server di gestire un numero elevato di richieste SYN senza esaurire le risorse, poiché non mantiene stato per le connessioni incomplete.

5.3 TCP hijacking attack + codice

```
▶ Internet Protocol Version 4, Src: 10.0.2.69, Dst: 10.0.2.68
▼ Transmission Control Protocol, Src Port: 23, Dst Port: 45634 ...
    Source Port: 23
    Destination Port: 45634
    [TCP Segment Len: 24]                ← Data length
    Sequence number: 2737422009          ← Sequence #
    [Next sequence number: 2737422033]   ← Next sequence #
    Acknowledgment number: 718532383
    Header Length: 32 bytes
    Flags: 0x018 (PSH, ACK)
```

5.3.1 TCP hijacking attack ed esempio rispetto al codice

Il TCP hijacking è un attacco in cui un attaccante prende il controllo di una sessione TCP già avviata tra due host, senza che uno dei due se ne accorga. L'attaccante invia pacchetti spoofati, fingendosi uno degli endpoint, sfruttando il fatto che TCP si basa su numeri di sequenza e acknowledgment per mantenere la connessione affidabile. L'obiettivo dell'attacco è quello di iniettare dati o comandi nella sessione o interromperla/alterarla.
Nel nostro scenario ipotetico, l'attaccante intercetta un pacchetto proveniente dal server verso il client in una connessione Telnet. Anche se non conosce l'intero stato della sessione, dal pacchetto osservato può dedurre quale sarà il prossimo numero di sequenza atteso dal server. A quel punto, può costruire un pacchetto contraffatto, con IP e porta del client e valori di sequenza e acknowledgment

coerenti, inserendo nel payload un comando arbitrario da far eseguire alla shell remota. Poiché Telnet è un protocollo testuale non cifrato, il comando viene trasmesso come semplice testo ASCII e, se ben formato, sarà interpretato come input valido dal server.

5.3.2 Codice

Questo pacchetto è parte di una connessione Telnet attiva dal server verso il client. L'attaccante ha intercettato questo pacchetto e ora conosce i seguenti dati cruciali:

- IP e porte della connessione
- Numero di sequenza attuale del server (2737422009)
- Numero di acknowledgment del server, che indica quale byte si aspetta dal client (718532383).
- La dimensione del payload (24 byte) che è la differenza tra sequence number e next sequence number.

Qual è il pacchetto da da spedire per portare a termine l'attacco?

L'attaccante vuole fingere di essere il client che invia dati al server. Per fare ciò, costruisce un pacchetto spoofato con questi campi:

- **Src IP:** 10.0.2.68 (IP del client)
- **Dst IP:** 10.0.2.69 (IP del server)
- **Src Port:** 45634 (porta del client)
- **Dst Port:** 23 (porta standard per Telnet)
- **Sequence number:** 718532383 (numero di sequenza che il server si aspetta dal client)
- **Acknowledgment number:** 2737422033
- **Flags:** PSH,ACK
- **Payload:** ls\n (per esempio)

Note dell'autrice:

- per trovare Src e Dst IP letteralmente inverti i due indirizzi iniziali
- Src Port è quella che prima stava sotto la dicitura Destination Port
- il Sequence number è quello che prima si chimava acknowledgment number
- l'acknowledgment number è il nezt sequence number

- le flags le copiamo
- Importante excursus sul payload: è necessario se vogliamo far eseguire al server un comando, va bene qualsiasi comando Telnet io ho mesos questo a caso

Nel caso si voglia far eseguire un comando al server, come si può procedere?

La risposta a sto point è chiaramente il payload!!
Per far sì che il server esegua un comando (es. in una sessione Telnet), è necessario inserire nel pacchetto un payload contenente il comando, terminato da un carattere di newline (\n), come avverrebbe nella digitazione manuale da parte dell'utente, ricordandosi che il comando deve avere senso nel contesto della shell remota.

5.4 Approcci alla scansione + FTP bounce scan

La scansione all'interno di una rete viene eseguita per recuperare informazioni su determinati host o server, l'obiettivo principale è ottenere informazioni sulle porta utilizzate (TCP/UDP), ciò quali porte sono aperte e in ascolto su determinati nodi, oltre che determinare quale sistema operativo è presente e se esistono sistemi di filtraggio o firewall in una determinata rete.
Lo scanning può essere attivo o passivo:

- **Attivo:** si immette traffico nella rete per recuperare informazioni
- **Passivo:** si fa sniffing senza intervenire attivamente

Diversi approcci:

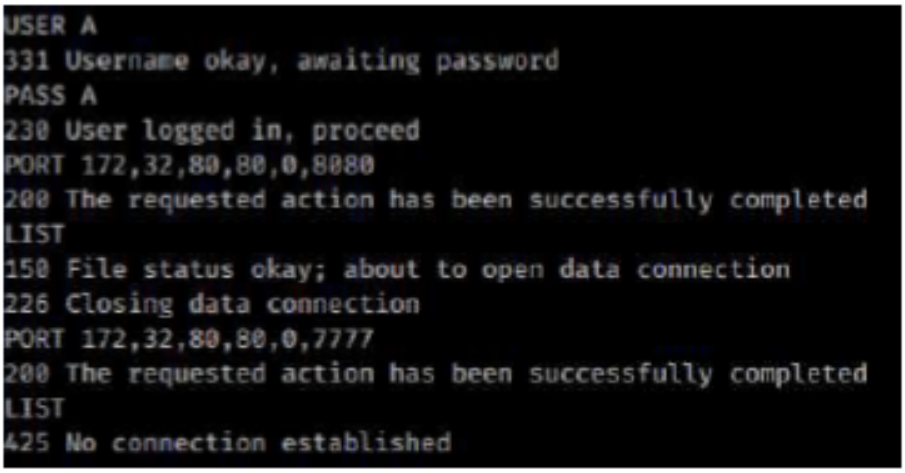
- **Verticale:** host che fa scanning di più target
- **Orizzontale:** molti host fanno scanning sullo stesso target
- **Ibrido:** mix tra i due

Infine il target può essere **singolo**, cioè una macchina, o **multiplo**, cioè anche una porzione di rete (se non tutta).

5.4.1 FTP bounce scan

L'attacco FTP bounce scan è una tecnica utilizzata per eseguire una scansione delle porte (port scanning) di un host di destinazione indirettamente tramite un server FTP, è una delle tecniche più note di port scanning attraverso terze parti.
L'attaccante invia al server FTP un comando PORT utilizzando l'indirizzo IP della vittima tramite un pacchetto spoofato. Se la porta del server è chiusa, quest'ultima risponderà con un pacchetto RST alla richiesta proveniente dal server FTP, mentre verrà eseguita una three-way handshake nel caso in cui la porta fosse aperta.

5.4.2 Commentare praticamente il risultato della seguente scansione



L'attaccante, utilizzando il comando port, riesce a capire che la porta 8080 è aperta (e quindi in ascolto), dato che viene indicato che è possibile trasferire file, mentre la porta 7777 non lo è, dato che non è stata stabilita nessuna connessione.

5.4.3 Due tecniche di scansione con TCP

TCP Connect Scan

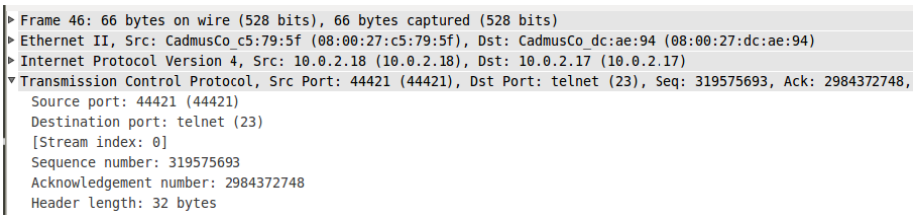
Questa tecnica tenta di stabilire una connessione TCP completa con la porta target inviando un pacchetto SYN, aspettando la risposta SYN-ACK e completando il handshake con un ACK.
Se la connessione si stabilisce, la porta è aperta. Se riceve un pacchetto RST (reset), la porta è chiusa.
E' semplice da implementare ma rumoroso.

SYN Scan (Half-open scan)

Invia un pacchetto SYN al target, ma non completa il handshake. Se riceve un SYN-ACK, invia un pacchetto RST per interrompere la connessione prima che venga completata.
Se riceve un SYN-ACK la porta è aperta, se riceve RST allora è chiusa.
Più furiva della connect scan poichè non comporta connessione, ma può essere rilevata da firewall

5.5 Reset Attack

Un TCP Reset Attack (attacco di reset TCP) è una tecnica usata per interrompere una connessione TCP esistente inviando un pacchetto TCP con il flag RST (Reset) attivo.



In una normale comunicazione TCP, il flag RST viene usato per indicare che qualcosa è andato storto e che la connessione deve essere chiusa immediatamente. Un attaccante, intercettando o osservando una connessione TCP, invia un pacchetto con flag RST a uno o entrambi i lati della connessione.

Capitolo 6

Attacchi

6.1 Problematiche di sicurezza del protocollo SSL

- le preferenze sulla cipher suite non sono autenticate
- usa MD5 come algoritmo hash per il MAC
- tutti i messaggi dell'hashake non sono protetti
- non supporta catene di certificazione o algoritmi non-RSA

E' inoltre possibile fare un man in the middle facendo ARP cache poisoning e posizionandosi come nodo tra client e server in maniera trasparente

6.2 Problematiche di sicurezza del protocollo ARP e discutere ARP poisoning attack

Il protocollo di ARP va ad associare un indirizzo fisico (MAC) a un indirizzo logico (IP): ogni client possiede una tabella cache in cui vengono inserite le associazioni già attive. Se non fosse presente una voce, viene fatta una richiesta in broadcast a tutti i nodi della LAN (viene chiesto chi ha un certo indirizzo IP). Questo genere di messaggi è vulnerabile ad attacchi di spoofing, in quanto si può impersonare un certo nodo della LAN e quindi eventualmente ricevere traffico che non sarebbe destinato all'attaccante. La tipologia di attacco definita ARP poisoning attack consiste nel generare diversi pacchetti spoofati di ARP request in modo da andare a riempire la tabella di cache di informazioni non corrette causando disservizi e rallentamenti nelle comunicazioni tra le LAN.

6.3 Descrivere in cosa consiste IP spoofing, e in dettaglio attacchi che fanno uso di tale tecnica

IP spoofing è un tipo di attacco in cui un client malevolo modifica un pacchetto andando a inserire un altro indirizzo IP rispetto al prossimo, con lo scopo di ingannare una vittima e fare il man in the middle in una determinata comunicazione, oppure può causare un denial of service in quanto può interrompere una connessione o causarne il congestionamento inviando diversi pacchetti. Esistono due tipologie di IP spoofing: non-blind, cioè che l'attaccante cerca di farsi passare per un host della sua stessa LAN, oppure blind, in cui l'attaccante cerca di farsi passare per un host di una qualsiasi sottorete. Gli attacchi di IP spoofing cercano di predirre il sequence number del target per potersi frapporre nella comunicazione, dopodiché si cerca di instaurare un three-way handshake per recuperare informazioni o anche solo causare denial of service.

6.4 Descrivere in dettaglio l'attacco basato su MAC flooding, conseguenze ed eventuali contromisure

Il MAC flooding consiste nell'inviare ad uno switch pacchetti appositamente costruiti per riempire la CAM table (Content Addressable Memory table) che permette di associare rapidamente un indirizzo MAC alla porta a cui è collegato il terminale dello switch, con indirizzi MAC fittizi. Le tabelle degli indirizzi MAC hanno dimensioni limitate. Il MAC flooding fa uso di questa limitazione per inviare allo switch un intero gruppo di indirizzi MAC di origine falsa in modo da saturare la tabella. Quando la tabella è satura lo switch entra in fail-open mode e si comporta come un hub, ovvero invia i pacchetti che riceve a tutti i nodi collegati allo switch

Capitolo 7

Scanning

7.1 tecnica IDLE scan illustrando con un esempio le risposte in caso di porta chiusa, aperta o filtrata

Nel IDLE SCAN abbiamo un **attaccante**, una **vittima** e uno **zombie**, vale a dire un terzo attore che verrà sfruttato dall'attaccante per colpire la vittima indirettamente e capire se una determinata porta è aperta, oppure no, utilizzando pacchetti TCP.
L'attaccante manda un pacchetto TCP di tipo SYN\ACK allo zombie. Quest'ultimo non si aspetta questo messaggio, perciò risponde con un pacchetto RST e un IPID, vale a dire l'identificativo del frame (es. 12345).
L'attaccante invia un pacchetto SYN (con la porta da scansionare) spoofato, utilizzando come mittente l'indirizzo IP dello zombie, alla vittima: in questo caso, se la porta è in ascolto, la vittima risponderà allo zombie con un pacchetto SYN\ACK e un IPID incrementato (es. 12346), a cui lo zombie risponde con un pacchetto RST, in quanto non si aspetta questo genere di comunicazione.
Infine l'attaccante invia un altro pacchetto di SYN\ACK allo zombie, che gli risponde con un RST e un IPID incrementato (es. 12347): confrontando i valori di IPID, l'attaccante capisce che la porta è aperta.
Se invece la porta fosse chiusa o filtrata, il valore di IPID sarebbe stato incrementato una sola volta, in quanto l'attaccante risponderebbe rispettivamente con un RST o non risponderebbe affatto al pacchetto spoofato inviato dall'attaccante, fingendosi lo zombie. Con il secondo SYN\ACK allo zombie, ci sarebbe quindi solo questo incremento di IPID.

7.1.1 contesto della porta 23 della vittima sapendo che l'ultima risposta ottenuta dallo zombie ha id=42380

L'attaccante vuole capire lo stato della porta TCP 23 del target/vittima usando lo zombie quindi invia un pacchetto SYN al target con IP sorgente spoofato dello zombie.

7.2 Descrivere sinteticamente i metodi di scansione stealth

I meccanismi di scansione stealth sono **tecniche usate per identificare porte aperte su un sistema senza suscitare allarmi o lasciare tracce evidenti nei log**, cercando di evitare il rilevamento da parte di firewall, IDS (Intrusion Detection System) o sistemi di logging.
Principi base delle scansioni stealth:

- **Evitare handshake completo:** invece di completare la connessione TCP (che è facilmente loggabile), si inviano pacchetti parziali o particolari per sondare la risposta
- **Minimizzare i pacchetti inviati:** meno traffico significa meno probabilità di essere rilevati
- **Usare pacchetti con flag TCP "particolari":** per confondere filtri e firewall

7.3 Differenza tra le tecniche scan stealth e non stealth

***NON* STEALTH:**

- Completano il normale handshake TCP a 3 vie, aprendo una connessione completa con la porta target
- Vengono registrate nei log del sistema target
- Facili da rilevare e bloccare, poichè utilizzano pacchetti standard senza trucchi particolari

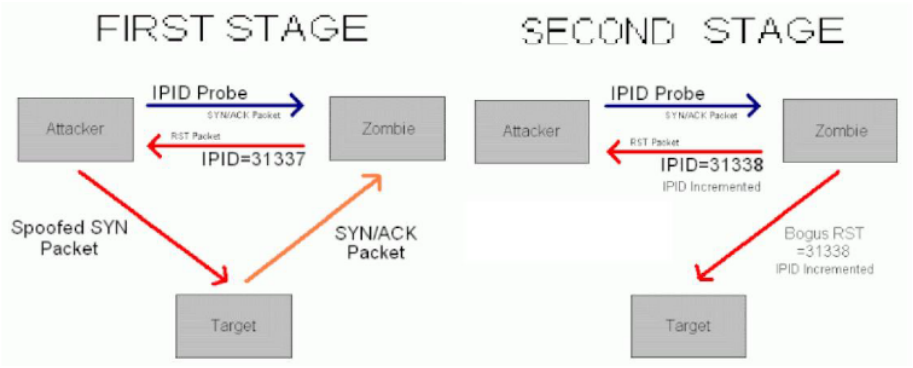
STEALTH:

- Non completano il handshake TCP o inviano pacchetti con flag TCP particolari per evitare di stabilire una connessione completa
- Invia meno pacchetti o pacchetti meno "comuni" per confondere firewall e filtri
- Ridotta probabilità di essere registrati nei log o rilevati da IDS

Capitolo 8

Network Scanning

8.1 Riconoscere e commentare il tipo di scan evidenziato in figura e aggiungere il caso mancante (porta chiusa/aperta)



Il tipo di attacco è conosciuto come IDLE scan in cui viene utilizzato un client intermedio come **zombie** per rendere complicata risalire all'attaccante. La sorgente manda un SYN/ACK allo zombie e aspetta un RST come risposta con IPID. Successivamente l'attaccante invia un pacchetto SYN spoofato con IP sorgente del client zombie verso la vittima con la porta che vuole scansionare. Se la porta è aperta, la vittima risponderà con un SYN/ACK allo zombie. Quest'ultimo non si aspetta un SYN/ACK e risponde perciò con un messaggio RST e con un IPID+1.
Infine l'attaccante manda nuovamente un pacchetto SYN/ACK al client zombie e, se IPID è aumentato, allora la porta è aperta.

Manca il caso in cui la porta è chiusa o filtrata: in questo caso IPID non viene aumentato; quindi, l'attaccante capisce che non c'è traffico su quella specifica porta.

8.2 Descrivere quali sono le condizioni rilevabili di una porta come risultato di uno scanning e cosa indicano ad un potenziale avversario e per ciascuno stato fare un esempio di un tipo di scan che produca quel tipo di stato

8.2.1 condizioni rilevabili

Una porta può essere:

- **aperta:** quindi un servizio è in ascolto su di essa
- **chiusa:** quindi non c'è nulla in ascolto su di essa
- **filtrata:** cioè che è presente un firewall che permette l'accesso solo a determinate sorgenti (in quest'ultimo caso non è possibile definire se è aperta o chiusa)

8.2.2 esempio per tipo:

Per testare lo stato di una porta, si può fare un TCP SYN scan che consiste nell'inviare un pacchetto TCP di tipo SYN + la porta da scansionare, simulando un three-way handshake: se la porta restituisce un pacchetto SYN/ACK, allora la porta è aperta, mentre se restituisco un RST, allora è chiusa. Nel caso di una porta filtrata, il comportamento dipende dal firewall.

8.3 obiettivi, natura degli approcci al port scanning

Guardare 5.4

8.3.1 risultati possibili per la scansione di una porta

Una porta può essere in tre stati: aperta, nel senso che c'è un protocollo in ascolto su di essa, pronto per far partire una comunicazione TCP. chiusa, vale a dire che non è presente nessun protocollo in ascolto e infine filtrata, cioè che è presente un firewall che lascia passare del traffico su una porta solo a determinate condizioni, come per esempio la provenienza da determinati indirizzi. In quest'ultimo caso non è possibile capire se un determinato protocollo è in ascolto o meno.

8.3.2 Descrivere in dettaglio un approccio allo scan

IDLE, guardare 7.1

Capitolo 9

Firewall e NIDS

9.1 Descrivere come funziona un firewall stateful

Uno stateful firewall analizza ogni pacchetto che lo attraversa singolarmente e in più tiene traccia delle connessioni e del loro stato, grazie a una tabella dello stato interna al firewall nella quale ogni connessione TCP e UDP viene rappresentata da due coppie formate da indirizzo IP e porta, una per ciascun endpoint della comunicazione.

9.2 Cosa è un IDS? Descrivere una possibile integrazione tra IDS e firewall

L'IDS è un sistema di monitoraggio utilizzato per identificare accessi non autorizzati a pc o reti locali. Ha tecniche e metodi realizzati per rilevare pacchetti dati sospetti a livello di rete, trasporto e applicazione.
Un IDS non può bloccare o filtrare i pacchetti in ingresso ed in uscita, né può modificarli.
Una possibile integrazione tra firewall e IDS:
Un IPS (intrusion prevention system) invece è comunemente considerato l'accoppiata tra firewall e IDS, cioè si parla di una tecnologia, che cerca di bloccare attacchi alle fasi preliminari, facendo analisi predittiva sulla base di informazioni precedenti ricevute.

9.3 Cosa si intende per stateful firewall? Che differenza esiste con un firewall stateless?

9.3.1 Stateful firewall

I firewall stateful sono in grado di riconoscere le connessioni e le trasmissioni e, ispezionandole, sono in grado di decidere cosa fare in base a molteplici parametri. Del traffico mantengono un log storico, con i dettagli relativi (indirizzi di origine e destinazione, numeri di porta, sequenze TCP, ecc.) e con questo sono in gradi di aprire o chiudere dinamicamente delle porte per consentire o bloccare il traffico.

9.3.2 Stateless firewall

I firewall stateless invece bloccano o consentono una comunicazione soltanto sulla base delle caratteristiche di quest'ultima.
In pratica i pacchetti sono bloccati in base a delle regole statiche (ad es. l'indirizzo sorgente o quello di destinazione, la porta utilizzata) e di ciò non viene tenuta memoria.

9.4 Come funziona un IDS e quali sono le differenze rispetto ad un IPS

9.4.1 IDS

Un IDS (intrusion detection system) è un sistema di monitoraggio utilizzato per identificare dei comportamenti malevoli, rilevando attacchi o altre violazioni alla sicurezza e fornendo informazioni su intrusioni avvenute, grazie all'uso di sonde posizionate sugli host, oppure in certi punti della rete.
Ci sono due tipologie di IDS: **passivi**, che fanno un controllo di firme, e **attivi**, che apprendono i dati del sistema e "imparano" dall'analisi statistica del funzionamento del sistema.

Dove andrebbe posizionato in una rete che abbia due accessi ad internet

In una rete con due accessi a Internet, un IDS va posizionato in un punto centrale dove converga il traffico proveniente da entrambi i collegamenti, dietro i firewall o router, ma prima della rete interna. Questo permette di monitorare tutto il traffico in ingresso e in uscita, indipendentemente da quale connessione venga utilizzata, garantendo una rilevazione completa delle eventuali minacce. Se i due accessi sono completamente separati, può essere necessario un IDS per ciascun punto o un sistema che raccolga il traffico da entrambi.

9.4.2 IPS

Un IPS (intrusion prevention system) invece è comunemente considerato l'accoppiata tra firewall e IDS, cioè si parla di una tecnologia, che cerca di bloccare attacchi alle fasi preliminari, facendo analisi predittiva sulla base di informazioni precedenti ricevute.
La prevenzione è intesa come velocizzazione dei tempi di risposta una volta che si individua un attacco, il problema relativo a questa tecnologia è il rischio di prendere decisioni sbagliate in automatico o il bloccaggio di traffico innocuo

Come realizzare IPS:

Per esempio si può utilizzare Snort (IDS open source molto noto e testato) in modalità inline mode, che riceve i pacchetti direttamente da iptable e collabora per bloccare il traffico sospetto.

9.5 Illustrare le differenze tra application-level gateway e circuit-level gateway

9.5.1 application-level gateway

Un application-level gateway è composto da una serie di proxy che esaminano il contenuto dei pacchetti a **livello applicativo**, fornendo un livello di sicurezza maggiore (es. contro attacchi di buffer overflow): posizionandosi tra client e server, impedisce la comunicazione diretta e può offrire anche servizi di load balancing del traffico.

9.5.2 circuit-level gateway

Un circuit-level gateway invece è un circuito tra client e server a **livello di trasporto** (non applicativo), perciò non fa inspection del traffico che gli passa, ma si occupa solo di tenere traccia delle comunicazioni sul circuito. Anch'esso spezza il modello client\server, facendo da tramite, ma a tutti gli effetti non fornisce sicurezza lato applicativo.

9.6 Cosa si intende per deep packet inspection? Quali funzioni addizionali si trovano generalmente integrate in firewall di questo tipo? Come si applicano a scenari di encrypted threats?

9.6.1 deep packet inspection

La deep packet inspection è una tecnica di packet filtering che va a controllare il contenuto dei pacchetti in transito in maniera approfondita, per identificare codice malevolo. A differenza dei metodi tradizionali che si limitano a ispezionare intestazioni (header) di pacchetti IP, la DPI analizza il contenuto completo dei pacchetti, compreso il payload, per identificare protocolli, applicazioni, comportamenti sospetti o specifici pattern di dati.

9.6.2 funzioni addizionali

- **Application Control (Controllo delle applicazioni):** Permette di identificare e controllare il traffico in base all'applicazione, non solo in base alla porta o al protocollo.
- **Intrusion Detection and Prevention (IDS/IPS):** Identifica comportamenti malevoli nel traffico di rete (es. exploit noti, attacchi buffer overflow).
- **Data Loss Prevention (DLP):**Identifica tentativi di esfiltrazione di dati sensibili, come numeri di carte di credito, codici fiscali, documenti riservati.

9.6.3 scenari di encrypted threats

In scenari di encrypted threats, la deep packet inspection può mettersi in mezzo a una comunicazione criptata in quanto è possibile abilitare la possibilità di far verificare un traffico cryptato per poter capire se è presente un man in the middle che sta sfruttando un canale cifrato per attaccare una rete: la procedura consiste nell'inviare i pacchetti su un server in cloud che fa analisi e mi dice se la connessione è "pulita" o se sono presenti minacce.

9.7 Principi inderogabili dei firewall

Sono 3:

- un fireall deve essere l'unico punto di contatto della rete interna con quella esterna

- solo il traffico autorizzato può attraversare il firewall
- il firewall deve essere esso stesso un sistema altamente sicuro

9.8 proxy firewall

Mediano connessioni applicative e gestiscono aspetti di sicurezza. un proxy firewall è generalmente composto da una serie di proxy che esaminano il contenuto dei pacchetti a livello applicativo.
Nell'interazione con le applicazioni, usare un approccio con ALG richiede uno specifico proxy per ogni applicazione il che comporta un ritardo nel supporto, maggiore consumo di risorse e basse prestazioni. Ha però il vantaggio di rompere il modello client/server tipico, cosa che da maggiore protezione ai server.
In breve i vantaggi sono:

- non permette connessioni dirette tra interno e esterno
- supporta l'autenticazione
- analizza i comandi a livello applicativo
- mantiene log del traffico e delle attività svolte
- analisi migliore del traffico applicativo rispetto ad un firewall stateful

9.8.1 Configurazione reverse proxy

Un reverse proxy è utilizzato per garantire l'accesso degli utenti esterni a risorse interne in modo sicuro e controllato. Si interpone tra i client e i server interni, inoltrando le richieste e nascondendo l'infrastruttura reale. Questo permette di proteggere i sistemi interni, applicare regole di accesso centralizzate e gestire le comunicazioni in modo efficiente e sicuro.

9.9 Come funziona una honey pot? A cosa serve e come la potrei realizare?

Una honeypot è un sistema di sicurezza informatica progettato per attirare, rilevare, deviare o studiare attacchi informatici. Si tratta di un sistema fittizio che simula una risorsa vulnerabile, come un server, un'applicazione o una rete, con l'obiettivo di ingannare gli attaccanti facendoli credere di aver trovato un bersaglio utile. viene realizzata una sorta di rete "finta" (la honey pot) per attirare gli attacchi poichè quando un attaccante vuole entrare, di norma usa falle note, quindi si realizza una rete ad hoc per osservare come gli attacchi vengano perpetrati. per questa ragione, si realizza una DMZ (generalmente 'e una sandbox) proprio a questo scopo. non sono molto usate, ma vengono quasi sempre usate dai provider di sistemi di sicurezza per osservare le vulnerabilità

in un sistema e fare le patch occorrente. nell'honey pot di norma si mettono moltissimi sensori IDS per misurare tutto quello che avviene.

9.10 Descrivere come funziona iptables in dettaglio

E' uno strumento della suite netfilter di Linux usato per filtrare pacchetti di rete, NAT, forwarding e controllo del traffico. È ampiamente usato per creare firewall su sistemi Linux. E' organizzato in:

- **Tabelle:** Ogni tabella contiene catene (chains) che a loro volta contengono regole.
- **Catene:** Ogni tabella contiene catene predefinite (built-in) e personalizzate.
- **Regole:** Ogni catena contiene una lista di regole che specificano:
  - *Condizioni:* cosa deve essere vero per applicare la regola
  - *Azioni (target):* cosa fare se le condizioni corrispondono

9.11 FTP Bounce Attack, dire di cosa si tratta e quale tipologia di firewall potrebbe bloccarlo e come

L'FTP Bounce Attack è un attacco che sfrutta una caratteristica del protocollo FTP in modalità passiva (PORT command) per rimbalzare connessioni attraverso un server FTP verso altri host o porte.
Questo tipo di attacco è risolvibile con uno stateful firewall. un FTP proxy potrebbe riconscere un uso imporprio del protocollo e terminare la connessione. Nella pratica il problema è stato risolto con aggiornamenti degli FTP server

9.12 Cosa è Snort e cosa permette di realizzare? Fare degli esempi di cosa si può ottenere

Snort è un IDS open source molto noto e testato, è in grado di funzionare sia in modalità sniffer (osserva il traffico e logga) sia in modalità IDS.
Ha quattro modalità di utilizzo previste:

- **sniffer mode:** legge i pacchetti dalla rete e li visualizza come un flusso continuo
- **packet logger mode:** logga i pacchetti salvandoli sul disco

- **network intrusion detection system mode (funziona da NIDS):** utilizza regole snort-based per individuare pattern di traffico sospetti
- **inline mode (funziona da IPS):** riceve i pacchetti direttamente da iptable e collabora per bloccare il traffico sospetto.

9.13 Cooperazione tra firewall e IPS

Un firewall e un IPS (Intrusion Prevention System) sono entrambi dispositivi di sicurezza di rete, ma con funzioni complementari.

- **Firewall:** Filtra il traffico in base a indirizzi IP, porte, protocolli, regole statiche
- **IPS:** Analizza il contenuto del traffico in profondità (payload), rileva attacchi, blocca minacce conosciute

Nella pratica l'IPS controlla il traffico permesso dal firewall ispezionando i pacchetti più ne dettaglio, cercando firme di attacchi noti e comportamenti anomali.

Esempio pratico:

L'utilizzo di entrambi potrebbe essere utile tipo nel caso di attacco SQL injection

9.14 Assumendo un firewall posizionato su un router, devono essere filtrati i traffici in ingresso al firewall stesso o solo quelli che devono essere "forwarded"? Spiegare le motivazioni constestualizzando lo scenario

Quando un firewall è posizionato su un router, è fondamentale filtrare sia il traffico destinato al router stesso (INPUT) sia quello che deve essere inoltrato verso altre reti (FORWARD). Questo perché il router non è solo un punto di passaggio del traffico, ma anche un dispositivo esposto e potenzialmente attaccabile. Se non si filtrano le connessioni in ingresso dirette al router (come SSH, ping o interfacce web), un attaccante potrebbe comprometterne il controllo. Allo stesso tempo, filtrare solo il traffico inoltrato non protegge l'infrastruttura di rete in modo completo. Per garantire una sicurezza efficace, quindi, bisogna gestire entrambe le tipologie di traffico con regole di firewall adeguate, riducendo la superficie di attacco e controllando l'accesso tra le varie zone della rete.

9.15 Descrivi tutti i tipi di firewall che conosci associando I diversi livelli ISO/OSI che sono in grado di analizzare

- **Firewall di tipo stateless:** forniscono packet filtering di tipo stateless, quindi un semplice controllo dei pacchetti in transito, basandosi su alcune regole preimpostate, principalmente legate a indirizzi di sorgente e destinazione (livello 3 ISO/OSI – Network).
- **Firewall di tipo stateful:** forniscono packet filtering di tipo stateful, in cui vengono analizzate più informazioni del pacchetto, comprese le porte utilizzate, i protocolli e viene tenuta traccia tramite log delle azioni. Inoltre si ricordano se una sessione TCP era attiva, così da far passare i pacchetti successivi (livello 4 ISO/OSI – Transport).
- **Application Gateway:** esaminano il contenuto dei pacchetti a livello applicativo, fornendo un livello di sicurezza maggiore (livello 7 – ISO/OSI – Application).

9.16 Quali sistemi firewall riescono a controllare il traffico applicativo? Come funzionano?

Un application-level gateway è composto da una serie di proxy che esaminano il contenuto dei pacchetti a livello applicativo, fornendo un livello di sicurezza maggiore (es. contro attacchi di buffer overflow): posizionandosi tra client e server, impedisce la comunicazione diretta e può offrire anche servizi di load balancing del traffico. Se un firewall è un IDS (intrusion detection system) vengono utilizzati insieme, si può ottenere una tecnologia chiamata IPS (intrusion prevention system), che va a fare attività predittiva sulla base a informazioni ricevute in precedenza che permettono di bloccare certi attacchi sul nascere.