Cognome:_____ Nome: _____ Matricola:_____

Sicurezza dei sistemi informatici e delle reti – 15 febbraio 2017 – 6 CFU

Tempo a disposizione: 70 minuti. Libri e appunti chiusi. Vietato comunicare con chiunque. Vietato l'uso di smartphone, smartwatch, calcolatrici e affini.

1. **Sicurezza del codice.** Commenta la sicurezza nei seguenti stralci di codice C relativi alla lettura di una stringa da standard input in cui la lunghezza della stringa è codificata in binario con due bytes all'inizio della stessa.

1.1.
```
int main(int argc, char** argv) {
        unsigned short len; /* intero di 2 bytes senza segno */
        char buffer*;
        read(stdin, &len, 2); /*legge l'intero direttamente in binario*/
        buffer = malloc(len+1);
        read(stdin, buffer, len);
        buffer[len]='\0';   /*termina con zero*/
        . . .
}
```

l'espressione "len+1" va in overflow se len==65535 poiché len è rappresentato con due bytes, quindi len+1==0, il che comporta una allocazione di un buffer lungo 0, ma una lettura di 65535 bytes.

1.2.
```
int main(int argc, char** argv) {
        unsigned short len; /* intero di 2 bytes senza segno */
        char buffer*;
        read(stdin, &len, 2); /*legge l'intero direttamente in binario*/
        if ( len >= 65535 ) {
              . . . /* gestione errore */
        }
        buffer = malloc(len+1);
        read(stdin, buffer, len);
        buffer[len]='\0';   /*termina con zero*/
        . . .
}
```

len non può essere maggiore di 65534 altrimenti viene eseguita la gestione dell'errore dove assumiamo che ci sia un return o exit.
L'espressione "len+1" non può andare in overflow quindi viene sempre allocato correttamente un buffer di lunghezza len+1 sufficienti a contenere len bytes più lo zero di terminazione.

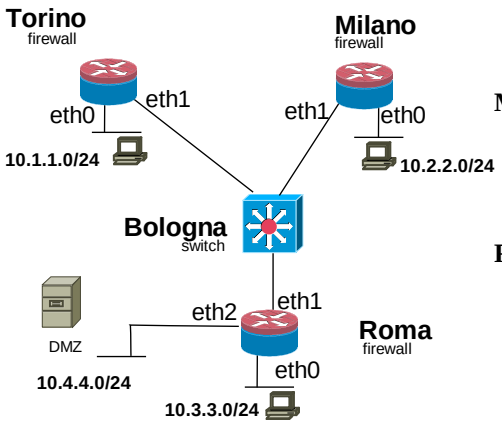
1.3.
```
int main(int argc, char** argv) {
        short len; /* intero di 2 bytes con segno */
        char buffer[1000];
        read(stdin, &len, 2); /*legge l'intero direttamente in binario*/
        if ( len >= 1000 ) {
              . . . /* gestione errore */
        }
        read(stdin, buffer, len);
        buffer[len]='\0';   /*termina con zero*/
        . . .
}
```

len è dichiarato con segno. Supponiamo che len sia negativo. In tal caso la verifica >=1000 non è mai vera quindi non si gestirà mai un errore. La read interpreta la rappresentazione di len in memoria sempre come se fosse unsigned leggendo 2^16-|len| bytes che sono più di quanto il buffer può contenere, per un ampio intervallo di valori di len.

Cognome:_____ **Nome:** _____ **Matricola:**_____

Sicurezza dei sistemi informatici e delle reti – 15 febbraio 2017 – 6 CFU

2. **Sicurezza delle reti.** I firewall delle sedi di Torino, Milano e Roma sono collegate tra loro tramite uno switch come in figura e le loro configurazioni riportate a destra con la notazione usata da iptables.



Torino

```
:FORWARD DROP
-A FORWARD --state RELATED,ESTABLISHED -j ACCEPT
-A FORWARD –i eth0 --state NEW -j ACCEPT
```

Milano

```
:FORWARD DROP
-A FORWARD --state RELATED,ESTABLISHED -j ACCEPT
-A FORWARD -i eth1 –s 10.3.3.0/24 --state NEW -j ACCEPT
-A FORWARD -i eth0 --state NEW -j ACCEPT
```

Roma

```
:FORWARD DROP
-A FORWARD --state RELATED,ESTABLISHED -j ACCEPT
-A FORWARD –i eth0 –o eth2 --state NEW -j DROP
-A FORWARD –o eth2 --state NEW -j ACCEPT
-A FORWARD –i eth1 –s 10.2.2.0/24 --state NEW -j ACCEPT
-A FORWARD –i eth0 --state NEW -j ACCEPT
```

2.1. Mostra in forma di matrice di accesso la policy di sicurezza realizzata dalle configurazioni dei firewall. Indica con Q (Query) la possibilità di iniziare una comunicazione e con R (Reply) che è solo possibile rispondere ad una comunicazione iniziata dalla controparte.

a da	Pc Torino	Pc Milano	Pc Roma	DMZ
Pc Torino				Q
Pc Milano			QR	Q
Pc Roma		QR		
DMZ	R	R		

2.2. Descrivi i problemi di spoofing che trovi nelle configurazioni date, specificando da quali sorgenti possono venire gli attacchi e verso quali destinazioni possono essere eseguiti con successo.

La presenza di regole basate su indirizzi IP sorgente offre la possibilità di avere attacchi basati sullo spoofing di tale indirizzo. In particolare i computer in 10.1.1.0/24 potrebbero inviare pacchetti con sorgente in 10.2.2.0/24 a 10.3.3.0/24 o 10.4.4.0/24 e con sorgente 10.3.3.0/24 a 10.2.2.0/24. Il fw di Torino non li bloccherebbe, i fw di Roma e Milano sono configurati per lasciarli passare.

2.3. Dai una soluzione che prevenga i problemi di spoofing trovati al punto precedente cercando di minimizzare il numero di modifiche alla configurazione. Scrivi esplicitamente le configurazioni da aggiungere ai router di Roma Milano e Torino.

la soluzione più semplice consiste nell'aggiungere alla configurazione del fw di Torino di un filtro anti spoofing:
-A FORWARD –i eth0 –s !10.1.1.0/24 --state NEW -j DROP
oppure nel rendere la regola già presente più specifica
-A FORWARD –i eth0 –s 10.1.1.0/24 –d 10.4.4.0/24 --state NEW -j ACCEPT
o anche solo con una tra le opzioni –s e –d

Alternative meno convenienti sono (1) far passare la comunicazione tra i fw di Roma e Milano su una VPN separata o (2) sostituire lo switch di Bologna con un fw dotato di filtri anti spoofing.

3. **Integrità.** Rispondi alle seguenti domande.

Cognome:_____ Nome: _____ Matricola:_____

Sicurezza dei sistemi informatici e delle reti – 15 febbraio 2017 – 6 CFU

3.1. Descrivi i principi su cui si basa il modello Biba.

vedi materiale didattico

3.2. Descrivi la strutura di un Merkle Hash Tree e la struttura della prova di autenticità del risultato di una query.

Struttura MHT

Albero bilanciato, composizione di hash, vedi materiale didattico per i dettagli

Prova di autenticità di una query.

La prova dell'esistenza e dell'integrità di una foglia v è data dalla sequenza dei fratelli dei nodi che si trovano sul cammino da v alla radice. Un client di un MHT, avendo solo il root hash memorizzato in maniera fidata può verificare la correttezza ricalcolando il root hash a partire da v e dalla prova.

4. Privacy.

4.1. Analogie e differenze tra i concetti di confidenzialità e privacy.

confidenzialità: vedi materiale didattico, solo un aspetto della sicurezza dei dati (di qualsiasi genere), non necessariamente regolato da una legge.

Per privacy si intende la sicurezza dei dati personali (cioè riconducibili ad una persona fisica). E' tipicamente regolata da una legge, e comprende non solo la confidenzialità ma anche l'integrità e la disponibilità di essi oltre ad una serie di procedure amministrative (es. consenso informato, comunicazione al garante ecc.)

4.2. Elenca 5 prescrizioni della legge 196/2003.

1.

vedi materiale didattico

2.

3.

4.

5.

Cognome:_____ **Nome:** _____ **Matricola:**_____

Sicurezza dei sistemi informatici e delle reti – 15 febbraio 2017 – 6 CFU

5. Protocolli crittografici.

5.1. Descrivi il concetto di perfect forward secrecy.

Un protocollo è dotato di perfect forward secrecy se i segreti a lungo termine non sono sufficienti per decifrare una comunicazione registrata in precedenza.

5.2. Come fa una autorità in possesso dei segreti a lungo termine di A e B a intercettare il contenuto di una comunicazione tra loro? Mostra uno schema.

Per protocolli senza PFS è sufficiente decifrare lo scambio della chiave di sessione (o di dati da cui deriva).
Per protocolli con PFS è necessairo che l'autorità intervenga, nel momento in cui la comunicazione sta iniziando, come MitM impersonando (tramite i segreti a lungo termine noti) A o B (nelle comunicazioni rispettivi con B o A)

A ⟷ C ⟷ B
C impersona B C impersona A

5.3. Supponi che A e B abbiano come segreto a lungo termine una chiave **simmetrica** K. Mostra un protocollo di mutua autenticazione e scambio di chiave di sessione che sia dotato di perfect forward secrecy.

Una possibile soluzione è il protocollo di Diffie-Helman in cui i messaggi sono firmati con un message authentication code con chiave K.

6. Sicurezza di sistema in Unix. "sudo".

6.1. Dipendentemente dalla configurazione sudo chiede una passowrd. Di quale utente?

Dell'utente che ha lanciato il comando.

6.2. Perché è preferibile usare il comando sudo rispetto ad usare direttamente un utenza amministrativa (ad esempio acceduta con il comando "su")?

I vantaggi sono vari:
- strict accountability: sudo scrive su log quale utenza ha eseguito ciascun comando e si premura di verificare che al terminale ci sia effettivamente l'utente in questione.
- possibilità di configurare in maniera fine quale utente può lanciare quale comando, vincolandone potenzialmente l'uso

6.3. In quali casi una adeguata configurazione di sudo può sostituire efficacemente l'uso di un wrapper?

Si presume che il comando in questione non debba essere eseguito con password ma l'input sia considerato non fidato (es. ping).
Se l'input non fidato è costituito solo da parametri sudo può essere configurato per limitare le stringhe che possono essere passate al comando e per non chiedere la password, agendo di fatto come un wrapper.