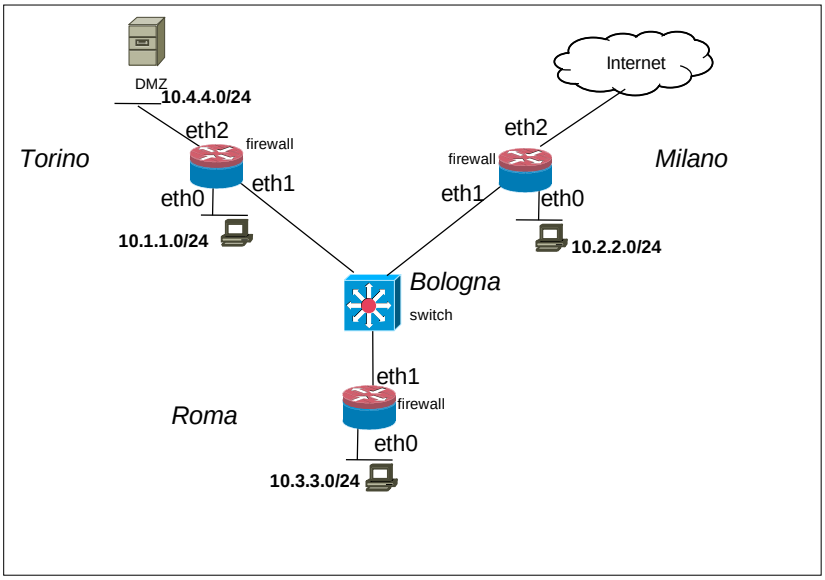
Cognome:_____ Nome: _____ Matricola:_____

Sicurezza dei sistemi informatici e delle reti – 19 febbraio 2013

Tempo a disposizione: 60 (5 cfu) o 70 (6 cfu) minuti. Libri e appunti chiusi. Vietato comunicare con chiunque. Vietato l'uso di cellulari, calcolatrici, palmari e affini.

1. Sicurezza delle reti.

Considera la rete in figura



I firewall delle sedi di Roma, Torino e Milano sono collegati tra loro tramite uno switch a Bologna.

Il firewall di Milano fa anche NAT rispetto a Internet.

La configurazione per **tutti** i firewall è la seguente.

Roma, Torino, Milano

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

1.1. Quali utenti tra quelli di Roma Milano e Torino possono accedere ad Internet? Perchè gli altri non ci riescono? Cosa puoi dire della DMZ?

Solo quelli di Milano. Gli altri vengono bloccati in ingresso su eth1 del fw di Milano. Per la DMZ la situazione è simmetrica. Solo gli utenti di Torino accedono gli altri bloccati su eth1

1.2. Sugerisci delle modifiche alle configurazioni per far sì che (1) gli utenti di Roma e Milano possano accedere ad Internet ma quelli di Torino no, (2) gli utenti di Torino e Roma possano accedere alla DMZ ma quelli di Milano no.

Da aggiungere a Milano: -A FORWARD –i eth1 –o eth2 –s 10.3.3.0/24 -m state --state NEW -j ACCEPT
Da aggiungere a Torino: -A FORWARD –i eth1 –o eth2 –s 10.3.3.0/24 -m state --state NEW -j ACCEPT

1.3. La configurazione che hai suggerito è vulnerabile a spoofing? In caso affermativo, suggerisci delle contromisure sottofoma di modifiche di configurazione.

Si, vulnerabile, quando utenti di Milano e Torino impersonano utenti di Roma. Contromisure:
Da aggiungere a Milano: -A FORWARD –i eth0 –s !10.2.2.0/24 -j DROP
Da aggiungere a Torino: -A FORWARD –i eth0 –s !10.1.1.0/24 -j DROP

Alternativamente si può modificare la regola –i eth0 ...ACCEPT già presente specificando –s 10.xx.xx.0/24, per ciascun firewll con il prefisso della lan degli utenti per quella sede

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

Sicurezza dei sistemi informatici e delle reti – 19 febbraio 2013

2. **Sicurezza del codice**. Considera il seguente codice C in cui la funzione quotedCopy() fa la copia di una stringa aggiungendo delle backslash prima di dei caratteri backquote « ` ».

```
void quotedCopy(char* from, char* to) { ... }

int main(int argc, char** argv) {
      char inpt[1001], quotedInpt[1003];
      char *command;
      scanf("%1000s",inpt);
      quotedCopy(inpt, quotedInpt);
      command=malloc(sizeof(quotedInpt)+20);
      sprintf(command, "ls -l %s", quotedInpt);
      system(command);
}
```

2.1. Sottolinea le righe di codice che secondo te introducono una vulnerabilità e descrivi i relativi problemi di sicurezza.

La chiamata a quotedCopy() va in buffer overflow se vi sono più due backquote nella stringa inpt.
In cascata anche sprintf va in buffer overflow ogni volta che lo quotedCopy() va in buffer overflow di più di 14 caratteri (20-strlen("ls –l ")), ovviamente supponendo che l'overflow precedente non abbia arrestato il processo.
La chiamata system() è vulnerabile a code injection, poiché quotedCopy() non tratta i caratteri "\" in input che possono annullare quelli inseriti dalla quotedCopy() stessa.

2.2. Se tu potessi cambiare il codice come risolveresti i problemi riscontrati?

Una possibilità è quella di cambiare l'interfaccia di quotedCopy() in modo da specificare la lunghezza del buffer di destinazione e troncare in caso di overflow. Un altro approccio potrebbe essere quello di far si che quotedCopy() calcoli la lunghezza del risultato e allochi essa stessa un buffer sullo heap.
Per la system(), l'approccio di modifica della stringa in modo che l'input non possa essere dannoso all'interno di una shell presuppone di trattare correttamente tutti i possibili casi di code injection in una shell (non solo il backquote), la cosa è complicata e quindi prona a errori. Un approccio più efficace è di sostituire system() con exec() che non usa la shell.

3. **Sicurezza delle Public Key Infrustructure (PKI).**

3.1. I certificati delle CA sono diversi dagli altri. Spiega le motivazioni di tale approccio e che verifiche deve fare un browser in quest'ambito.

Un certificato deve specificare se può essere usato per firmare altri certificati o no, altrimenti chiunque potrebbe firmare certificati nella catena di fiducia che fa capo ad una CA senza che la CA possa sapere avere controllo sulle verifiche a cui i subject devono essere sottoposti. Il browser deve verificare che tutti i certificati tranne quello di partenza siano relativi ad una CA.

3.2. Validità del certificato. Perché è necessario avere tale vincolo in un certificato? Come si deve comportare un browser in quest'ambito?

Per limitare il rischio di disclose della chiave privata ad un certo intervallo di tempo (più l'intervallo è lungo più alto il rischio). Il browser deve verificare che l'istante attuale cada nell'intervallo di validità di tutti di certificati della catena.

3.3. Cosa succede se una chiave privata viene erroneamente pubblicata? Descrivi rischi e contromisure previste.

La chiave privata può essere usata da terze parti, per decifrare e/o firmare e/o autenticare illecitamente. Le contromisure prevedono la revoca e la verifica che i certificati usati non siano in CRL o che non siano stati revocati tramite server OCSP.

4. Principi di progettazione

4.1. Descrivi brevemente il principio del confinamento

Vedi materiale didattico

4.2. Descrivi brevemente il principio del minimo privilegio

Vedi materiale didattico

4.3. Elenca **tre** meccanismi a te noti nell'ambito dei sistemi operativi (ad esempio nell'ambito dei sistemi unix o windows), che sono preposti ad attuare politiche di confinamento e/o restrizione di privilegio.

Il meccanismi di controllo di accesso in unix per l'accesso ai file, il maccanismo del chroot, i vari sistemi di virtualizzazione, i meccanismi di controllo di accesso di SELinux, il controllo di accesso in windows.

4.4. Il principio del minimo privilegio in unix è molto complesso da attuare a livello di processo, cioè come vincoli imposti alle system call che il processo può invocare. Mostra **due** esempi di politiche difficili da supportare in unix.

- il processo non deve poter aprire socket server su porte da 3000 a 4000
- il processo deve poter configurare la rete ma non deve poter accedere in scrittura ad alcun tipo di file

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

Sicurezza dei sistemi informatici e delle reti – 19 febbraio 2013

5. **Perfect forward secrecy.** Descrivi.

Vedi materiale didattico.

6. **[solo per 270]** Considera i rischi legati all'adozione di un **database** in cloud pubblica rispetto alle seguenti coordinate di sicurezza: confidenzialità, integrità, disponibilità. Per ciascuna di esse descrivi brevemente, rischio, una contromisura ed eventuali limiti di applicabilità della contromisura indicata.

6.1. Confidenzialità.

Rischio
Diffusione dei dati in chiaro presso individui non autorizzati
Contromisura
cifratura

Limiti
Difficile eseguire query. (le cose migliorano un po' con la cifratura omeomorfa)

6.2. Integrità.

Rischio
Manomissione dei dati.
Contromisura
Meccanismi di firma dei record o ADS su intere tabelle

Limiti
Solo riconoscimento della manomissione non recupero del dato originale.

6.3. Disponibilità.

Rischio
A casua di guasti hw o di connettiività i dati non sono disponibili nel momento del bisogno o sono completamente persi
Contromisura
Replica (le repliche dovrebbero essere accessibili mediante collegamenti completamente separati)

Limiti
Aumento considerevole dei costi.