# Sicurezza di rete – attacchi di base

Laboratorio del corso "Sicurezza dei sistemi informatici" (01GSD)
Politecnico di Torino – AA 2010/11
Prof. Antonio Lioy

*preparata da:*
Cataldo Basile (cataldo.basile@polito.it)
Andrea Atzeni (shocked@polito.it)

v. 2.93 (13/12/2010)

## Scopo dell'esercitazione

La crescente dipendenza dei sistemi informativi da Internet e dalle reti in generale porta in primo piano il problema della pirateria informatica. Spesso si crede – erroneamente – che gli attacchi alle reti possano essere effettuati solo da persone con una grande esperienza o con un quoziente intellettivo superiore alla media. Dimostreremo invece in questa esercitazione che, con programmi liberamente disponibili e poche nozioni di base, si possono creare delle situazioni potenzialmente molto pericolose.

**ATTENZIONE:** alcune delle operazioni proposte costituiscono un reato. Scopo di questo documento è presentarle ai soli fini didattici. Agli studenti è richiesto di svolgerle solo sui PC del laboratorio, ed eventualmente su PC di loro proprietà. Gli autori declinano ogni responsabilità per le azioni svolte fuori dal suddetto contesto.

Nel corso di questa esercitazione verranno usati i seguenti strumenti:

**Nmap** - network e port scanner open source. Progettato per poter effettuare velocemente la scansione di reti di grandi dimensioni. Permette di scoprire ad esempio che tipo di sistema operativo e quali servizi sono attivi sugli host della rete. Disponibile per Linux e Win32.
Home page = `http://www.insecure.org/nmap/`

**OpenVAS** - tool open source che effettua scansione delle vulnerabilità. Permette di scoprire ad esempio versioni software affette da vulnerabilità note e configurazioni insicure dei servizi attivi sugli host della rete. Disponibile per Linux e Win32.
Home page = `http://openvas.org`

**Ettercap** - tool open source per attacchi man in the middle e sniffing all'interno di una LAN. Disponibile per Linux e Win32.
Home page = `http://ettercap.sourceforge.net/`

Sarà inoltre richiesto di avviare e terminare alcuni servizi tramite i seguenti comandi:

**Apache2** - Web server
Usare il comando `/etc/init.d/apache2 { start | stop | restart }`
La configurazione delle porte su cui il server è in ascolto si trova in `/etc/apache2/ports.conf`. Per caricare nuovi moduli è possibile inserire in `/etc/apache2/httpd.conf` comandi del tipo:
```
LoadModule nome_modulo percorso/file.so
```
I moduli standard sono disponibili in `/usr/lib/apache2/modules`.

**VSFTP** - FTP server
Usare il comando `/etc/init.d/vsftpd { start | stop | restart }`

**SSH2** - SSH server
Usare il comando `/etc/init.d/ssh { start | stop | restart }`

**Postfix** - Mail server
Usare il comando `/etc/init.d/postfix { start | stop | restart }`

Nel corso dell'esercitazione dovrete spesso lanciare comandi che richiedono i privilegi di `root`. Per questo motivo vi suggeriamo di diventare `root` sin dall'inizio.

# 1 Scansione degli host

## 1.1 Network Scanning

La prima fase di preparazione di un attacco consiste nell'identificazione dei bersagli. La tecnica nota come *Network Scanning* ha come obiettivo quello di ottenere informazioni su quali host di una determinata rete sono attivi e quali invece non lo sono.

Come si realizza in pratica? Per quale motivo è sconsigliabile un esercizio di questo genere nella presente configurazione di laboratorio?

Una volta individuati i sistemi attivi si può procedere ad identificare la loro natura. In particolare, una serie di tecniche note sotto il nome di *Network Fingerprinting* permettono di ottenere informazioni sul sistema operativo di un host remoto. Il principio si basa sulla diversa implementazione degli stack TCP/IP da parte dei vari sistemi operativi.

Il programma `nmap` è un network scanner molto versatile e potente che permette anche di deteminare il sistema operativo remoto tramire TCP/IP fingerprinting. Per una descrizione delle tecniche usate visionare:

```
man nmap
```

Ora formate delle coppie e procedete in questo modo:

1. attivate il server web Apache
2. provate ora a effettuare un collegamento TCP (`-sT`) sulla porta 80 (`-p 80`) per ottenere informazioni sul sistema operativo (`-O`) dell'host del vostro collega:

   ```
   nmap -sT -P0 -p 80 -O -v indirizzo-bersaglio
   ```

Riuscite ad immaginare perché può essere utile evitare il ping sull'host (`-P0`)?

## 1.2 Port Scanning

Dopo aver identificato la natura del bersaglio, non resta che scoprire quali servizi sono attualmente attivi sull'host. La tecnica nota come *Port Scanning* ha come obiettivo quello di ottenere informazioni su quali porte di un determinato host sono aperte e quali invece non lo sono. Con alcuni accorgimenti si può anche capire quali porte sono filtrate da un firewall o da un filtro di pacchetti. Nmap è uno strumento molto potente che permette molti tipi di port scanning, dove viene ottenuto per ogni porta il nome del servizio noto (se esiste), il numero, lo stato (open, filtered, unfiltered) e il protocollo. Per una descrizione delle tecniche di port scanning attualmente supportati da Nmap fare riferimento all'omonima sezione all'interno del `man`.

Formate delle coppie e procedete in questo modo:

1. scegliete due servizi tra `http`, `ftp`, `ssh`, `smtp`
2. assicuratevi che solo i servizi scelti siano attivi
3. provate ora a fare una scansione TCP (`-sT`) sulle prime 1024 porte dell'host del vostro collega:

   ```
   nmap -sT -P0 -p 1-1024 -v indirizzo-bersaglio
   ```

Riuscite a scoprire quali servizi il vostro collega ha scelto?

Ora eseguite le seguenti operazioni:

1. scegliete nuovamente due servizi tra `http`, `ftp`, `ssh`, `smtp`
2. attivate i servizi scelti su porte diverse da quelle standard (fate riferimento alle note introduttive)
3. provate ora a effettuare nuovamente una scansione TCP sull'host del vostro collega

Avete incontrato dei problemi?

Come pensate si possano risolvere?

### 1.3 Identificazione dei servizi

Per tentare un'identificazione dei servizi in ascolto sulle porte aperte dell'host del vostro collega provate a sfruttare l'opzione (`-sV`) di Nmap:

```
nmap -sV -P0 -p 1-1024 -v indirizzo-bersaglio
```

Per una descrizione dettagliata delle tecniche usate da Nmap per identificare i servizi, fare riferimento nuovamente al `man`.

Quali tecniche e strumenti di sicurezza possono essere usati per contrastare i tipi di attacchi visti finora?

## 2 Cattura del traffico

### 2.1 Man in the middle

**Nota:** non abbiamo testato questo esercizio con reti WiFi. Se lo fate, per esempio nel vostro ambiente casalingo, siamo interessati a sapere se riscontrate problemi.

Ettercap è un tool molto versatile per eseguire attacchi di tipo man-in-the-middle in una LAN. In particolare, in questo esercizio ci concentriamo sulla tecnica nota come "ARP poisoning".

Per una descrizione dei parametri di ettercap e della sua configurazione:

```
man ettercap
man etter.conf
```

Data la natura dell'attacco, sarà utile il comando `arp` per visualizzare il contenuto della cache ARP:

```
man arp
```

Formate gruppi di 3 host (diciamo Alessandro, Beatrice e Claudio) e procedete nel modo seguente:

1. esaminate il funzionamento della rete in condizioni normali: Alessandro, Beatrice e Claudio scambiano dei ping e prendono nota della cache ARP di ciascun host
2. Claudio prova a sniffare il traffico di rete con `tcpdump`:

   ```
   tcpdump -vv -i eth0 icmp
   ```

3. Claudio lancia l'attacco, avviando ettercap nel modo seguente (attenzione ai caratteri "/"):

   ```
   ettercap -Tq -M arp /indirizzo_Ale/ /indirizzo_Bea/
   ```

   Il programma si avvia in modalità interattiva. Claudio può visualizzare una breve guida premendo `h`. Mentre eseguite il comando, verificate il traffico sniffando i pacchetti sui tre host con `tcpdump`:

   ```
   tcpdump -vv -i eth0 icmp or arp
   ```

4. Alessandro e Beatrice scambiano nuovamente dei ping

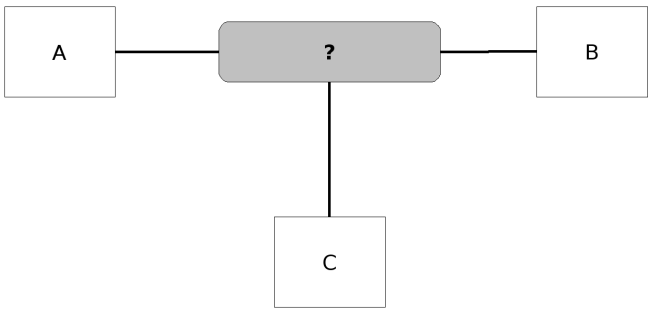


Figura 1: Man in the middle

Che cosa poteva sniffare Claudio prima dell'attacco? E cosa riesce a sniffare dopo?
Riuscite a dedurre dal traffico sniffato il funzionamento dell'attacco?
Cosa vi aspettate sia successo alle cache di Alessandro, Beatrice e Claudio? Verificate le vostre ipotesi.

Riflettete su questa domanda, riferendovi alla figura 1, immaginate di sostituire il dispositivo "`?`" con un *hub*, uno *switch*, un *router* o addirittura *Internet*: quando è possibile effettuare l'attacco man-in-the-middle con ARP poisoning e quali sono i suoi effetti?

Facoltativamente, proviamo ad identificare una possibile contromisura all'attacco:

1. Claudio modifica, nel file `/etc/etter.conf`, la riga `arp_poison_delay` impostandola a `1000` e riavvia ettercap
2. Beatrice cancella dalla cache ARP la riga corrispondente all'host di Alessandro e prova a fare un ping ad Alessandro

Che cosa è successo?

Che cosa aveva fatto Claudio al punto 1?

Che cosa regola la variabile `arp_poison_delay`? Riprovate l'esperimento impostandola a 1 o al valore di default (10).

## 2.2 DNS spoofing

In questo esercizio vediamo come applicare le tecniche di spoofing contro il DNS.

**Nota:** in ambiente virtuale o quando il DNS server è presente nella stessa rete di attaccante e vittima, questi test potrebbero non funzionare. Infatti, il successo del DNS poisoning è basato sulla maggiore rapidità di risposta dell'attaccante rispetto al server legittimo. Ad esempio, nella configurazione NATted proposta, lo switch virtuale (processo software che simula il comportamento di uno switch) è collegato anche ad un DNS server che risponde (quasi sempre) prima di qualsiasi attaccante.

Nuovamente utilizziamo `ettercap` per i nostri scopi. Analogamente all'esercizio precedente, la prima cosa che dobbiamo fare è porci nel mezzo tra la nostra vittima e il suo DNS. A questo punto `ettercap` si occuperà di "filtrare" le richieste della vittima rispondendo prima del DNS.

Formate gruppi di 2 host (diciamo Alessandro, l'attaccante, e Beatrice, la vittima) e procedete nel modo seguente:

1. Alessandro sostituisce il file di test `etter.dns` in `/usr/share/ettercap/` con quello fornito dal corso o aggiunge il comando
   ```
   www.polito.it A 130.192.1.8
   ```
   in quello esistente;
2. Beatrice controlla il contenuto del file `/etc/resolv.conf`
3. esaminate il funzionamento della rete in condizioni normali: Beatrice prova a connettersi ad Internet, ad esempio al sito `www.polito.it`. Osservate direttamente l'interrogazione al DNS con il comando:
   ```
   nslookup www.polito.it
   ```

4. Alessandro, dopo aver identificato l'indirizzo del gateway (potete usare il comando `route`) avvia l'attacco:

   ```
   ettercap -T -M arp:remote /indirizzo_Bea/ /indirizzo_GW/ -P dns_spoof
   ```

5. Beatrice riprova a connettersi al sito www.polito.it.

Cosa succede questa volta?

Cosa può essere successo se l'attacco non è andato a buon fine? Fate le vostre ipotesi (e confrontatele con la nota iniziale).

Cosa vede Alessandro dall'output di `ettercap`?

Analizzate la configurazione del plug-in DNS, nel file `/usr/share/ettercap/etter.dns`.

Come un attaccante può ottenere effetti simili a questo attacco in caso non si trovi nelle vicinanze della vittima?

### 2.3 Sniffing

Passiamo ora a sperimentare attacchi di tipo passivo. In particolare, lo *sniffing* è un attacco di tipo passivo che consiste nel catturare i pacchetti che passano attraverso la nostra scheda di rete Ethernet impostata in modo promiscuo. In questo modo tutti i pacchetti che dovrebbero essere ignorati, ossia quelli che non corrispondono all'indirizzo MAC della scheda, vengono invece copiati in un buffer.

Avete già sperimentato questa tecnica negli esercizi precedenti, catturando pacchetti di rete tramite il comando `tcpdump`: ora ci concentreremo sulla cattura di dati sensibili scambiati all'interno dei protocolli applicativi.

Per sperimentare attacchi di sniffing useremo ancora `ettercap` che estrae già username e password dei più comuni servizi (incluso FTP). Un altro tool utile a questo scopo è Ngrep (http://ngrep.sourceforge.net).

Lavorate sempre divisi in gruppi di 3 host:

1. Alessandro attiva il servizio `ftp`, dopo aver modificato il file di configurazione `/etc/vsftpd.conf` decommentando la riga:

   ```
   local_enable=YES
   ```

2. Alessandro crea gli utenti `ale` e `bea`, impostandone la password:

   ```
   adduser ale
   adduser bea
   ```

3. Claudio lancia l'attacco MITM con `ettercap` come nell'esercizio 2.1

4. Beatrice si collega al server FTP presente sulla macchina di Alessandro, esegue il login e si disconnette

**Nota:** In alcuni casi, l'esercizio seguente (opzione `-e` di `ettercap`) non funziona correttamente con le macchine virtuali. L'ARP spoofing avviene correttamente, ma il modulo di identificazione delle regexp non riesce a riconoscere le stringhe.

Provate ora ad estrarre contenuti sensibili dal traffico mail:

1. Alessandro avvia il server di posta Postfix

   (altrimenti il server di posta locale non accetta connessioni da remoto)

2. Claudio esegue il comando:

   ```
   ettercap -T -M arp /indirizzo_Ale/25 /indirizzo_Bea/ -e ¨Carta di Credito¨
   ```

3. Beatrice invia una mail a ale@grml con questo messaggio "Ho bisogno di un voto maggiore di 27, la mia Carta di Credito e' 7865-8993-6282-8282. Grazie!" usando il client di posta `smtp`

   ```
   smtp -h indirizzo_Ale ale@grml -s ¨Esame di Sicurezza¨
   ```

   (premete `Ctrl-D` una volta digitato il contenuto)

4. Alessando può verificare con un client di posta che l'utente `ale` abbia ricevuto il messaggio, ad esempio può usare il client testuale `mutt`:

   ```
   su - ale
   mutt
   ```

Provate infine a spiare la navigazione web:

1. Alessandro attiva il servizio `http`
2. Claudio entra in modalità grafica. Se site in console mode, digitate:

   ```
   exit
   <x>   <f>
   ```

   Abilitate l'accesso al display all'utente `root`: aprite un terminale e digitate

   ```
   xhost +
   ```

3. Claudio, con i privilegi di `root`, avvia il browser e lancia `ettercap`, attivando il plug-in `remote_browser`

   ```
   su
   firefox &
   ettercap -T -M arp /indirizzo_Ale/80 /indirizzo_Bea/ -P remote_browser
   ```

   Nota: la configurazione del plug-in si trova in `/etc/etter.conf`.

4. ora Beatrice può collegarsi con un browser al server http presente sulla macchina di Alessandro, ad esempio:

   `w3m http://`*`indirizzo_Ale`*`/info2www/`

   e seleziona alcuni link

Cosa è successo nei precedenti test?

Come potrebbe essere contrastato questo attacco?

Provate a eseguire un login tramite protocollo SSH e catturare username e password: cosa è successo?

# 3 Attacchi contro la disponibilità (opzionale)

Vediamo ora un esempio di attacco di tipo Denial of Service (DoS).

Come nell'esercizio 2.1 Claudio lancia un attacco MITM contro Alessandro e Beatrice.

**Nota:** abbiamo avuto problemi ad effettuare questo attacco in ambiente virtuale con VMware Server 2.0. Le motivazioni non sono ben chiare e probabilmente legate all'implementazione di questa versione di VMware Server (fino alla 1.0.4, la versione utilizzata l'anno scorso, non erano stati riscontrati problemi).

- se non lo ha già fatto, Alessandro avvia il server SSH
- Beatrice si connette via SSH al computer di Alessandro, e prova a lanciare alcuni comandi
- Claudio apre un nuovo terminale e lancia l'attacco utilizzando il comndo `tcpkill`:

  `tcpkill -i eth0 host` *`indirizzo_Bea`* `and port 22`

- Beatrice prova a lanciare ulteriori comandi via SSH, e ad aprire una nuova sessione SSH con Alessandro

Che cosa succede, quando `tcpkill` è attivo? Verificate le vostre ipotesi sniffando il traffico di rete sui 3 host.

Che cosa deve conoscere Claudio per poter chiudere la sessione TCP tra Alessandro e Beatrice?

# 4 Identificazione delle vulnerabilità dei servizi (da svolgere a casa)

L'ultima versione di GRML, la 2009_10, ha rimosso Nessus, uno scanner di vulnerabilità molto semplice da utilizzare e molto completo. La ragione è da ricercarsi nella scelta di Nessus di far pagare i plugin avanzati. Da Nessus è stato derivato (come *fork*) OpenVAS (maggiori dettagli su `http://www.openvas.org/`) che ne eredita il ruolo open source e le funzionalità.

Putroppo la transizione verso il nuovo strumento in GRML non è stata ultimata (o non sarà mai fatta per ragioni di spazio su CD) ed il demone di scansione `openvas-server` non è presente nella distribuzione live. È necessario quindi scaricarlo con gli strumenti standard di Debian. Tuttavia questo processo richiede alcuni minuti (11 minuti sulle nostre macchine) e richiede di scaricare quasi 66.2 MB di dati, cosa che potrebbe mettere in crisi la rete del laboratorio qualora tutti lo scaricassero contemporaneamente.

**NOTA:** se avete creato delle macchine virtuali con 256 MB e 512 MB di RAM il processo non andrà a buon fine perché il RAMdisk su cui GRML salva i dati temporanei e di installazione non è sufficiente. Abbiamo testato l'esercizio con 1 GB e tutto funziona correttamente (richiede circa 870 MB).

Aggiornate prima la lista dei repository di package conosciuti da `apt-get` con:

```
apt-get update
```

Quindi, installate `openvas-server`:

```
apt-get install openvas-server
```

Successivamente scaricate ed aggiornate i plugin con il seguente script:

```
openvas-nvt-sync
```

Passiamo ora all'esercizio vero e proprio. Per maggiori informazioni consultate:

```
man openvas-client
man openvasd
```

Formate delle coppie (diciamo Alessandro e Beatrice).

1. Alessandro avvia Apache
2. Beatrice avvia il demone `openvas-server`

   ```
   /etc/init.d/openvasd start
   ```

   e crea un utente per OpenVAS, con il comando `openvas-adduser`
3. Beatrice avvia il server grafico X (`startx`) e successivamente `openvas-client`. Poi procede come segue:
   (a) effettua il login (con l'utente precedentemente creato)
   (b) crea un nuovo task (menù Task → New)
   (c) crea un nuovo scope (menù Scope → New)
   (d) seleziona le plug-in (deselezionate tutto eccetto "General", "Misc" e "Web application abuses")
   (e) definisce il target (IP_Alessandro)
   (f) definisce le porte (1-1024)
   (g) avvia la scansione (menù Scope → Execute)

Che cosa avete notato?

Ora procedete cercando di rimediare alle vulnerabilità identificate:

1. Alessandro riconfigura Apache, cercando di eliminare quanto riscontrato
2. Beatrice ripete lo scanning

Siete riusciti ad eliminare tutte le segnalazioni?

Tutte le segnalazioni vi sembrano ugualmente rilevanti?

Provate a rimuovere “ServerTokens Prod”.

**Nota:** in questo caso non dovreste avere incontrato problemi, ma spesso le soluzioni (in particolare i comandi) che vengono proposti da OpenVAS non sono aggiornati o non sono formattati correttamente (es. spazi tra il - ed il comando), vi consigliamo di guardare sempre la documentazione ufficiale come ulteriore verifica e di diffidare dei copia-e-incolla dalla finestra OpenVAS.