# Reti Avanzate — Sicurezza dei Dati
# A.A. 2017–2018, secondo semestre
# Traccia delle lezioni

Mauro Brunato

Versione 2018-04-22

**Caveat lector**

Lo scopo principale di questi appunti è quello di ricostruire quanto detto a lezione. Queste note non sono complete, e la loro lettura non permette, da sola, di superare l'esame. Le fonti utili a ricostruire un discorso coerente e completo sono riportate alla pagina web del corso, dov'è disponibile anche la versione più recente di queste note:

`http://disi.unitn.it/~brunato/RetiAvanzate/`

Alcune fonti per approfondimenti sono indicate nelle note a pie' di pagina di questo documento.

Si suggerisce di confrontare la data riportata sul sito web con quella che appare nel frontespizio per verificare la presenza di aggiornamenti.

Alcune esercitazioni di laboratorio non sono riportate in questa dispensa perché descritte dal codice commentato disponibile alla pagina web del corso, oppure riportate in dettaglio in un documento a sé stante.

# Changelog

## 2017-04-22

- Principi di crittografia
- Protocollo Diffie-Hellman
- Funzioni hash crittografiche

## 2017-03-20

Quarta e quinta settimana

- Firewall, access control list
- Architetture di rete e zone
- NAT: UDP hole punching
- Esercizi e domande teoriche

## 2017-03-12

Terza settimana

- Sicurezza delle reti: ricerca di macchine e servizi con il port scanning
- Principali metodi di attacco

## 2017-03-05

Seconda settimana

- Livello Rete: utilizzo delle sottoreti IP
- Routing fra VLAN
- Livello Rete/trasporto: NAT
- Domande di comprensione sul livello Data Link e Rete; NAT
- Seconda esercitazione

## 2018-02-26

Versione iniziale

- Livello Data Link: protocollo Ethernet, hub, switch
- Reti locali virtuali (VLAN)

# Indice


**I Appunti di teoria** **4**

**1 Livello 2: Data Link** **5**
1.1 Le reti locali (Local Area Networks, LAN), Ethernet . . . . . 5
1.1.1 Apparati di rete . . . . . 5
1.2 LAN virtuali (VLAN) . . . . . 7

**2 Livello 3: Networking** **12**
2.1 Organizzazione delle sottoreti IP . . . . . 12
2.1.1 Un esempio . . . . . 12
2.1.2 Mappatura in reti locali . . . . . 14
2.2 InterVLAN Routing . . . . . 14
2.2.1 Router a due porte . . . . . 14
2.2.2 Router a una porta . . . . . 15
2.2.3 Layer 3 switches . . . . . 15

**3 Livello 4: Trasporto** **17**
3.1 Network Address Translation (NAT) . . . . . 17
3.1.1 Indirizzi privati . . . . . 17
3.1.2 Caso base: connessione TCP . . . . . 17
3.1.3 Connessioni in ingresso . . . . . 20
3.1.4 STUN: Session Traversal Utilities for NAT . . . . . 20

**4 Livello Applicazione: sicurezza delle reti** **23**
4.1 Le fasi di un attacco hacker . . . . . 23
4.1.1 Ricostruzione delle macchine presenti in una rete . . . . . 23
4.1.2 Ricerca delle porte aperte . . . . . 23
4.1.3 Sfruttamento dei punti deboli di una macchina . . . . . 24
4.2 Esempi da approfondire . . . . . 26
4.3 Firewall . . . . . 26
4.3.1 Packet filtering . . . . . 27
4.3.2 Stateful inspection . . . . . 28
4.3.3 Proxy applicativi . . . . . 28
4.4 Configurazioni di rete . . . . . 30
4.4.1 Reti Small Office / Home Office (SOHO) . . . . . 30
4.4.2 Reti più complesse . . . . . 31
4.4.3 Irrobustimento dei server . . . . . 31

**5 Crittografia** **32**
5.1 Motivazioni . . . 32
5.2 Definizioni . . . 32
5.2.1 Principio di Kerckhoffs . . . 33
5.2.2 Cifrari dimostrabilmente sicuri . . . 33
5.2.3 Categorie . . . 34
5.2.4 Attacchi . . . 34
5.3 Condivisione della chiave: Diffie-Hellman . . . 34
5.3.1 L'algoritmo . . . 35
5.3.2 Suscettibilità agli attacchi Man-in-the-Middle . . . 35
5.4 Funzioni hash crittografiche . . . 36

**II Domande ed esercizi** **38**

**A Domande teoriche e spunti di riflessione** **39**
A.1 Livello Data Link . . . 39
A.1.1 Prerequisiti . . . 39
A.1.2 LAN virtuali . . . 39
A.2 Livello rete . . . 40
A.2.1 Prerequisiti . . . 40
A.2.2 Inter-VLAN routing . . . 40
A.3 Livello trasporto . . . 40
A.3.1 Prerequisiti . . . 40
A.3.2 NAT . . . 40
A.4 Livello applicativo . . . 40
A.4.1 Prerequisiti . . . 40
A.4.2 Tipi di attacco . . . 41
A.4.3 Firewall . . . 41
A.4.4 Configurazioni di rete . . . 41
A.5 Crittografia . . . 41
A.5.1 Prerequisiti . . . 41
A.5.2 Definizioni . . . 41
A.5.3 Diffie-Hellman . . . 42
A.5.4 Funzioni hash crittografiche . . . 42
A.6 Domande a risposta multipla . . . 42

**B Esercizi** **45**

# Parte I

# Appunti di teoria

# Capitolo 1

# Livello 2: Data Link

## 1.1 Le reti locali (Local Area Networks, LAN), Ethernet

Una LAN[1] è una rete privata tra terminali "fisicamente" vicini (fino a qualche chilometro), connessi mediante schede di rete ed opportuno cablaggio (hub, switch, cavi rame o fibra, onde radio).

Ethernet[2] è ormai lo standard *de facto* nelle LAN. È nata come sistema broadcast su canale (bus) condiviso (trasmissione simultanea a più stazioni in banda base, ossia usando tutta la banda disponibile, su cavo coassiale), e si è sviluppata adottando man mano strategie più efficienti (collegamenti punto a punto, doppini intrecciati, fibra ottica).

L'indirizzamento Ethernet è "piatto", non riflette la topologia della rete: ogni scheda terminale ha un identificativo unico, fissato nel firmware (indirizzo MAC, MAC address)[3], da 48 bit (6 byte); l'intestazione Ethernet riporta, nell'ordine, il MAC address del destinatario, quello del mittente e un identificativo da 2 byte del protocollo usato nel payload.

### 1.1.1 Apparati di rete

Un *hub*[4] è un dispositivo di livello fisico che replica il segnale entrante in una porta su ogni altra porta, opportunamente ripulito e amplificato.

Uno *switch*[5] è un dispositivo di livello 2 che inoltra una trama Ethernet entrante esclusivamente sulle porte dove è possibile che il destinatario sia in ascolto. Per fare ciò, lo switch mantiene una tabella (dizionario) che associa a ogni indirizzo MAC già noto la porta a cui è collegato (vedi Fig. 1.1).

Un moderno cablaggio Ethernet prevede una gerarchia di switch ad albero, nella quale i terminali sono le foglie, con eventuali collegamenti ridondati per evitare che un singolo guasto porti alla partizione della rete.

Possiamo distinguere i dispositivi di una rete locale in *terminali* e *di comunicazione*:

- Dispositivi *terminali* (Data Terminal Equipment, **DTE**)— sono quegli apparati che fungono da mittenti o da destinatari delle trame Ethernet, e le cui porte hanno un indirizzo MAC: PC, stampanti, scanner, telefoni IP...
  Anche i router appartengono a questa categoria: infatti sono i destinatari finali delle trame contenenti un carico di livello rete da inoltrare all'esterno della LAN: anche le loro porte Ethernet hanno un MAC address, perché i PC debbono poter indirizzare le trame in uscita verso di loro.

[1] `https://en.wikipedia.org/wiki/Local_area_network`
[2] `https://en.wikipedia.org/wiki/Ethernet`
[3] `https://en.wikipedia.org/wiki/MAC_address`
[4] `https://en.wikipedia.org/wiki/Ethernet_hub`
[5] `https://en.wikipedia.org/wiki/Network_switch`

```
1.  inizializza tabella ← dizionario vuoto          Inizialmente nessun destinatario
2.  quando ricevi frame F dalla porta P
3.  ⌈ tabella[F.mittente] ← P                       Registra da dove arriva il mittente
4.  ⌊ accoda F, P                                   Metti il frame nella coda di invio
5.  quando estrai F, P dalla coda
6.  ⌈ se esiste tabella[F.destinatario]             Se il destinatario è registrato
7.  |    inoltra F alla porta F.destinatario        Inoltra direttamente
8.  | altrimenti                                    Altrimenti broadcast sulle altre porte
9.  | ⌈ per ogni porta R ≠ P
10. | ⌊    inoltra F alla porta R
```

Figura 1.1: Pseudocodice per il mantenimento di una MAC address table all'interno di uno switch


Figura 1.2: Collegamento DTE-DCE con cavo diretto e collegamento DTE-DTE (o DCE-DCE) con cavo incrociato. In tutti i casi, i piedini di uscita di un dispositivo sono collegati ai corrispondenti piedini di ingresso dell'altro. Nota bene: le pedinature sono esemplificative, non corrispondono a nessuno standard reale.

- Dispositivi *di comunicazione* (Data Communication Equipment, **DCE**) — non sono destinatari finali delle trame, e normalmente le loro porte Ethernet non hanno nemmeno un MAC address. Servono a inoltrare le trame da una porta all'altra.

La distinzione fra DTE e DCE è importante in quanto si riflette sul cablaggio della rete. In base allo standard Ethernet, le pedinature dei connettori DTE e DCE invertono le linee dati di ingresso con quelle di uscita. Di conseguenza (vedere Fig. 1.2):

- I cavi utilizzati per connettere un DTE e un DCE collegano semplicemente i piedini dei connettori aventi numerazione corrispondente (piedino 1 a piedino 1 e così via). In questo modo collegano gli ingressi di un dispositivo alle uscite dell'altro e viceversa. Sono detti cavi "diretti", "straight-through" (o "straight-thru"), o semplicemente "patch".
- I cavi utilizzati per connettere dispositivi della stessa categoria (DTE con DTE, oppure DCE con DCE) invertono coppie corrispondenti di piedini. Sono setti cavi "incrociati", "crosslink", "crossover" o semplicemente "cross"[6].

### Domini di collisione e di broadcast

Due dispositivi connessi da un hub non possono trasmettere contemporaneamente: l'hub replicherebbe ciascuno dei due segnali corrompendoli. I due dispositivi appartengono allo stesso *dominio di collisione*[7].

Due dispositivi connessi da uno switch possono trasmettere contemporaneamente (lo switch partecipa al protocollo MAC di Ethernet), quindi uno switch separa i propri ingressi in domini di collisione distinti. Uno switch inoltra i pacchetti broadcast (MAC di destinazione `FF:FF:FF:FF:FF:FF`) su tutte

[6] `https://en.wikipedia.org/wiki/Crossover_cable`
[7] `https://en.wikipedia.org/wiki/Collision_domain`

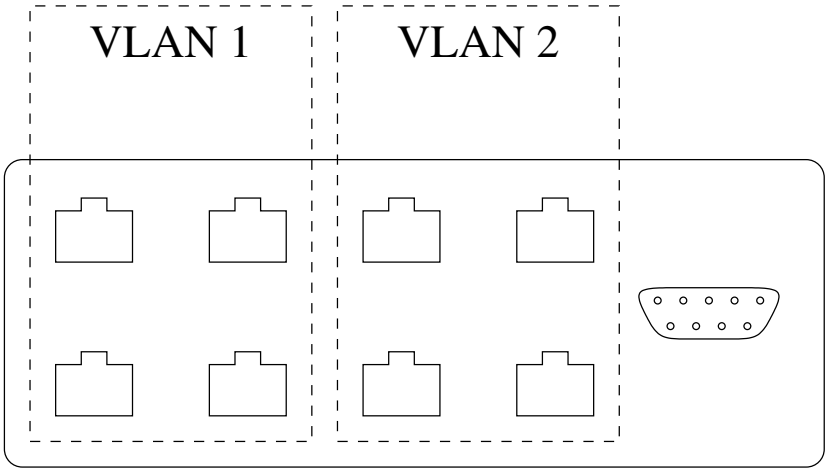



Figura 1.3: Partizione di uno switch in più VLAN.

le uscite. Una rete locale consiste normalmente in pochi domini di broadcast[8] e di molti domini di collisione.

## 1.2 LAN virtuali (VLAN)

- In uno switch di livello 2 è possibile raggruppare alcune delle porte che lo compongono (o alcuni dei MAC Address ad esso afferenti) a formare un dominio di broadcasting autonomo (VLAN: virtual LAN)[9].
- Creando più VLAN si ottiene quindi un numero equivalente di domini di broadcasting del tutto indipendenti, come se avessimo suddiviso lo stesso switch fisico in più switch logici fra loro separati.
- Il vantaggio sta quindi:
  - nel risparmio economico di acquisto e gestione;
  - nella riduzione del traffico di broadcasting;
  - nella possibilità di gestire con maggior granularità gli aspetti legati alla sicurezza

Nel caso più frequente e più semplice, ogni porta viene assegnata a una specifica VLAN (Fig. 1.3). Nel singolo switch si definiscono i nomi delle varie VLAN (es.: `vlan1`, `vlan2`...) e si associano a ciascuna le relative porte.

Se un host viene spostato da una porta a un'altra occorre riconfigurare lo switch, ma questo offre un vantaggio in termini di sicurezza.

È possibile estendere una VLAN attraverso più switch, come si vede in Fig. 1.4.

Due LAN possono anche essere completamente separate da un punto di vista logico, pur condividendo alcune connessioni. Un collegamento fra switch può essere infatti utilizzato per una sola VLAN, oppure per portare pacchetti di più VLAN diverse, ad esempio quando le stesse VLAN occupano edifici diversi (Fig. 1.5):

[8] `https://en.wikipedia.org/wiki/Broadcast_domain`
[9] `https://en.wikipedia.org/wiki/Virtual_LAN`

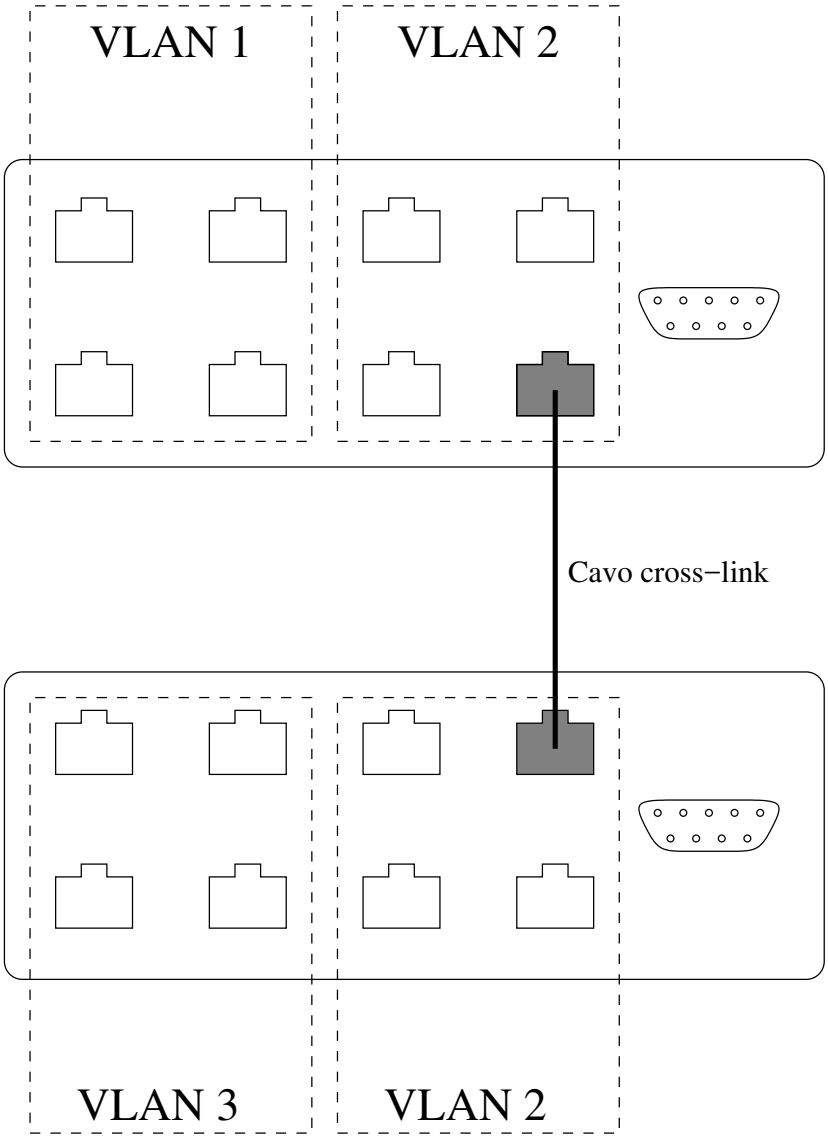



Figura 1.4: Estensione di una VLAN su più switch.

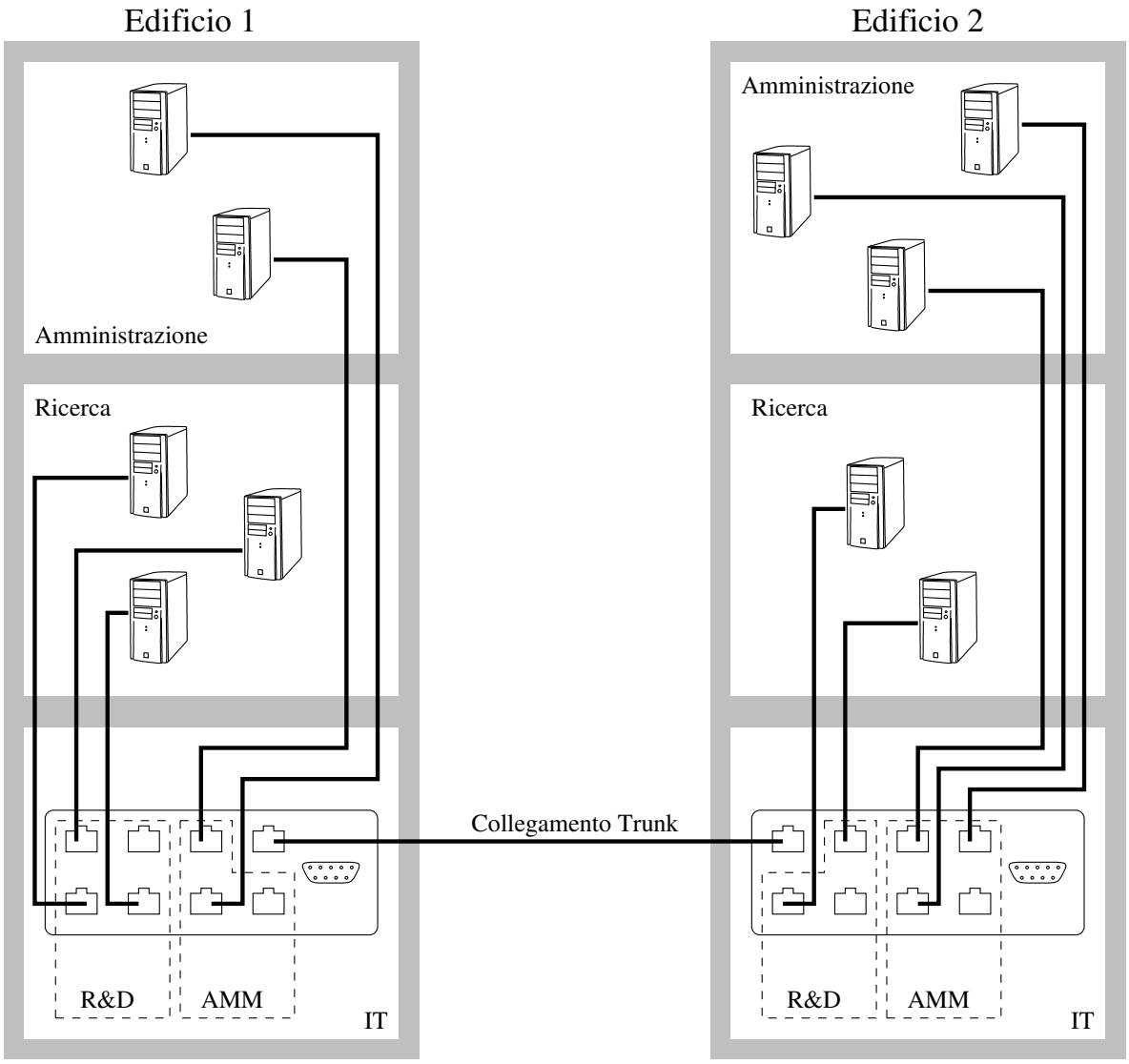



Figura 1.5: VLAN estese fra edifici, con un link condiviso (*trunk link*).

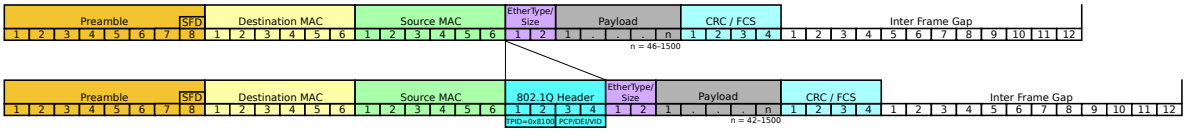



Figura 1.6: Inserimento del tag VLAN in una trama ethernet lungo un trunk link.

- Access link — Nel primo caso, le due porte appartenenti alla connessione sono associate a una VLAN specifica (si dice che sono configurate in modalità *access*). Tutti i pacchetti che transitano per quella linea sono implicitamente appartenenti alla stessa VLAN.
- Trunk link — È il caso più interessante: lo stesso link porta trame appartenenti a varie VLAN. Esempio, il link fra edifici visto in precedenza.

Se più trame possono transitare per lo stesso link, devono contenere l'informazione della VLAN di appartenenza.

La porta che immette la trama nel trunk link inserisce nella trama un campo di 4 byte che contiene il valore identificativo (12 bit) della VLAN. Tale campo è detto *tag*. Lo standard che estende in tal senso la definizione dell'intestazione Ethernet è IEEE 802.1Q.

Come si vede in Fig. 1.6, il campo viene inserito prima del campo Length / Protocol. Contiene:

- Il *Tag Protocol Identifier* (TPID, 2 byte), sempre `0x8100`.
- Il *Tag Control Identifier* (TCID, 2 byte), suddiviso in:
  - *Priority Code Point* (PCP, 3 bit), da 0 a 7;
  - *CanonicalFormat Indicator* (CFI, 1 bit), 0 in Ethernet;
  - *VLAN Identifier* (VID, 12 bit), numero della VLAN.

Il campo addizionale (tag) viene utilizzato dalla porta ricevente per indirizzare il pacchetto esclusivamente alle altre porte dello switch appartenenti alla stessa VLAN.

Una trama che transita attraverso un trunk link è quindi detta "tagged" ("taggata", etichettata) ad indicare che essa contiene l'identificativo della VLAN di appartenenza.

### Eccezione

Lungo un trunk link possono anche passare trame senza tag (untagged); esse possono essere associate ad una ed una sola VLAN che viene detta *nativa*.

La Figura 1.7 presenta un esempio di trattamento di una trama mentre transita per i diversi link di tipo access e trunk che connettono la sorgente alla destinazione.

- Tre LAN virtuali: PC1/PC3/PC5/PC7 (`vlan1`), PC2/PC8 (`vlan2`), PC4/PC6 (`vlan3`).
- Una trama da PC1 a PC7 viaggia in modo nativo da PC1 allo switch 1; viene munita di tag nei due segmenti trunk, poi torna in modo nativo nell'ultimo access link.

Una porta di tipo trunk viene utilizzata non solo nei link fra gli switch ma anche nel caso in cui a una porta afferiscano dispositivi diversi (es. un PC ed un telefono VoIP), che inviano/ricevono rispettivamente trame senza tag 802.1Q (PC) e trame con tag (telefono VoIP).

- Il PC invia di solito alla porta trame prive di tag 802.1Q, e non è consapevole dell'esistenza di una LAN virtuale; il telefono invia trame con tag appartenenti ad una VLAN specifica.
- Se lo switch riceve dati da entrambi attraverso una stessa porta, questa viene definita di tipo trunk, ma ad essa viene associata anche una VLAN nativa, in modo che essa possa ricevere le trame prive di tag del PC.

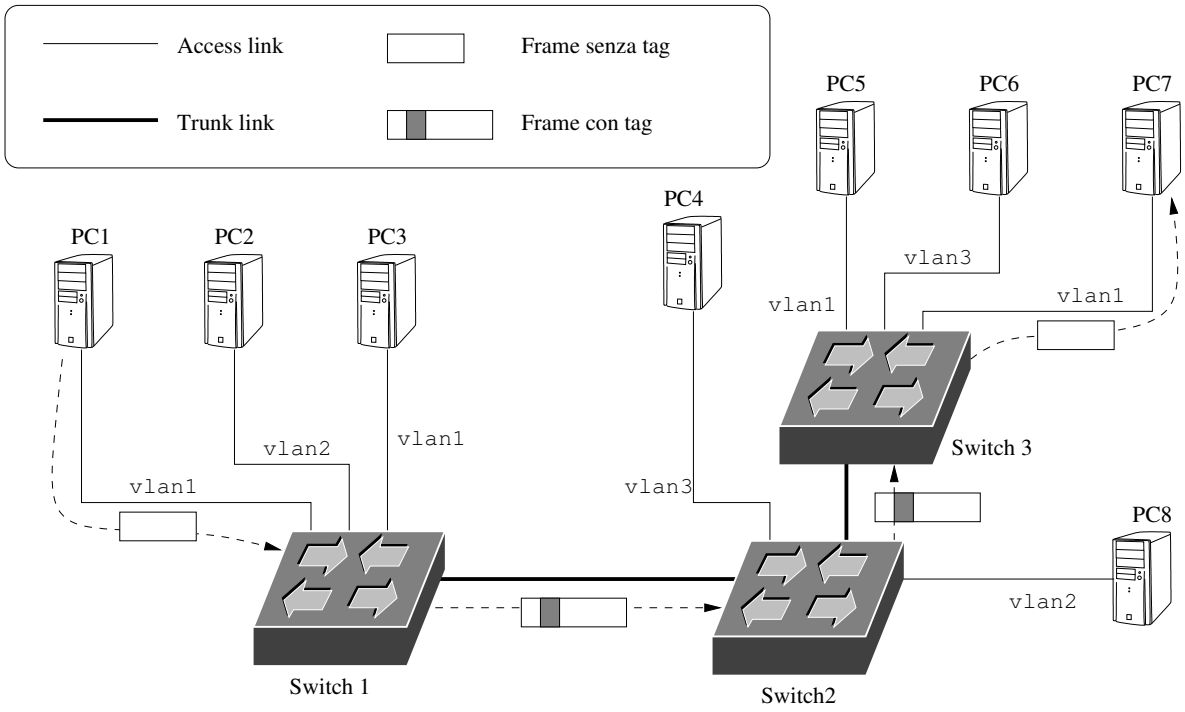



Figura 1.7: Transito di un pacchetto atraverso vari link.

# Capitolo 2

# Livello 3: Networking

## 2.1 Organizzazione delle sottoreti IP

Può accadere di avere a disposizione un intervallo di indirizzi IP contigui da suddividere in sottoreti in modo efficiente.

### 2.1.1 Un esempio

Si supponga di dover gestire la rete `192.168.10.0/24` in modo da accomodare tre sottoreti da un massimo di 10 host l'una.

La più piccola rete in grado di indirizzare almeno 10 host è la `/28`, con 4 bit di host (quindi in grado di distinguere 14 indirizzi host). Le tre sottoreti di cui abbiamo bisogno sono dunque:

- `192.168.10.0/28`, con indirizzi host da `.1` a `.14`, e `.15` come indirizzo di broadcast.
- `192.168.10.16/28`, con indirizzi host da `.17` a `.30`, e `.31` come indirizzo di broadcast.
- `192.168.10.32/28`, con indirizzi host da `.33` a `.46`, e `.47` come indirizzo di broadcast.

Ovviamente non è possibile utilizzare gli indirizzi rimanenti (da `.49` a `.255`) come se componessero un'unica sottorete. Infatti, il valore binario della rete successiva (`.48`, in binario `00110000`) termina con soli quattro zeri, quindi non permette una rete più ampia di una `/28`.

L'intervallo più vasto disponibile subito in seguito alle tre sottoreti indicate sopra è dunque

- `192.168.10.48/28`, con indirizzi host da `.49` a `.62`, e `.63` come indirizzo di broadcast.

La sottorete `.64` (in binario `01000000`) mette a disposizione 6 bit per indirizzare l'host, quindi la rete successiva è

- `192.168.10.64/26`, con indirizzi host da `.65` a `.126`, e `.127` come indirizzo di broadcast.

Infine, la sottorete che inizia da `.128` (binario `10000000`) permette di collocare il resto degli indirizzi disponibili:

- `192.168.10.128/25`, con indirizzi host da `.129` a `.254`, e `.255` come indirizzo di broadcast.

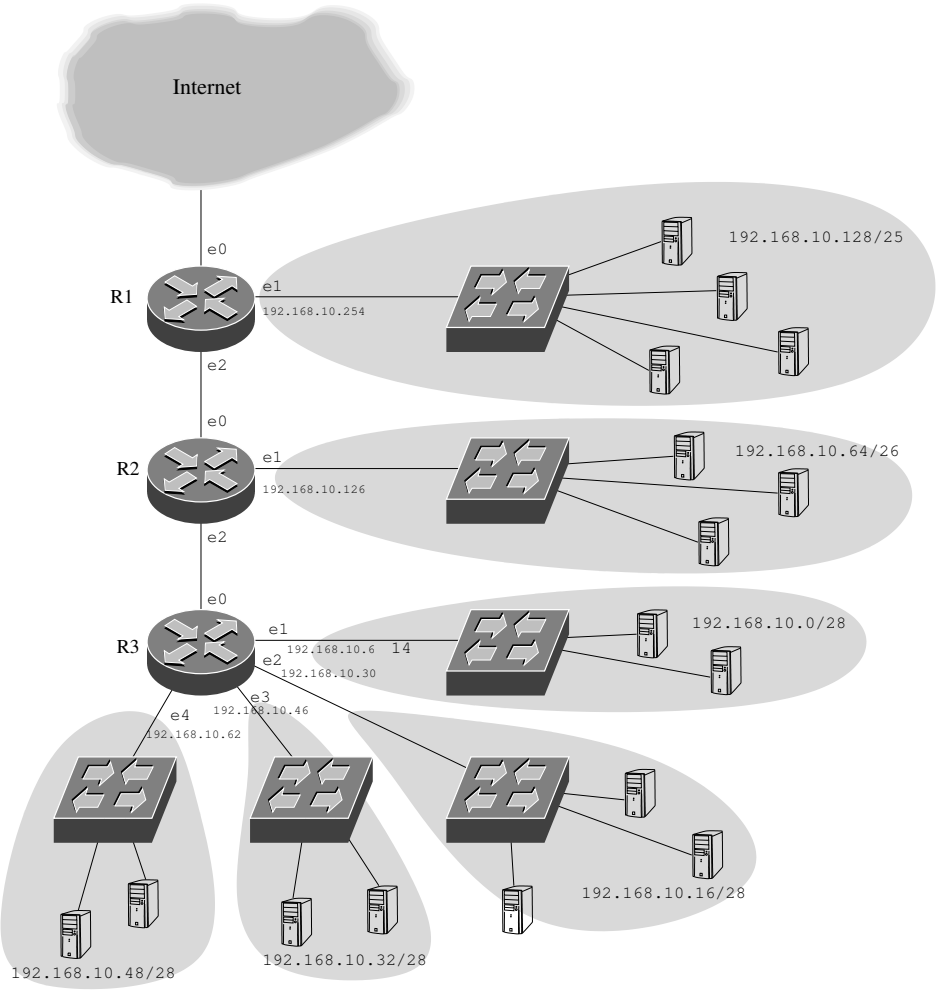



Figura 2.1: Mappatura fisica di sottoreti IP.

Router R1

| Subnet | Via |
|---|---|
| 192.168.10.128/25 | e1 |
| 192.168.10.0/25 | e2 |
| 0.0.0.0/0 | e0 |

Router R2

| Subnet | Via |
|---|---|
| 192.168.10.64/26 | e1 |
| 192.168.10.0/26 | e2 |
| 0.0.0.0/0 | e0 |

Router R3

| Subnet | Via |
|---|---|
| 192.168.10.0/28 | e1 |
| 192.168.10.16/28 | e2 |
| 192.168.10.32/28 | e3 |
| 192.168.10.48/28 | e4 |
| 0.0.0.0/0 | e0 |

Figura 2.2: Tabelle di instradamento per i tre router di Fig. 2.1.

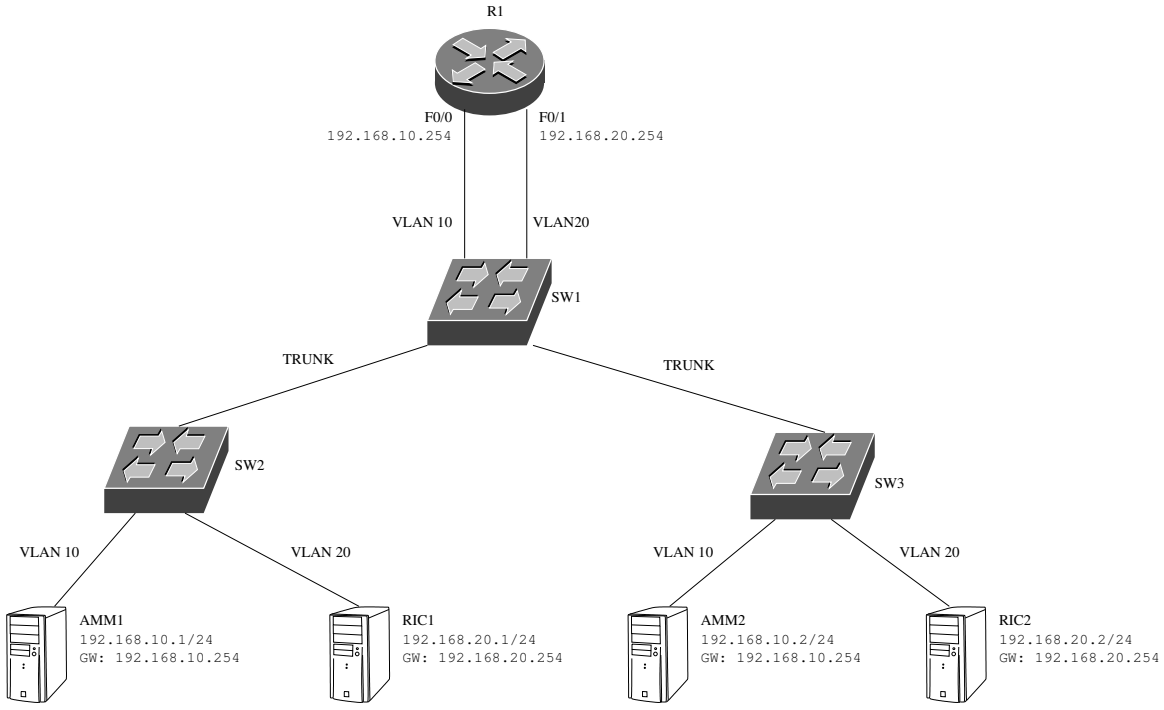



Figura 2.3: Routing fra due VLAN.

### 2.1.2 Mappatura in reti locali

È buona norma che la topologia della rete locale rispecchi il più possibile la struttura della suddivisione in sottoreti. Questo permette una maggior chiarezza nella progettazione e nel mantenimento, oltre a ridurre le informazioni necessarie a mantenere e a far operare la rete. Ad esempio, la struttura della suddivisione appena descritta potrebbe rispecchiarsi nella rete di Fig. 2.1.

Le tabelle di instradamento dei tre router potrebbero essere quelle rappresentate in Fig. 2.2. Si noti come, dal punto di vista di R1, le sottoreti gestite dai router R2 ed R3 non hanno motivo di essere distinte.

## 2.2 InterVLAN Routing

Abbiamo visto che host appartenenti a VLAN differenti non possono comunicare fra loro a livello 2 in quanto appartengono a LAN separate. È dunque necessario operare a livello 3 della pila ISO/OSI.

Soluzioni per routing tra VLAN:

1. Router a due porte
2. Router a una porta
3. Layer-3 switches: rendere gli switch un po' più intelligenti;

### 2.2.1 Router a due porte

La soluzione più ovvia e classica, ma normalmente poco efficace in termini di costo, è quella di Fig. 2.3: trattare le due VLAN come reti fisicamente separate e frapporre un router fra esse. Le due interfacce separate del router, `F0/0` e `F0/1`, agiscono da default gateway per le due VLAN.

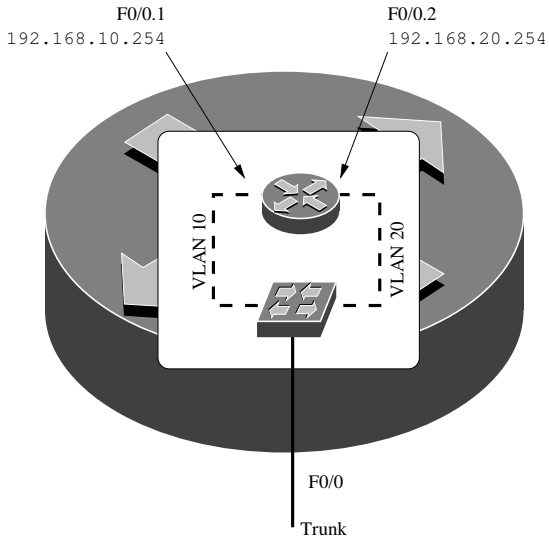



Figura 2.4: Router-on-a-stick.

### 2.2.2 Router a una porta

Altrimenti detta "router-on-a-stick" o "one-legged router", prevede un router collegato a un'unica linea trunk accessibile a entrambe le VLAN. Il router preleva i pacchetti a lui destinati (a livello 2, in quanto default gateway), opera le sostituzioni previste dalla routing table, li reimmette sulla stessa linea con il tag dell'altra rete.

In Fig. 2.4 si vede lo schema "logico" interno al router. L'unica interfaccia fisica `F0/0`, in modalità trunk, è separata internamente in due interfacce logiche `F0/0.1` e `F0/0.2`, ciascuna assegnata a una diversa VLAN e configurata come gateway per le due sottoreti.

### 2.2.3 Layer 3 switches

- Un router collega di norma sottoreti diverse ed ha tante schede di rete quante sono le sottoreti da collegare.
- Possiamo vedere lo switch di livello 3 come un router nel quale, al posto di una scheda di rete, abbiamo una VLAN alla quale è associato un certo indirizzo IP.
- Uno switch è detto di livello 3 quando, oltre a gestire normalmente diverse VLAN, è in grado di passare frame dall'una all'altra sulla base delle informazioni di livello 3 inserite nel frame.

Come si vede in Fig. 2.5:

- al posto della scheda di rete di un router, troviamo la VLAN identificata in base alle porte che la compongono;
- a ogni VLAN vengono associati un indirizzo IP ed una subnet mask, come fosse una scheda di rete; essa rappresenta il default gateway per tutte le porte associate;
- la tabella di routing dello switch di livello 3 consentirà quindi, analogamente a un router, l'inoltro dei pacchetti da una VLAN all'altra;
- le tabelle di routing sono del tutto eguali a quelle di un normale router; cambia solo la definizione dell'interfaccia fisica (es `vlan2` al posto di `eth2`).

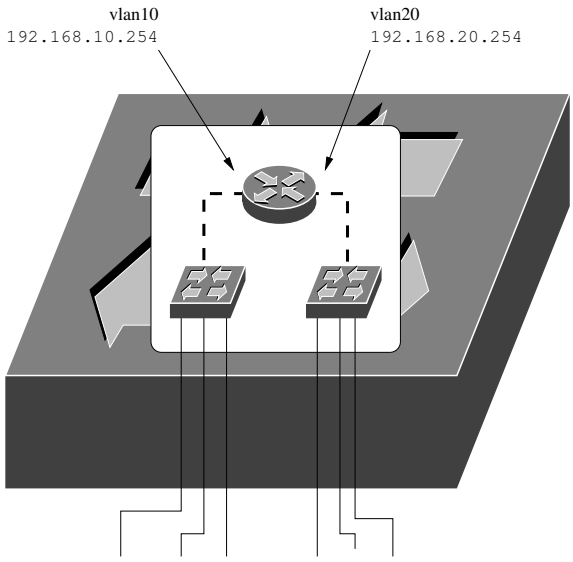



Figura 2.5: Switch di livello 3.

# Capitolo 3

# Livello 4: Trasporto

## 3.1 Network Address Translation (NAT)

### 3.1.1 Indirizzi privati

Il protocollo IP definisce alcuni intervalli come privati:

- `10.0.0.0/8`
- `172.16.0.0/12` diviso in 16 sottoreti:
  - `172.16.0.0/16`
  - ...
  - `172.31.0.0/16`
- `192.168.0.0/16` diviso in 256 sottoreti:
  - `192.168.0.0/24`
  - ...
  - `192.168.255.0/24`

Questi indirizzi non possono viaggiare fuori da una rete locale perché non corrispondono univocamente a degli host.

Come si vede in Fig. 3.1, la comunicazione è ovviamente possibile all'interno di una stessa rete locale, oppure se le due reti sono interconnesse all'interno di sottoreti gestite da una stessa entità, quindi esistono dei router che "conoscono" gli indirizzi privati.

Se la macchina di destinazione è remota, il pacchetto non arriva a destinazione, oppure non conosce la strada per tornare indietro (Fig. 3.2).

La soluzione, rappresentata in Fig. 3.3, è far sì che il router "impersoni" la macchina privata sostituendo il proprio indirizzo pubblico.

### 3.1.2 Caso base: connessione TCP

Supponiamo di avere i seguenti componenti in rete:

- Un client, con indirizzo di rete privato `saddr`.
- Un default gateway (router), con indirizzo di rete pubblico `raddr` (l'indirizzo di rete dal lato LAN non ci riguarda perché non apparirà mai sui pacchetti IP).

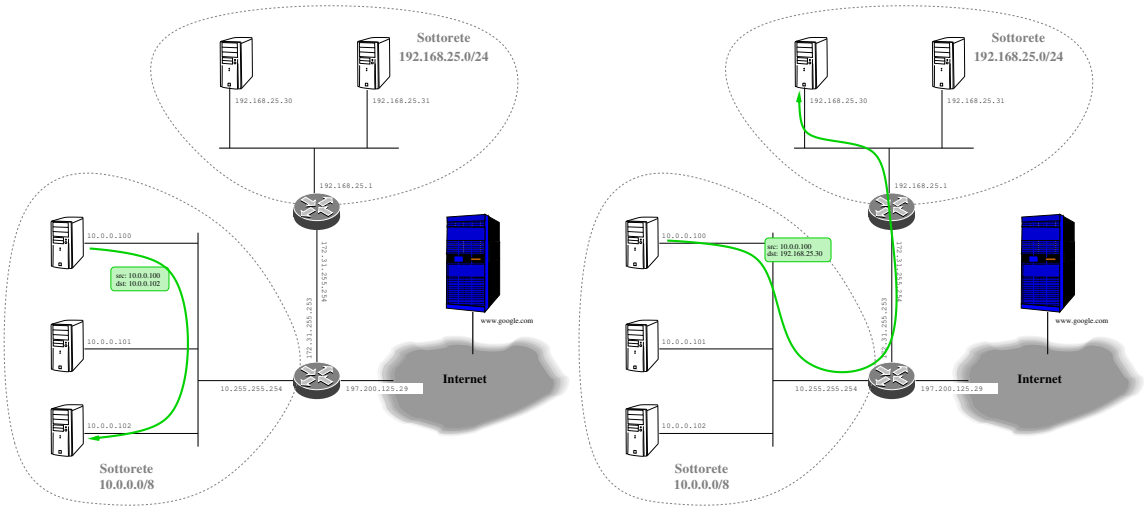



Figura 3.1: Comunicazione con IP privati all'interno di una stessa sottorete (sinistra) o fra reti appartenenti alla stessa unità di gestione (destra).

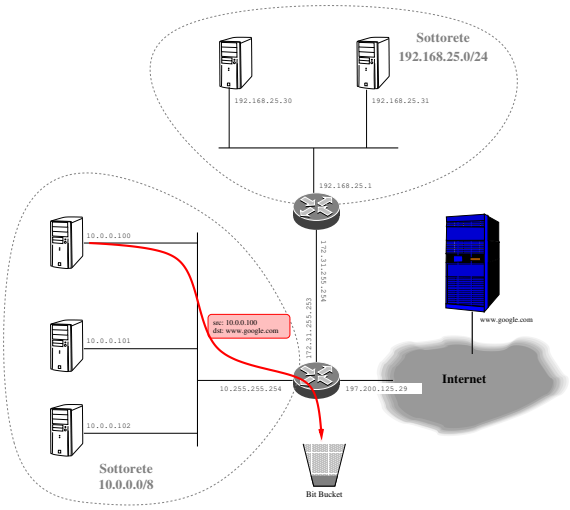



Figura 3.2: Comunicazione con IP privati verso una macchina esterna.

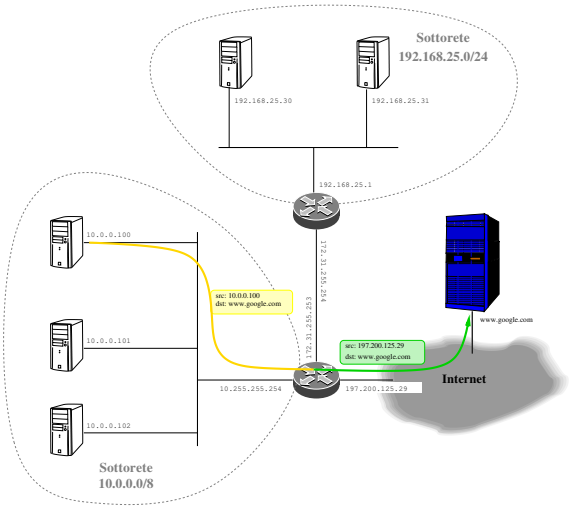



Figura 3.3: NAT: traduzione dell'IP privato.

- Un server remoto con indirizzo IP pubblico `saddr`.

Il server mantiene la seguente tabella a tre colonne, con almeno una riga per ogni connessione attiva:

| Porta del router | IP privato del client | Porta del client |
|---|---|---|
| | | |
| | | |

**Inizio della connessione**

- Il client invia il pacchetto `SYN(caddr,cport,saddr,sport)` verso il default gateway.
- Il default gateway individua una porta effimera `rport` e imposta la seguente tabella di traduzione:

| `rport` | `caddr` | `cport` |
|---|---|---|

- Il default gateway modifica la parte mittente del pacchetto: `SYN(raddr,rport,saddr,sport)`, dove `raddr` è il suo indirizzo pubblico, e lo invia verso l'esterno.

**Nota bene** — La traduzione, anche del solo indirizzo IP di destinazione, impone comunque di modificare la checksum TCP o UDP, quindi sono sempre necessarie modifiche a livello 4.

**Ulteriori pacchetti TCP in uscita**

- Il client invia il pacchetto `DATA(caddr,cport,saddr,sport)` verso il default gateway.
- Il default gateway ricava la porta in uscita in base alla riga pertinente della tabella di traduzione:

| `rport` | `caddr` | `cport` |
|---|---|---|

- Il default gateway modifica la parte mittente del pacchetto: `DATA(raddr,rport,saddr,sport)`, dove `raddr` è il suo indirizzo pubblico, e lo invia verso l'esterno.

**Pacchetti TCP in ingresso**

- Il server remoto invia il pacchetto `DATA(saddr,sport,raddr,rport)` verso il router, da esso ritenuto il vero client.
- Il default gateway ricava la vera destinazione in base alla riga pertinente della tabella di traduzione:

| rport | caddr | cport |
|---|---|---|

- Il default gateway modifica la destinazione del pacchetto: `DATA(saddr,sport,caddr,cport)`, e lo inoltra attraverso l'interfaccia di rete locale.

### 3.1.3 Connessioni in ingresso

Il problema: NAT scarta tutti i pacchetti in ingresso che non è in grado di mappare con la tabella di traduzione. Quindi, solo le connessioni in uscita riescono a funzionare.

Come si fa a contattare un host privato dalla rete pubblica?

Le soluzioni:

- Port forwarding
- TCP/UDP hole punching.

**Port forwarding**

- Utilizzata per abilitare servizi peer-to-peer o giochi in rete.
- Se l'host locale con indirizzo privato `caddr` ha un servizio in ascolto alla porta `cport`, è sufficiente impostare la seguente riga nella tabella di traduzione del router:

| Porta del router | IP privato del client | Porta del client |
|---|---|---|
| `cport` | `caddr` | `cport` |

  In tal modo, ogni pacchetto inviato alla porta `cport` del router viene inoltrato verso la stessa porta dell'host privato. In Fig. 3.4 si può vedere la forma di una tabella di port forwarding messa a disposizione da un router ADSL domestico.
- È ovviamente possibile differenziare la porta del router da quella del client, permettendo così di accedere dall'esterno allo stesso servizio su più client interni. Ad esempio, la seguente tabella consente di accedere al servizio SSH (porta `22`) dell'host `192.168.15.43` attraverso la porta `22` del router, e allo stesso servizio dell'host `192.168.15.228` attraverso la porta `2022` del router:

| Porta del router | IP privato del client | Porta del client |
|---|---|---|
| 22 | 192.168.15.43 | 22 |
| 2022 | 192.168.15.228 | 22 |

### 3.1.4 STUN: Session Traversal Utilities for NAT

Il problema: Un'applicazione ha bisogno di conoscere l'indirizzo IP "visto da fuori" per includerlo nei propri pacchetti.

La soluzione:

- Il client invia un pacchetto UDP verso uno STUN server remoto.
- Il server inserisce l'IP di provenienza del pacchetto, così come da lui rilevato, in un pacchetto di risposta.

Normalmente combinata con tecniche di NAT traversal per la raggiunguibilità di host privati.

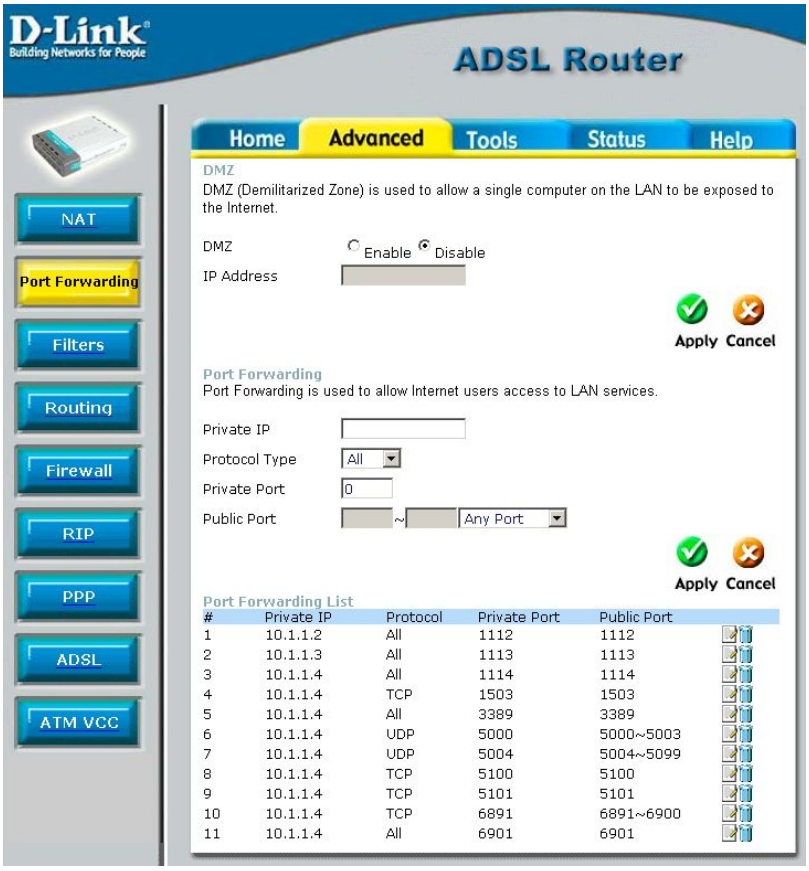
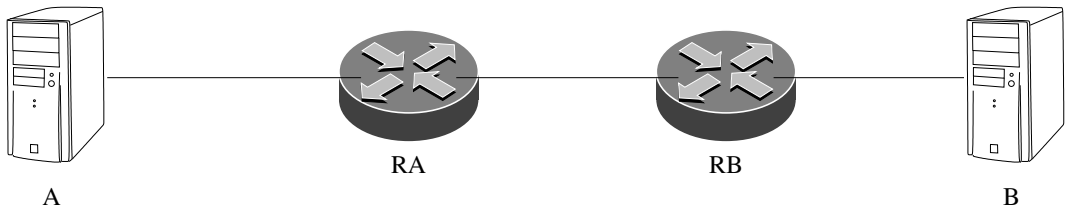



Figura 3.4: La tabella di port forwarding di un router ADSL

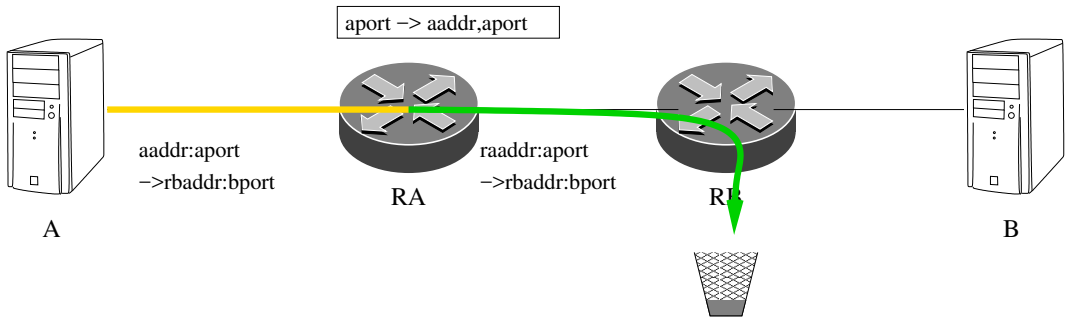
### UDP hole punching

Il problema è quello di "aprire la strada" ai pacchetti in ingresso inviando un opportuno pacchetto in uscita. Due client vogliono comunicare direttamente, entrambi hanno indirizzi IP privati e sono nascosti da NAT non configurabili; possiamo supporre che i client conoscano i rispettivi indirizzi pubblici (ad esempio, hanno già comunicato con un server comune).

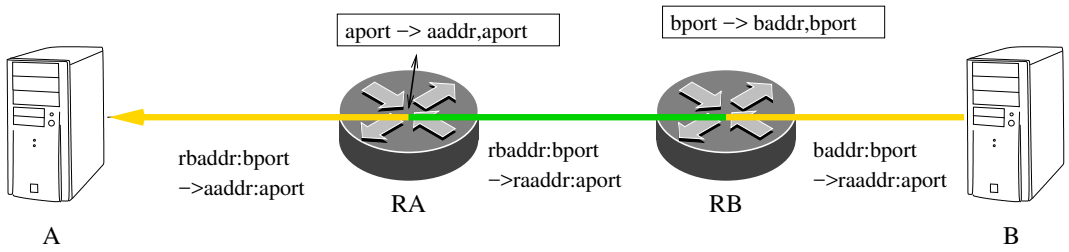
**Caso semplice:** supponiamo che i router NAT coinvolti preservino il numero di porta.



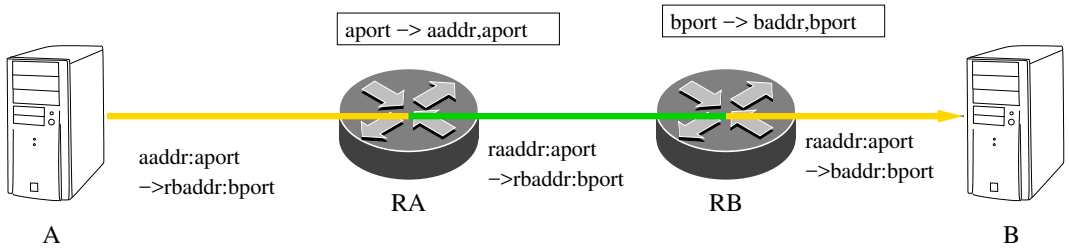

A prende l'iniziativa e spara un pacchetto verso il router di B:




Il pacchetto non arriva a B, perché il suo router non ha nulla nella sua tabella di traduzione, ma provoca l'aggiunta di una riga nella tabella di traduzione di A. B, poco dopo, fa la stessa cosa verso il router di A:




Questa volta, il pacchetto arriva ad A perché il suo router sa tradurlo. Inoltre, il pacchetto ha causato l'aggiornamento della tabella di traduzione di B. Ora, anche A può inviare pacchetti a B.




Problema: . . . E se il router non preserva le porte?

**Caso generale:** i router non preservano il numero di porta. In tal caso, con UDP, è possibile usare un server esterno "connivente" che comunica la porta esterna corrispondente. Le macchine A e B contattano, ad esempio, un server che restituisce a ciascuna il numero di porta usato dal router dell'altra.

# Capitolo 4

# Livello Applicazione: sicurezza delle reti

In questa serie di lezioni cercheremo di delineare gli scenari di attacco più comuni e di studiare qualche contromisura. Partiremo dall'ipotesi che la rete sia completamente aperta, con tutte le macchine dotate di un indirizzo IP pubblico (uno scenario "anni 90") e motiveremo l'aggiunta di misure di sicurezza.

## 4.1 Le fasi di un attacco hacker

### 4.1.1 Ricostruzione delle macchine presenti in una rete

Ogni attacco inizia con la raccolta di dati della rete obiettivo. Si parte da pochi dati pubblici a disposizione (un'URL web o email, ad esempio) e quindi, essenzialmente, da un nome di dominio.

Con particolari programmi (ad es. `DNSMap`), che combinano il nome del dominio con i nomi più comuni utilizzati per i server (es. `www`, `mail`, `ns`...), si cerca di individuare i principali indirizzi IP pubblici della rete oggetto dell'attacco.

È anche possibile chiedere al server DNS di autorità di un dominio di fornire il file di zona ("zone file"), in modo da avere una mappa completa dei nomi di dominio e delle associazioni con gli indirizzi IP.

**Contromisura** La maggior parte dei server DNS non fornisce più il file di zona.

### 4.1.2 Ricerca delle porte aperte

Come seconda fase, si cerca di ricostruire un elenco di porte aperte sui vari server attraverso varie tecniche di *port scanning*.

#### Port scan

Per cercare quali servizi TCP sono in ascolto, ad esempio, è sufficiente inviare un pacchetto `SYN` a ciascuna porta di ciascuna macchina della rete sotto attacco. Se la risposta consiste nel corrispondente pacchetta `SYN+ACK`, allora si può ipotizzare che l'applicazione corrispondente è in ascolto. Altrimenti, il computer risponderà con un pacchetto `RST`, oppure con un pacchetto ICMP *destination unreachable*.

L'utility `nmap`, ad esempio, ha esattamente questa funzione.

**Contromisura** Un filtro di rete (un dispositivo apposito, oppure un programma di firewall in funzione sul computer stesso) può facilmente capire se la macchina è soggetta a un port scan (tanti `SYN` verso porte diverse da uno stesso indirizzo IP). Può inoltre registrare l'IP mittente dei pacchetti e inviarlo a un amministratore per individuare l'attaccante.

#### Idle scan

Per evitare di essere scoperto, l'attaccante può appoggiarsi su altre macchine (dette zombie) alle quali non ha nemmeno bisogno di accedere. Sia A la macchina attaccante, B la macchina obiettivo.

A individua una macchina Z (lo zombie) con scarso traffico. Supponiamo che A voglia verificare la porta 25 di B.

**Prima fase** Inizialmente, A manda a Z un pacchetto TCP `SYN` verso una porta chiusa di Z. Z risponde ad A con un pacchetto `RST`. Quando lo riceve, A registra il campo *identification* dell'intestazione IP (chiamiamo $n$ il suo valore numerico).

**Seconda fase** A manda alla porta 25 di B un pacchetto `SYN`, indicando Z come mittente.

1. Se la porta 25 di B è aperta, B invia al mittente dichiarato nel pacchetto (cioè Z) un pacchetto `SYN+ACK`, alla cui ricezione Z replica con un pacchetto `RST` ("non so di cosa parli") verso B. Il campo `identification` dell'intestazione IP di quest'ultimo pacchetto vale $n + 1$, perché è il pacchetto IP successivo a quello emesso nella prima fase.
2. Se invece la porta 25 di B non è aperta, B invia a Z un pacchetto `RST`, che Z si limiterà a cestinare (non si risponde mai ai pacchetti `RST`), non inviando alcun pacchetto.

**Terza fase** A ripete quanto fatto nella prima fase. Se il campo `identification` dell'intestazione IP della risposta di B vale $n + 2$, allora A sa che nella seconda fase si è verificato il caso 1, e conclude che la porta 25 è aperta. Se invece il campo vale $n + 1$ conclude che la porta è chiusa, perché siamo nel caso 2.

La tecnica può essere applicata utilizzando molti zombie, in modo da dissimulare meglio l'attacco. Ovviamente, il presupposto è che i valori del campo *identification* di IP siano generati sequenzialmente, e che lo zombie non emetta altri pacchetti IP durante l'attacco (altrimenti si generano di falsi positivi).

**Contromisura** Quasi tutti i sistemi operativi moderni non generano campi *identification* sequenziali precisamente per questo motivo. Ciononostante, sono presenti in rete molte macchine non aggiornate ancora utilizzabili per questo genere di attacco.

### 4.1.3 Sfruttamento dei punti deboli di una macchina

Una volta identificate le porte aperte, attraverso una connessione TCP è spesso possibile sapere la versione del gestore del servizio (la maggior parte dei programmi di rete dichiara la propria versione per permettere alla controparte di adeguare il proprio protocollo).

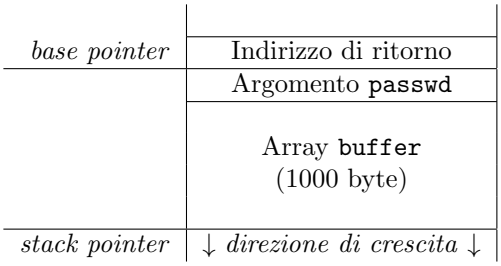
#### Attacchi di buffer overflow

L'errore di programmazione sfruttabile più facilmente è dovuto al sottodimensionamento dei buffer per la memorizzazione dei dati inviati in rete. Supponiamo ad esempio che la routine di verifica della password inizi con il seguente codice:

```
int verifica_password (const char *passwd)
{
char buffer[100];
```

```
strcpy (buffer, passwd);
......
}
```

Normalmente, le variabili locali sono allocate sul return stack del processo che, dopo la chiamata della funzione `check_password`, conterrà i seguenti dati:




Se l'attaccante riesce a inserire una password contenente più di 1000 caratteri, la funzione `strcpy`, che non prevede salvaguardie, deborderà dal buffer, arrivando a sovrascrivere l'indirizzo di ritorno della funzione. Al `return`, dunque, il controllo passerà a un indirizzo inserito ad arte dall'attaccante, che può arrivare in casi estremi ad assumere in controllo di una shell con permessi di superutente.

**Contromisure** Standard di codifica rigidi, linguaggi in cui non è possibile disabilitare il controllo dei limiti di un vettore. Inserimento nello stack di alcuni valori di controllo ("canarini") che se compromessi indicano un problema di buffer overflow in corso.

### Denial of Service

Un attacco abbastanza diffuso consiste nell'inviare molte richieste di connessione `SYN` a uno stesso servizio, inserendo mittenti fasulli. Ad ogni richiesta di connessione, il server deve allocare alcune risorse per gestire le fasi successive dell'handshake. Se le richieste arrivano a saturare le risorse disponibili, altre richieste legittime verranno ignorate (Denial of Service, DoS).

Si noti che le risorse allocate dal server verranno rilasciate dopo un timeout, oppure dopo il ricevimento di eventuali pacchetti `RST` che negano l'esistenza della richiesta, o di pacchetti ICMP che negano l'esistenza del mittente.

L'attacco è dunque tanto più efficace quante più richieste `SYN` possono essere inviate in breve tempo; a tale scopo, l'attaccante può appoggiarsi a molte macchine di cui ha preventivamente assunto il controllo tramite virus. In tal caso si parla di *Distributed* Denial of Service (DDoS).

**Contromisure** L'uso dei cosiddetti *SYN Cookies* permette al server di non allocare risorse al momento della ricezione del pacchetto `SYN`, ma di codificare le informazioni necessarie all'impostazione della connessione nel campo *sequence number* dell'intestazione TCP della risposta. Normalmente, il numero di sequenza è inizializzato con un valore casuale; in questo caso, invece, viene impostato con una timestamp, l'indicazione della MSS usata e un valore hash calcolato sulla base degli IP, delle porte, del timestamp e della MSS. Se l'handshake non prosegue, il server non ha allocato nulla; se invece l'handshake prosegue, allora il successivo pacchetto ricevuto dal server conterrà il SYN cookie (incrementato di 1) nel campo *Acknowledgment number*, e il server potrà verificare le informazioni di connessione e riservare le risorse necessarie.

### Attacco *smurf*

Consiste nell'inviare pacchetti ICMP *echo request* verso l'indirizzo di broadcast di una sottorete IP. la sottorete recapita il pacchetto a tutti gli host, i quali generano una risposta diretta al mittente. La "tempesta" di ping risultante può causare malfunzionamenti della rete. Inoltre, l'IP mittente dei

pacchetti originali può essere impostato verso la vera vittima dell'attacco, verso la quale puntano tutti i pacchetti di risposta. In quest'ultimo caso, la rete che riceve il ping agisce da "amplificatore".

**contromisure** Si configurano i router in modo da non inoltrare pacchetti con indirizzi di broadcast.

## 4.2 Esempi da approfondire

Esempi citati a lezione:

- Heartbleed (aprile 2014): buffer overflow in Openssl, la libreria di sicurezza più diffusa nei progetti open source:
  `https://en.wikipedia.org/wiki/Heartbleed`
- Controllo remoto di un'auto elettrica Nissan Leaf a causa di un'API non protetta (febbraio 2016):
  `https://www.troyhunt.com/controlling-vehicle-features-of-nissan/`
- Punti deboli nell'hardware e nella microprogrammazione delle CPU:
  - Accesso a memoria privilegiata:
    `https://en.wikipedia.org/wiki/Spectre_(security_vulnerability)`
    `https://en.wikipedia.org/wiki/Meltdown_(security_vulnerability)`
  - Corruzione di celle di memoria:
    `https://en.wikipedia.org/wiki/Row_hammer`

Molti esempi (per chi non si lascia spaventare dai dettagli tecnici) sono disponibili al blog del Project Zero di Google:

`https://googleprojectzero.blogspot.it/`

## 4.3 Firewall

Un firewall è un dispositivo fisico, oppure un'applicazione in esecuzione su una piattaforma di rete, in grado di ispezionare i pacchetti in transito e di decidere, sulla base di regole, se lasciarli passare.

È possibile classificare i firewall in modi diversi. Una prima classificazione riguarda il loro posizionamento:

- Personal firewall — un modulo del sistema operativo che filtra i pacchetti in ingresso e in uscita da un computer e decide se lasciarli passare sulla base di regole; serve alla protezione di una singola macchina. Esempi: Windows Firewall, iptables/netfilter.
- Network firewall — un dispositivo di rete (stand-alone, oppure aggregato a un router) che effettua la stessa operazione di filtraggio nel passaggio di pacchetti da un'interfaccia all'altra.

Talora un personal firewall può operare da firewall di rete se è in esecuzione su una macchina con due interfacce (*dual-homed*) che condivide il proprio accesso di rete con altre.

Un'altra classificazione utile riguarda la "profondità" dell'ispezione:

- Packet filtering — ispezione delle intestazioni di livello 3 e 4 di un pacchetto per decidere se lasciarlo passare o no;
- Stateful inspection — come sopra, ma mantenendo informazioni sullo stato delle connessioni aperte.
- Application proxy — blocco completo delle connessioni, consentite solo attraverso una particolare macchina che agisce da tramite.

La distinzione fra i vari tipi non è netta, e la terminologia non è necessariamente condivisa da tutti.

### 4.3.1 Packet filtering

Un Packet Filtering firewall filtra pacchetti sulla base degli indirizzi di origine e destinazione, ai livelli 3 (indirizzi IP) e 4 (porte). Normalmente le regole sono elencate in una lista, detta Access Control List (ACL). Per ogni pacchetto, la ACL viene scorsa sequenzialmente, e la prima regola adeguata viene applicata.

Supponiamo che una rete locale, `193.205.100.0/24` sia collegata all'esterno attraverso un router; in linea di principio, visto che la rete espone indirizzi pubblici, tutte le macchine possono ricevere e spedire pacchetti verso qualunque altra destinazione. Se il PC `193.205.100.10` in particolare ospita un server web, che dev'essere accessibile dall'esterno, possiamo impostare la seguente tabella per i pacchetti in ingresso (provenienti dall'esterno):

| Provenienza | Destinazione | Azione |
|---|---|---|
| `0.0.0.0/0` | `193.205.100.10/32` | PASS |
| `0.0.0.0/0` | `0.0.0.0/0` | DROP |

Le azioni "PASS" ("permit", "allow") e "DROP" ("deny", "reject") specificano se il pacchetto può essere inoltrato o no. I campi di provenienza e destinazione possono rappresentare sottoreti, oppure singoli host (identificato dalla maschera `/32`). Se più regole sono applicabili, vale la prima. L'ultima regola, detta "di default", è sempre applicabile (`0.0.0.0/0` corrisponde a qualunque indirizzo), e viene attivata tutte le volte che nessuna delle regole precedenti corrisponde a un pacchetto. In questo caso, tutti i pacchetti diretti verso il server web passano, tutti gli altri no.

Una tabella più dettagliata può specificare anche il protocollo e le porte, per bloccare tutti i tentativi di accesso non pertinenti al servizio web:

| Protocollo | Provenienza | Destinazione | Azione |
|---|---|---|---|
| TCP | `0.0.0.0/0:0` | `193.205.100.10/32:80` | PASS |
| * | `0.0.0.0/0:0` | `0.0.0.0/0:0` | DROP |

dove `:0` rappresenta tutte le porte, e "*" rappresenta tutti i protocolli.

In molti casi, il blocco indiscriminato delle altre destinazioni è troppo restrittivo, e vogliamo almeno permettere le connessioni TCP iniziate da macchine interne. Il modo più semplice è quello di consentire il transito verso le macchine interne per i pacchetti destinati verso porte non well-known (quindi probabilmente effimere):

| Protocollo | Provenienza | Destinazione | Azione |
|---|---|---|---|
| TCP | `0.0.0.0/0:0` | `193.205.100.10/32:80` | PASS |
| TCP | `0.0.0.0/0:0` | `193.205.100.0/24:>1023` | PASS |
| * | `0.0.0.0/0:0` | `0.0.0.0/0:0` | DROP |

Naturalmente, nulla impedisce a una macchina esterna di aprire una connessione verso una macchina interna in ascolto a una porta non well-known (ad esempio, un server database). Un'alternativa è l'aggiunta di una colonna in grado di specificare la presenza di eventuali flag:

| Protocollo | Provenienza | Destinazione | Flag | Azione |
|---|---|---|---|---|
| TCP | `0.0.0.0/0:0` | `193.205.100.10/32:80` | — | PASS |
| TCP | `0.0.0.0/0:0` | `193.205.100.0/24:>1023` | `ACK\|RST` | PASS |
| * | `0.0.0.0/0:0` | `0.0.0.0/0:0` | — | DROP |

La seconda regola impedisce l'ingresso di pacchetti `SYN` necessari ad aprire una connessione TCP. Tutti gli altri pacchetti TCP conterranno l'acknowledgment di un segmento precedente, quindi potranno passare.

Regole ulteriori permetteranno il passaggio di altri pacchetti di servizio, ad esempio ICMP.

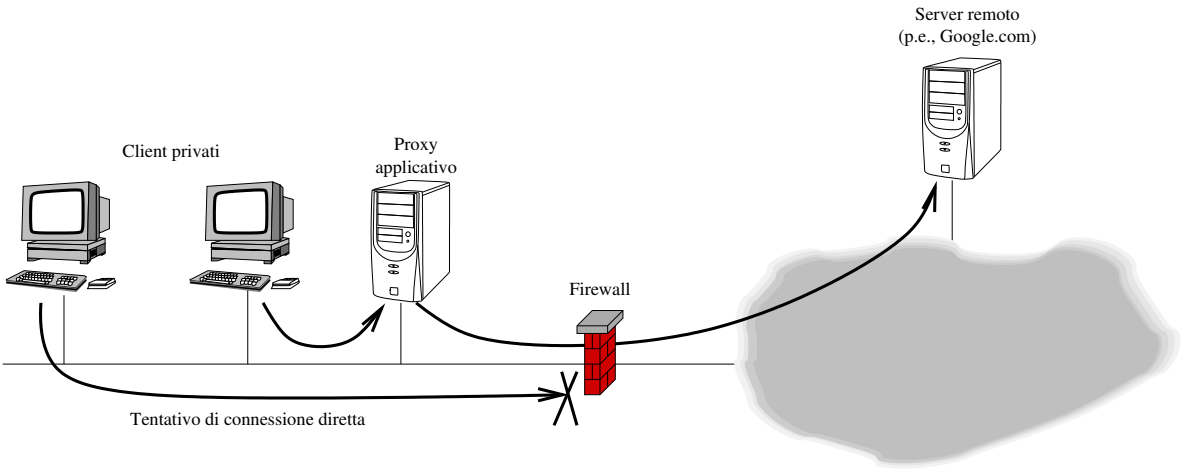



Figura 4.1: Solo il proxy può comunicare con l'esterno.

### 4.3.2 Stateful inspection

Un firewall di tipo Stateful inspection mantiene in una propria tabella interna lo stato delle connessioni TCP (e in subordine dei flussi UDP e ICMP) che vede transitare. Ad esempio, la riga 2 dell'ultima versione dell'ACL finora descritta potrebbe diventare:

| TCP | `0.0.0.0/0:0` | `193.205.100.0/24:>1023` | ESTABLISHED | PASS |
|---|---|---|---|---|

L'opzione "ESTABLISHED", presente ad esempio in `iptables`[1], lascia passare soltanto pacchetti in ingresso corrispondenti a connessioni TCP già stabilite. L'apertura e la chiusura di connessioni TCP viene dedotta dal firewall osservando il transito di pacchetti `SYN`, `FIN` e `RST`, unitamente all'uso di timer per lasciare scadere le connessioni non più utilizzate e non chiuse in modo appropriato. Per quanto riguarda il transito di pacchetti UDP, saranno utilizzati esclusivamente dei timer.

Il mantenimento dello stato è particolarmente utile nel caso di protocolli applicativi come FTP, in cui l'apertura di una connessione dall'esterno può essere richiesta dal client locale. Ispezionando il contenuto del protocollo, il firewall può decidere di lasciar passare proattivamente delle richieste di connessione provenienti dall'esterno.

### 4.3.3 Proxy applicativi

In alcuni casi, si ritiene opportuno impedire il passaggio di qualunque pacchetto tra la rete interna e l'esterno. È possibile individuare un singolo PC, opportunamente irrobustito, e delegare ad esso tutte le connessioni con l'esterno, come si vede in Fig. 4.1.

Per poter accedere a un servizio esterno, un'applicazione client, ad esempio un browser web, deve essere configurata in modo da dirigere tutte le richieste a uno specifico indirizzo IP e porta. La configurazione può essere impostata direttamente nell'applicazione, attraverso un opportuno file, o in un'opzione dedicata dal protocollo DHCP.

Il client invia la query al proxy utilizzando un protocollo simile a quello originario (ad esempio, una richiesta `GET` del protocollo HTTP rimarrà quas iinvariata, ma richiederà di specificare l'intera URL e non soltanto il percorso del file). Come si vede in Fig. 4.1, l'interazione fra il client e il server è spezzata in due connessioni. Il client è consapevole di comunicare con un proxy, mentre dal punto del server remoto il proxy non è diverso da qualunque altro client (anche se può identificarsi attraverso gli opportuni campi della richiesta).

[1] L'opzione "ESTABLISHED" esiste anche nel linguaggio di Cisco, però fa riferimento allsa presenza di flag `ACK` e `RST` menzionata prima.

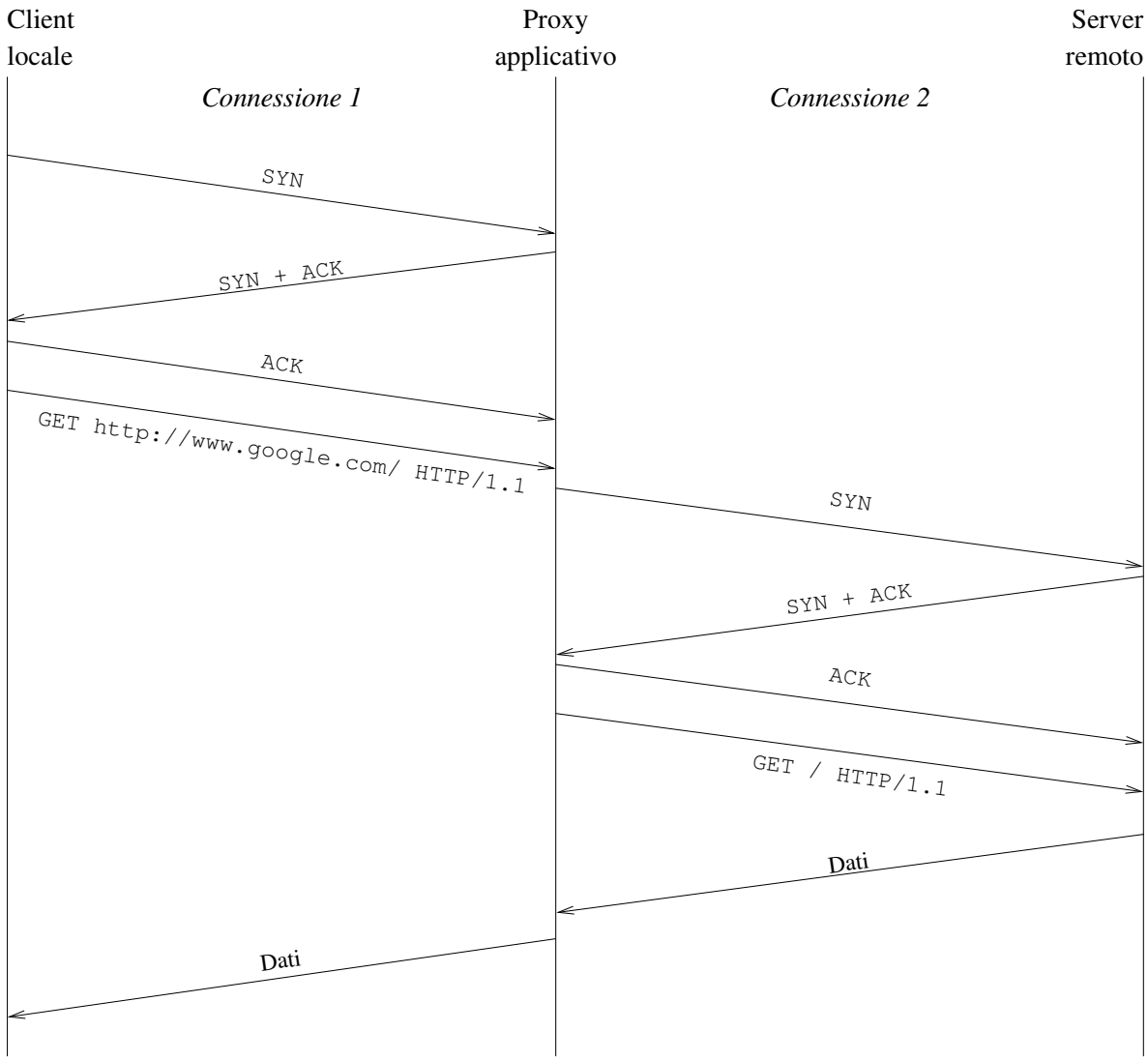



Figura 4.2: Separazione di una sessione web in due connessioni.

Le funzioni svolte da un application proxy possono essere molteplici:

- controllo delle informazioni trasmesse a livello applicativo (il proxy conosce il protocollo);
- registrazione delle connessioni e delle richieste (logging);
- caching delle richieste più comuni, con conseguente velocizzazione delle risposte;
- identificazione dell'utente, possibile attraverso un meccanismo di username e password.

Per contro, un application proxy può soccombere in condizioni di traffico elevato, in quanto può essere necessaria una potenza di elaborazione non indifferente, a seconda delle funzionalità richieste.

Infine, un proxy applicativo può essere usato, in modo trasparente, per gestire le connessioni provenienti dall'esterno verso un server web interno da proteggere. Un client esterno può collegarsi al proxy, che nei suoi confronti si comporta come fosse il server, ma che apre a sua volta una connessione con il vero server.

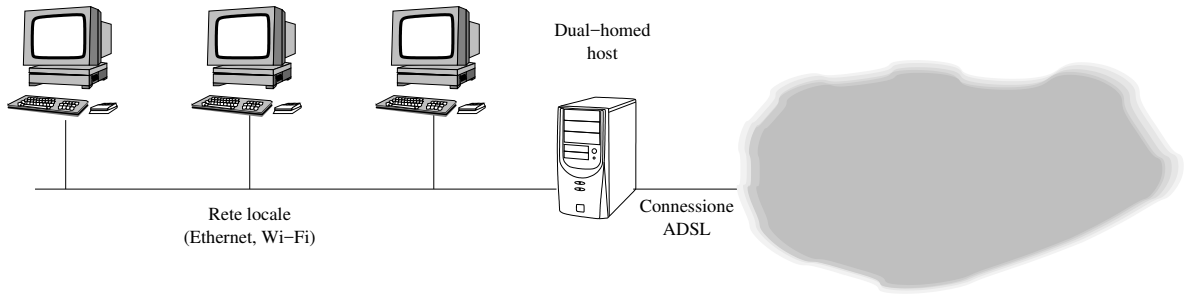



Figura 4.3: Configurazione con dual-homed host.

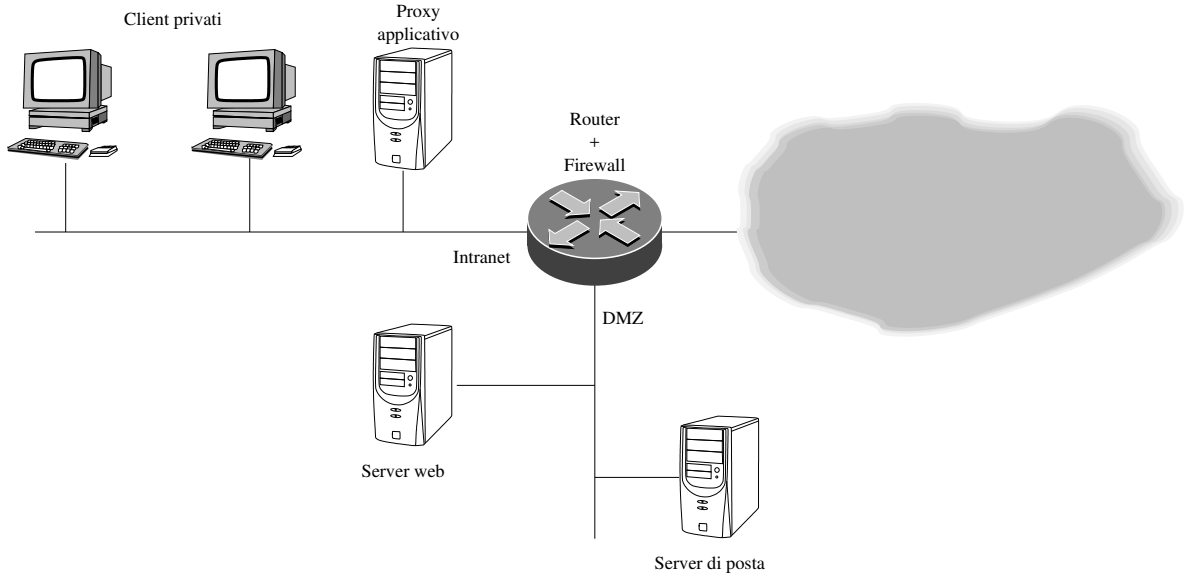



Figura 4.4: Three-legged architecture.

## 4.4 Configurazioni di rete

La configurazione migliore per garantire la sicurezza di una rete locale dipende sia dal livello di sicurezza desiderato, sia dal budget disponibile.

### 4.4.1 Reti Small Office / Home Office (SOHO)

Per budget particolarmente ristretti (piccoli uffici, residenze) è possibile, come in Fig. 4.3, fornire un PC con due schede di rete (ad esempio, una scheda Ethernet e una scheda ADSL), e utilizzarlo come firewall (ad esempio usandolo sotto Linux con iptables, eventualmente configurato con qualche utility di alto livello come Shorewall). Il PC può ospitare anche servizi di frontiera, ad esempio un proxy applicativo.

Se si ha a disposizione un router con funzioni di firewall, è possibile realizzare la cosiddetta "three-legged architecture" (Fig. 4.4) in cui la rete locale è divisa in due zone:

- un'Intranet vera e propria, molto protetta, con accessi molto limitati da e per l'esterno;
- una "zona demilitarizzata" (Demilitarized Zone, DMZ) in cui risiedono server che richiedono una maggiore libertà d'accesso, ma che devono essere più affidabili e sicuri.

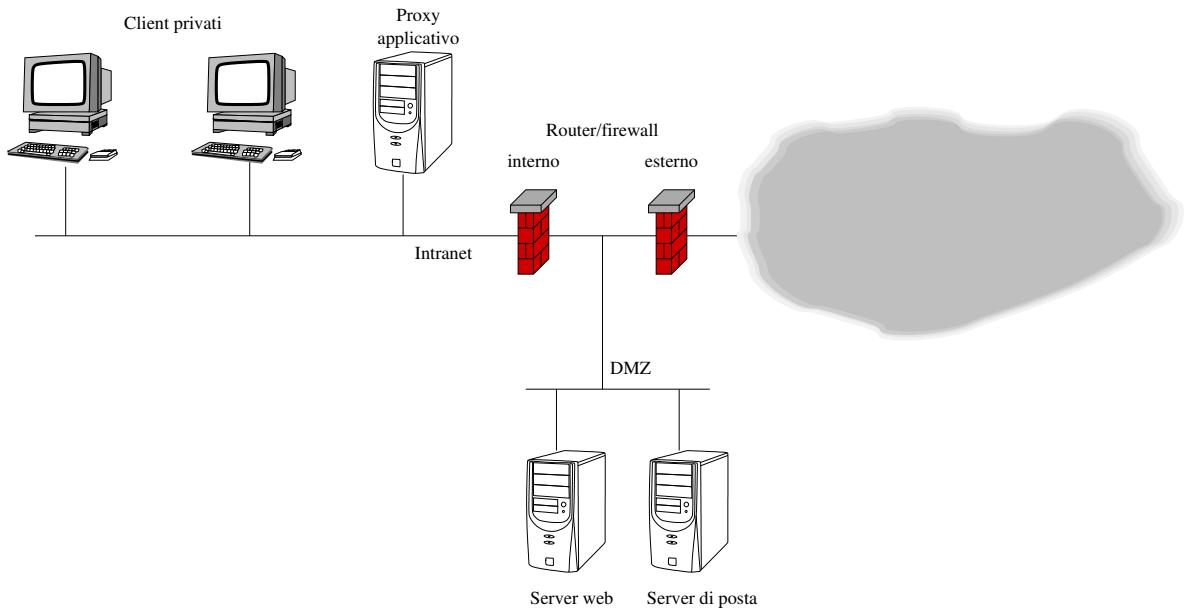



Figura 4.5: Architettura a più livelli di schermatura.

### 4.4.2 Reti più complesse

Nel caso di budget più elevati, è possibile estendere la protezione installando un firewall all'ingresso di ciascuna zona. In tal modo, l'accesso ai PC più protetti richiede l'abbattimento di due diverse linee di difesa.

### 4.4.3 Irrobustimento dei server

I server esposti nella DMZ richiedono una particolare attenzione per quanto riguarda la robustezza dei servizi offerti. Il processo col quale si chiudono tutte le porte non utilizzate e si limitano i servizi all'essenziale, possibilmente con l'aggiunta di un personal firewall che lasci cadere tutti i pacchetti non direttamente correlabili a uno dei servizi offerti, si chiama *hardening* (irrobustimento).

# Capitolo 5

# Crittografia

## 5.1 Motivazioni

- Consideriamo uno scenario nel quale Alice voglia effettuare delle operazioni di Internet banking con la sua banca (Bob).
- Alice deve innanzitutto autenticarsi inviando le proprie credenziali (nome utente e password). Non desidera inoltre che dati sensibili vengano conosciuti.
- Se queste informazioni sono inviate in chiaro possono essere intercettate e chi riesce a carpire queste informazioni può, da quel momento in poi, accedere senza problemi al conto di Alice.
- Occorre quindi individuare delle misure di sicurezza che consentano di cifrare le informazioni trasmesse.

Con il termine "cifratura" si intende l'uso di un procedimento matematico che consente, mediante un apposito algoritmo di cifratura (encryption algorithm), la modifica del messaggio, prima della sua trasmissione.

Anche se il messaggio venisse intercettato, non si riuscirebbe a interpretarlo, perchè solo il destinatario è in grado di applicare l'opportuno algoritmo di decifratura (decryption algorithm), che riesce a riportare il messaggio al suo valore originario.

## 5.2 Definizioni

- Generalmente il testo in chiaro ("plaintext") e il testo cifrato ("ciphertext") si indicano rispettivamente con le lettere $m$ (come "messaggio") e $c$ (come "codice"); la chiave con il simbolo $k$ ("key").
- La funzione di cifratura viene indicata con il simbolo $f$, con $f^{-1}$ quella di decifratura. Si scrive pertanto $c = f_k(m)$, e analogamente $m = f_k^{-1}(c)$.
- La funzione $f$ è necessariamente *iniettiva*: se due messaggi $m_1$ ed $m_2$ dessero origine allo stesso codice $c = f_k(m_1) = f_k(m_2)$, esso non sarebbe riconducibile in modo univoco a un messaggio e la funzione $f^{-1}$ non sarebbe ben definita.
- Una funzione di cifratura si dice simmetrica se

  $$f_k^{-1}(f_k(m)) = m,$$

  ossia se la chiave usata per cifrare e decifrare è la stessa.

Un principio molto importante in crittoanalisi è il seguente:

### 5.2.1 Principio di Kerckhoffs

> È necessario che il sistema non richieda segretezza, e che possa senza problemi cadere in mano nemica[1].

Traduzione: la robustezza deve stare tutta nella chiave; la scoperta della funzione $f$ da parte nemica non deve compromettere il sistema.

Oppure: "Il nemico conosce il sistema" (Claude Shannon).
Il principio di Kerckhoffs ha molte ragioni d'essere:

- Un algoritmo noto e studiato offre più garanzie di uno tenuto nascosto, quindi mai passato per le mani di crittoanalisti seri.
- La "Security through obscurity" non ha mai retto un'analisi approfondita.
- La chiave è generalmente più semplice e può essere cambiata più frequentemente dell'algoritmo.
- Esistono ormai tecniche di comprovata efficienza, è assurdo crearne di nuove per ogni applicazione (anche se ovviamente la ricerca prosegue).

### 5.2.2 Cifrari dimostrabilmente sicuri

Esiste un cifrario assolutamente sicuro: è il cosiddetto "One-time pad".

Formalmente:
Supponiamo che Alice e Bob condividano una lunghissima (al limite infinita) sequenza $K$ di bit generati in modo casuale:

$$K \in \{0,1\}^{\mathbb{N}}.$$

Se Alice deve mandare una stringa binaria $m$ di lunghezza $L$: $m \in \{0,1\}^L$ a Bob, la codificherà effettuando un OR esclusivo con i primi $L$ bit di $K$:

$$c = m \oplus K_{1\ldots L}.$$

Charlie, che intercetta, non vede altro che una sequenza di bit casuali, quindi non è in grado di decifrare il messaggio.

Bob, che condivide la sequenza $K$, può roittenere il messaggio rieseguendo l'OR esclusivo con gli stessi bit della chiave:

$$m = c \oplus K_{1\ldots L}.$$

Ogni volta che una parte della chiave è stata usata, questa viene eliminata, quindi la codifica di un altro messaggio $m' \in \{0,1\}^L$ utilizzerà gli $L$ bit successivi della chiave: $c' = m' \oplus K_{L+1\ldots 2L}$.

Il metodo era usato in pratica nella versione di Vernam[2]: le due parti in comunicazione condividono un blocco (pad) di chiavi di sostituzione alfabetica generate con un procedimento casuale, e cambiano la chiave ad ogni lettera. Per la trasmissione di un messaggio si utilizza un foglio del blocco, il quale viene poi distrutto. Ovviamente, il blocco non deve cadere in mano nemica.

#### Problemi

- La chiave è lunga quanto il messaggio (condizione necessaria per la sicurezza dimostrata da Shannon, ma scomoda).
- Mittente e destinatario devono essere sincronizzati (essere sicuri di partire dalla stessa pagina del blocco).
- Difficoltà a diffondere il blocco chiave con sicurezza.

[1] Auguste Kerckhoffs, *La Cryptographie Militaire* (1883)
`https://en.wikipedia.org/wiki/Kerckhoffs%27s_principle`

[2] `https://en.wikipedia.org/wiki/One-time_pad`

### 5.2.3 Categorie

Gli algoritmi di cifratura moderni si possono dividere in due categorie:

**Block ciphers** Il messaggio viene diviso in blocchi, ogni blocco subisce la stessa trasformazione (a meno di una variazione di chiave).

**Stream ciphers** Cerca di riprodurre i vantaggi del one-time pad: ogni elemento del messaggio viene combinato (ad esempio tramite XOR) con un elemento proveniente da un flusso pseudocasuale, spesso generato a partire da una chiave iniziale (detta vettore di inizializzazione). In alcuni casi, il flusso casuale può dipendere dagli input precedenti.

### 5.2.4 Attacchi

Un algoritmo di cifratura può essere attaccato in molti modi diversi:

#### Forza bruta

Un attaccante in possesso del messaggio cifrato $c$ può provare a decodificarlo sistematicamente con tutte le chiavi $k$ possibili finché il messaggio ottenuto non risulta intelleggibile. Un'ovvia debolezza del metodo è il numero di chiavi da provare, spesso proibitivo; inoltre, in alcuni metodi (come OTP visto prima) al variare della chiave può venir ottenuto qualunque messaggio.

#### Crittanalisi

Osservando piu codici, oppure venendo a conoscenza di una o più coppie messaggio-codice, è talora possibile restringere il campo di ricerca della chiave.

#### Crittanalisi "col tubo di gomma" (*rubber-hose cryptanalysis*)

Estorsione della chiave con metodi violenti — spesso tristemente più economica ed efficace della crittanalisi matematica.

#### Ingegneria sociale (*social engineering*)

Circuire il detentore della chiave con metodi non violenti. Esempio: phishing.

## 5.3 Condivisione della chiave: Diffie-Hellman

Il problema più grave negli algoritmi appena visti è quello di scambiare la chiave fra le due parti Alice e Bob senza che questa venga intercettata da Charlie.

Uno scambio *brevi manu* è sempre l'opzione più sicura, ma spesso non è fattibile.

Possibili soluzioni: inviare pezzi di chiave attraverso canali diversi (posta+email+SMS), nascondere la chiave in un testo (steganografia, ma si ricade nel problema della *security through obscurity*).

L'algoritmo di Diffie-Hellman permette la crittografia simmetrica senza scambio delle chiavi[3].

[3] `https://en.wikipedia.org/wiki/Diffie-Hellman_key_exchange`

| Alice | | Bob |
|---|---|---|
| $p = 23,\ g = 5$ | | |
| $a = 6$ | | |
| $A = 5^6 \equiv 8$ | $\rightarrow (p, g, A) \rightarrow$ | $p = 23, g = 5, A = 8$ |
| | | $b = 15$ |
| | $\leftarrow B \leftarrow$ | $B = 5^{15} \equiv 19$ |
| $K = B^a = 19^6 \equiv 2$ | | $K = A^b = 8^{15} \equiv 2$ |

Figura 5.1: Il protocollo Diffie-Hellman esemplificato con valori piccoli.

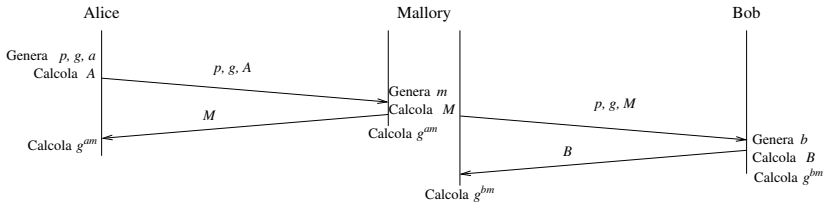



Figura 5.2: Un attacco Man-in-the-Middle al protocollo Diffie-Hellman

### 5.3.1 L'algoritmo

L'algoritmo si basa sulla difficoltà di invertire l'elevamento a potenza modulo un numero primo (grande). Il problema di invertire tale operazione è detto *Problema del Logaritmo Discreto*.

- Alice decide un numero primo $p$ molto grande e un intero $g < p$.
- Alice genera un numero $a < p$, segretamente; calcola il numero $A = g^a \pmod p$ e trasmette pubblicamente $p$, $g$ ed $A$ a Bob.
- Bob genera un numero $b < p$; calcola il numero $B = g^b \pmod p$ e trasmette $B$ ad Alice.
- Ora entrambi possono calcolare il numero $K = g^{ab} \pmod p$ utilizzando le comuni proprietà delle potenze:
  - Alice lo calcola come $K = B^a \pmod p$;
  - Bob lo calcola come $K = A^b \pmod p$;
- Charlie, pur conoscendo $p$, $g$, $A$ e $B$, non è in grado di ricavare né $a$ né $b$ (troppo difficile), quindi non può calcolare $K$.

Una corretta esecuzione dell'algoritmo richiede alcuni passi in apparenza molto complessi, ma che sono risolvibili in tempo polinomiale rispetto alla dimensione dei numeri in gioco. La determinazione di un grande numero primo $p$ (tipicamente dell'ordine delle migliaia di bit) è risolvibile attraverso tecniche di teoria dei numeri. Inoltre, non tutti i numeri $g$ sono ugualmente sicuri: solitamente, si richiede che $g$ sia una *radice primitiva* modulo $p$, ossia un numero $g$ con la proprietà che i valori $g^i \pmod p$ sono tutti distinti per $i = 1, \ldots, p-1$. In Fig. 5.1 troviamo un esempio con numeri piccoli.

Non è inoltre necessario che Alice determini valori diversi di $p$ e $g$: è possibile che un particolare protocollo li fissi una volta per tutte.

### 5.3.2 Suscettibilità agli attacchi Man-in-the-Middle

L'algoritmo Diffie-Hellman non autentica gli interlocutori, è perciò sensibile ad un attacco di tipo "Man in the Middle", in cui un attaccante (Mallory), in grado di interporsi fra Alice e Bob impersona Bob per Alice e viceversa, come mostrato in Fig. 5.2. Come si vede in Fig. 5.2, Mallory genera un proprio


Figura 5.3: La contromisura di Telegram a possibili attacchi MitM da parte dei server: un'immagine generata a partire dalla chiave.

segreto $m < p$ e lo calcola il valore pubblico $M = g^m \pmod p$. Sostituisce $M$ ad $A$ al messaggio di Alice prima di inoltrarlo a Bob, fingendosi così Alice nei confronti di Bob; inoltre, invia $M$ ad Alice fingendosi Bob.

A questo punto, Alice può calcolare la chiave condivisa $K_A = M^a \pmod p$; crede di condividerla con Bob, ma in realtà la condivide con Mallory, che calcola lo stesso valore usando il proprio segreto: $K_A = A^m \pmod p$. Allo stesso modo, la chiave calcolata da Bob, $K_B = M^b \pmod p$, è condivisa con Mallory, che la calcola come $K_B = B^m \pmod p$.

### Contromisure

Si noti che Alice e Bob si ritrovano con due chiavi diverse. Per evitare attacchi "Man-in-the Middle" si possono usare tecniche ad hoc. Ad esempio, in un canale VoIP Alice e Bob generano un breve hash della chiave e lo leggono ad alta voce. Se sono diversi, allora c'è un intruso. Oppure cercheranno di confrontare un piccolo sottoinsieme della chiave attraverso un canale diverso da quello con cui l'hanno creata, confidando nel fatto che Mallory non controlla tutte le comunicazioni.
Una volta che Alice e Bob hanno istituito una prima chiave condivisa sicura $K_{\text{AB}}$, non è necessario che la usino per codificare tutta la comunicazione col rischio che venga compromessa da troppi casi d'uso.

Ad esempio, il popolare client di chat Telegram (e ultimamente anche WhatsApp) permette di istituire una chiave condivisa Diffie-Hellman, e per garantire la sua sicurezza permette ad Alice e a Bob di generare un'immagine basata su una funzione hash della chiave, come nell'esempio di Fig. 5.3; Alice e Bob possono scambiare le immagini attraverso un programma esterno (posta elettronica, oppure di persona) e assicurarsi che la chiave è la stessa.

## 5.4 Funzioni hash crittografiche

La crittografia non si occupa solamente del problema di trasmettere informazioni in modo confidenziale. Un problema quasi altrettanto importante è quello dell'integrità: Alice trasmette un messaggio $m$, non necessariamente riservato, oppure lo pubblica; Bob lo scarica, e vuole assicurarsi che il messaggio che lui legge sia esattamente quello pubblicato da Alice. Una soluzione completa al problema, come vedremo più avanti, è costituita dalla *firma digitale*, un elemento fondamentale della quale sono le cosiddette *funzioni hash crittografiche*, che possono essere utilizzate anche separatamente.

Come esempio, supponiamo che Alice abbia realizzato un'applicazione e ne voglia distribuire il file di installazione (ad esempio un file zip contenente gli eseguibili). Chiamiamo $m$ il file di Alice. Dal nostro punto di vista, $m$ è una stringa di simboli in un alfabeto $\Sigma$ (una sequenza di byte). Se $m$ è molto

grande, Alice ricorre a una Content Distribution Network (CDN, rete di distribuzione di contenuti) con mirror distribuiti (ad esempio, SourceForge).

Se Bob vuole scaricare il file di Alice, viene diretto verso il server a lui più prossimo. Se però Charlie riesce a compromettere quel server, può sostituire il file $m$ con una versione $m'$ in cui ha inserito un virus. Come fa Bob a fidarsi della CDN?

Alice può ovviare al problema utilizzando una *funzione hash*

$$H : \Sigma^* \to \{0 \ldots, N-1\}.$$

La funzione, applicata a una qualunque stringa $\Sigma^*$, restituisce un intero compreso fra 0 e $N-1$. Alice applica la funzione al proprio file $m$, ottenendo il valore $h = H(z)$, detto "riassunto" (digest) o "impronta digitale" (fingreprint), e lo pubblica sulla sua pagina web, che è sotto il suo diretto controllo, quindi più affidabile. Si noti che Alice preferisce non pubblicare direttamente $m$ sul suo sito web perché ha bisogno di una distribuzione efficiente.

Quando Bob scarica $m$ dalla CDN, legge anche il valore $m$ dal sito di Alice e può controllare se $h = H(m)$.

Charlie ha quindi bisogno di realizzare una versione $m'$ che non solo contenga il virus, ma il cui valore attraverso la funzione di hash sia lo stesso della versione originale, $h$; vuole attuare un *attacco di collisione*. Ci sono due modi per farlo:

1. generare tante varianti $m'_1, m'_2, \ldots$, ad esempio aggiungendo un file allo zip contenente un valore casuale, finché una di queste varianti, tutte pienamente funzionali e infette, ha lo stesso valore hash di $m$ (attacco *brute force*);
2. esaminare la funzione $H$ per capire dove agire per conservare il valore corretto (attacco analitico).

Le principali funzioni hash classiche, utilizzate per la realizzazione di dizionari, sono passibili di entrambi gli attacchi:

1. i loro codomini sono piccoli (il valore di $N$ non è elevato), quindi dopo $O(N)$ tentativi casuali si arriva a trovare una collisione;
2. sono basate su semplici funzioni aritmetico/logiche e su poche operazioni logiche (ad esempio XOR), quindi è spesso possibile "invertire" l'effetto delle modifiche agendo su porzioni inutilizzate del file zip.

Per poter garantire la sicurezza necessaria ad Alice e Bob, le funzioni hash debbono essere resistenti agli attacchi di collisione. In generale, una funzione hash è detta "crittografica" quando gode delle seguenti proprietà:

- resistenza agli attacchi di collisione: dato $m$, è difficile generare $m'$ tale che $H(m) = H(m')$;
- resistenza agli attacchi di preimmagine: dato un valore hash $h$, è difficile trovare una stringa $m$ tale che $H(m) = h$;
- resistenza agli attacchi di collisione (versione 2): è difficile trovare due stringhe $m_1$ e $m_2$ tali che $H(m_1) = H(m_2)$.

Le funzioni hash crittografiche più diffuse sono MD5 ("Message Digest 5"), che genera un valore a 128 bit, e le funzioni della famiglia SHA ("Secure Hash Algorithm"): SHA-1 (160 bit), SHA-2 (da 224 a 512 bit). Si basano su applicazioni di funzioni aritmetiche e logiche, scambi di blocchi di bit e permutazioni, ripetute molte volte.

Le funzioni MD5 e SHA-1 sono ormai considerate insicure, e alcune collisioni sono state trovate e dimostrate[4]

[4]Per MD5 c'è l'imbarazzo della scelta:
`https://natmchugh.blogspot.it/2014/10/how-i-created-two-images-with-same-md5.html`
`http://www.mathstat.dal.ca/~selinger/md5collision/`
`https://twitter.com/bascule/status/838927719534477312`
Per SHA-1, la scelta è per ora più limitata, la prima collisione risale al febbraio 2017:
`https://security.googleblog.com/2017/02/announcing-first-sha1-collision.html`

# Parte II

# Domande ed esercizi

# Appendice A

# Domande teoriche e spunti di riflessione

Questo capitolo raccoglie alcune domande di autovalutazione per verificare autonomamente la comprensione degli argomenti del corso.

## A.1 Livello Data Link

### A.1.1 Prerequisiti

- Sono in grado di elencare e, nei limiti del possibile, motivare i livelli della pila protocollare ISO/OSI?
- E la pila TCP/IP?
- Quali informazioni sono contenute nell'intestazione di un frame Ethernet?
- Perché l'indirizzamento su Internet non si basa direttamente sui MAC address del livello 2?
- A cosa serve la distinzione DTE-DCE?
- Qual è il ruolo di ciascun dispositivo di rete (di livello 1, 2 e 3)?
- Cosa si intende per dominio di collisione? E di broadcast?

### A.1.2 LAN virtuali

- Quali vantaggi comporta l'uso delle LAN virtuali?
- È possibile utilizzare dispositivi di livello 1 e 2 che non "conoscono" le LAN virtuali all'interno di una rete locale suddivisa in VLAN?
- A cosa si riferisce la distinzione fra modalità "access" e modalità "trunk"? Alla porta di un dispositivo, al collegamento, al tipo di cavo...
- Come funziona l'incapsulamento 802.1Q? Quante VLAN è possibile definire in uno stesso segmento trunk?

## A.2 Livello rete

### A.2.1 Prerequisiti

- Ricordo per sommi capi il funzionamento dei principali protocolli di livello rete, in particolare IP e ICMP?
- A cosa serve ARP?
- Cos'è una rete di classe A, B, C? So usare le maschere di rete e la notazione CIDR?
- Data una sottorete, come calcolo l'indirizzo di broadcast?
- Quali intervalli di indirizzi IP sono privati?
- Dato un indirizzo IP, o un intervallo di indirizzi, come calcolo la più piccola sottorete che lo contiene?
- In quali casi posso aggregare più sottoreti in un'unica riga di una tabella di routing?
- Che cosa succede se riutilizzo lo stesso indirizzo IP in punti diversi di una rete? In quali casi è consentito farlo senza generare confusione?

### A.2.2 Inter-VLAN routing

- In cosa consiste un'interfaccia "virtuale" di un router?
- Di che informazioni ha bisogno uno switch layer 3 per funzionare?

## A.3 Livello trasporto

### A.3.1 Prerequisiti

- Perché gli indirizzi di livello 4 si chiamano "porte"? A cosa sono normalmente associati?
- Come funzionano i numeri di sequenza TCP?
- Quando conviene usare UDP come protocollo di trasporto?

### A.3.2 NAT

- Per quali ragioni gli indirizzi IP privati sono sempre più diffusi?
- Quali informazioni aggiuntive sono richieste per operare in modalità Port Forwarding?
- Quali informazioni deve conservare un router per gestire una connessione TCP in modalità NAT?
- Su quali campi delle intestazioni TCP/IP agisce un router NAT?

## A.4 Livello applicativo

### A.4.1 Prerequisiti

- A cosa servono i protocolli DNS, RIP, DHCP, HTTP? Su quali protocolli di livell trasporto si basano? Quali servizi offrono?
- So interpretare una query SQL, un frammento di codice HTML?
- Cosa sono lo stack e lo heap di un'applicazione?

### A.4.2 Tipi di attacco

- Perché lo scambio dei file di zona è sempre più raro nel protocollo DNS?
- Che caratteristiche deve avere una macchina zombie da utilizzare in un idle port scan?
- Come si previene un attacco di tipo *buffer overflow*?
- Quali sono i fattori che rendono gli attacchi *denial of service* ancora attuali?

### A.4.3 Firewall

- Qual è la struttura di massima di una ACL?
- Perché usare un semplice packet filtering firewall, quando esistono soluzioni più complesse e versatili?
- Quali sono i vantaggi e gli svantaggi di un proxy applicativo?
- Quali soluzioni sono trasparenti rispetto ai PC utente, e quali non lo sono?

### A.4.4 Configurazioni di rete

- Perché è importante poter dividere la propria rete locale in zone?
- Che cos'è una DMZ?

## A.5 Crittografia

### A.5.1 Prerequisiti

- Quando una funzione è iniettiva? E quando è suriettiva?
- So ragionare in aritmetica modulare? Conosco i principali operatori logici?

### A.5.2 Definizioni

- Perché è importante che la funzione di cifratura sia iniettiva?
- Perché il cifrario di Cesare funzionava (ai tempi di Cesare)?
- Motivare (o contestare) la seguente affermazione: "un cifrario a blocchi è soltanto una sostituzione alfabetica con un alfabeto molto grande".

**Crittografia moltiplicativa (non modulare)** Supponiamo che Alice debba spedire a Bob una sequenza di $N$ interi positivi $m_1, \ldots, m_N$. Alice e Bob sono d'accordo su una chiave $k$, anch'essa numero positivo. Alice spedisce a Bob i messaggi moltiplicati per $k$: $c_i = km_i$, $i = 1, \ldots, N$. Bob può ricavare i valori originali dividendo per $k$ i codici ricevuti.
Charlie vede transitare solo i codici $c_i$ e non conosce né i messaggi originari, né la chiave. Come può cercare di dedurre qualcosa?

**Doppio one-time pad** Supponiamo che Alice debba trasmettere a Bob un messaggio $m$, ma che non possano stabilire una chiave condivisa. Allora ricorrono al seguente protocollo[1]:

- Alice genera una chiave $k_A$, lunga quanto $m$, e crea il codice $c_1 = m \oplus k_A$ (messaggio in or esclusivo con la chiave). Alice spedisce $c_1$ a Bob.
- Quando Bob riceve $c_1$, genera una propria chiave $k_B$, lunga quanto $c_1$, e crea il codice $c_2 = c_1 \oplus k_B$. Bob spedisce il codice $c_2$ ad Alice.
- Quando Alice riceve $c_2$, crea il codice $c_3 = c_2 \oplus k_A$ (ovviamente, usando la stessa chiave di prima). Alice spedisce $c_3$ a Bob.
- A questo punto, Bob calcola $c_3 \oplus k_B$, che è ovviamente uguale a $m$, come si dimostra facilmente notando che l'applicazione in xor di una stessa chiave un numero pari di volte ne annulla l'effetto:

  $$c_3 \oplus K_B = c_2 \oplus K_A \oplus K_B = c_1 \oplus K_B \oplus K_A \oplus K_B = m \oplus K_A \oplus K_B \oplus K_A \oplus K_B = m \oplus K_A \oplus K_A \oplus K_B \oplus K_B = m.$$

Ovviamente, Charlie non vede mai passare il messaggio $m$ in chiaro, quindi questo protocollo sembra garantire lo scambio segreto di messaggi in mancanza di una chiave condivisa.
Qual è il problema? Ovviamente, supponiamo che Charlie intercetti tutto lo scambio...

### A.5.3 Diffie-Hellman

- Che cosa succede esattamente se $p$ non è primo?
- E se $g$ non è una radice prima modulo $p$?

### A.5.4 Funzioni hash crittografiche

- Che cos'è una funzione hash?
- Che cosa si intende per collisione?
- Perché è importante la resistenza agli attacchi di collisione? E di preimmagine?
- In cosa consiste il meccanismo della *proof of work*?

## A.6 Domande a risposta multipla

In questa sezione raccogliamo alcune domande a risposta multipla. Ogni domanda ha una sola risposta "corretta", anche se in alcuni casi è possibile che qualche ambiguità causi incertezze.

1. Quale di queste soluzioni permette la separazione di una LAN in più domini di broadcast?
   (a) L'uso switch al posto dei più economici hub.
   (b) L'uso di LAN virtuali.
   (c) L'uso di NAT.
2. Quale di queste soluzioni permette la separazione di una LAN in più domini di collisione?
   (a) L'uso switch al posto dei più economici hub.

[1] si tratta di un tentativo di trasporre in chiave informatica il protocollo fisico che prevede che Alice inserisca il messaggio in una cassetta che chiude con un lucchetto di cui solo lei ha la chiave; la fa recapitare a Bob, il quale aggiunge alla chiusura un lucchetto di cui solo lui ha la chiave; la cassetta viene rispedita ad Alice, la quale toglie il proprio lucchetto e rispedisce la cassetta, ora chiusa dal solo lucchetto di Bob.

(b) L'uso di NAT.

(c) L'uso di LAN virtuali.

3. Quale tra i seguenti può essere considerato uno svantaggio nell'impiego di un proxy applicativo?

   (a) La necessità che il server che lo supporta sia dual-homed.

   (b) Il fatto che il client debba essere informato della sua esistenza, rendendo la soluzione poco trasparente.

   (c) Il caching dei dati, che fa venir meno la garanzia che il server sia stato veramente contattato.

4. Il termine *router-on-a-stick* fa riferimento a:

   (a) un router che utilizza una sola porta fisica per gestire più VLAN.

   (b) un router che utilizza più porte fisiche per instradare i pacchetti fra reti diverse.

   (c) uno switch con capacità di routing limitate all'instradamento di pacchetti fra VLAN.

5. Qual è l'indirizzo IP di broadcast della sottorete `162.39.102.80/29`?

   (a) `162.39.102.87`

   (b) `162.39.102.255`

   (c) `162.39.102.84`

6. I cavi ethernet che collegano due switch devono essere:

   (a) Dipende se le porte sono in modalità trunk o access

   (b) Dritti (straight-through)

   (c) Incrociati (cross-link)

7. Due switch sono collegati fra loro in modalità access. Che cosa succede se le due porte sono associate a due VLAN diverse?

   (a) I frame passano senza problemi.

   (b) Non passano frame, perché VLAN diverse non possono comunicare tra loro.

   (c) I frame vengono spediti dallo switch mittente, ma sono subito scartati dallo switch destinatario.

8. In cosa consiste il meccanismo del *port forwarding*?

   (a) Un router NAT cerca di riutilizzare per quanto possibile la porta effimera scelta dal client.

   (b) Il livello trasporto di un server utilizza l'informazione di porta per sapere a quale applicazione inoltrare l'informazione.

   (c) Una tabella NAT può avere alcune righe preimpostate per permettere l'accesso a servizi specifici da parte di client remoti.

9. Qual è il tipico scenario in cui si utilizza il *port scan*?

   (a) Un attaccante cerca di capire quali servizi siano attivi su una macchina remota per prenderne possesso.

   (b) All'accensione, un PC si informa sui servizi disponibili nella LAN cui appartiene.

   (c) Un attaccante cerca di capire quale sia la porta meno protetta di un router per guadagnare l'accesso a una LAN.

10. Se più righe di una ACL soddisfano i requisiti di un pacchetto TCP/IP, quale viene scelta dal firewall?

    (a) Quella con il prefisso di rete più corto.

    (b) La prima nell'ordine in cui sono elencate.

    (c) Quella con il prefisso di rete più lungo.

11. Quale dei seguenti dispositivi è un DTE?

    (a) Router

    (b) Hub

    (c) Switch

12. A quali livelli appartengono le informazioni su cui si basa il NAT per operare le traduzioni?

    (a) Data Link, Rete e Trasporto

    (b) Trasporto

    (c) Rete e Trasporto

13. Quanti indirizzi IP della sottorete `130.111.237.128/27` possono essere assegnati a interfacce fisiche?

    (a) 32

    (b) 30

    (c) 31

14. Se più righe di una routing table corrispondono a uno stesso indirizzo IP di destinazione, quale viene scelta dal router?

    (a) Quella con il prefisso di rete più corto.

    (b) La prima nell'ordine in cui sono elencate.

    (c) Quella con il prefisso di rete più lungo.

15. Lo standard 802.1Q regola...

    (a) ...l'inserimento del numero della VLAN di appartenenza nel payload di un frame di livello 2.

    (b) ...l'inserimento del numero della VLAN di appartenenza nell'intestazione di un frame di livello 2, sostituendo un campo inutilizzato.

    (c) ...l'inserimento del numero della VLAN di appartenenza nell'intestazione di un frame di livello 2, allungandola.

# Appendice B

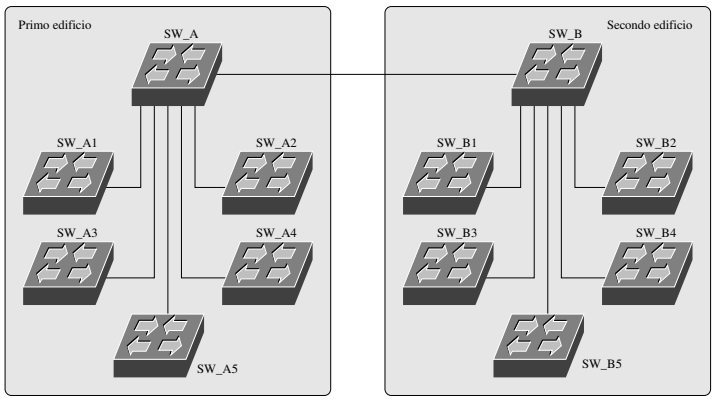
# Esercizi

**Esercizio 1**

Un amministratore di rete deve coprire due edifici con una rete locale cablata composta da tre LAN virtuali, che chiameremo 1, 2 e 3.
La rete 1 deve ospitare 100 PC distribuiti fra i due edifici; la rete 2 ospiterà 50 PC, tutti nel primo edificio; la rete 3 ospiterà 50 PC nel solo secondo edificio.
Tutti i PC devono avere indirizzo IP nella sottorete `192.168.100.0/24`.
**1.1)** Proporre un'architettura di rete che permetta di associare all'occorrenza ogni porta di rete a una VLAN arbitraria. Descrivere la configurazione degli switch.
**1.2)** Le tre VLAN debbono poter comunicare tra di loro (e con l'esterno) a livello IP. Si ha a disposizione un router con una porta Ethernet e una connessione WAN. Descrivere la configurazione del router (connessioni, interfacce reali e virtuali, tabella di instradamento) in grado di garantire tutte le connessioni.
**1.3)** Descrivere la configurazione (IP, maschera, gateway, indirizzo di broadcast) di un PC per ciascuna delle tre VLAN.

**Soluzione 1**

**1.1)** Vogliamo realizzare tre VLAN, chiamiamole `vlan1`, `vlan2` e `vlan3`. Suddividiamo la sottorete disponibile in tre parti, in modo che la prima possa indirizzare almeno 100 host; dato vhe $2^7 = 128 > 100$, la prima sottorete sarà `192.168.100.0/25`, e la parte rimanente potrà essere suddivisa ulteriormente in due parti, `192.168.100.128/26` e `192.168.100.192/26`.
L'esercizio chiede completa libertà nell'assegnazione delle porte Ethernet alle singole sottoreti, quindi possiamo limitarci alla seguente struttura:



Tutte le connessioni indicate saranno in modalità trunk, mentre tutte le altre porte degli switch andranno configurate in modalità access per assegnarle a una VLAN specifica. La regola secondo la quale `vlan2` e `vlan3` sono limitate a un edificio non è implementata a questo livello. Se si desidera imporre questa regola, si può configurare il collegamento fra SW_A e SW_B in modalità access.
**1.2)** Dobbiamo ovviamente usare il router in configurazione "on-a-stick"; per farlo dovremo impostare la sola interfaccia ethernet del router (chiamiamola `e0`) in modalità trunk e associarla a tre interfacce virtuali:

- l'interfaccia virtuale `e0.1`, associata a `vlan1` e avente indirizzo nella corrispondente sottorete, ad esempio `192.168.100.126/25`;
- l'interfaccia virtuale `e0.2`, associata a `vlan2` e avente indirizzo nella corrispondente sottorete, ad esempio `192.168.100.190/26`;
- l'interfaccia virtuale `e0.3`, associata a `vlan3` e avente indirizzo nella corrispondente sottorete, ad esempio `192.168.100.254/26`.

Collegheremo `e0` a un'interfaccia in modalità trunk di uno dei due swicth di livello superiore; ovviamente, il collegamento fra SW_A e SW_B dovrà lasciar passare tutt'e tre le VLAN, quindi dovrà essere posto in modalità trunk.

| Destinazione | Interfaccia | Metrica |
|---|---|---|
| `192.168.100.0/25` | `e0.1` | 1 |
| `192.168.100.128/26` | `e0.2` | 1 |
| `192.168.100.192/26` | `e0.3` | 1 |
| `0.0.0.0/0` | *configurazione lato ISP* | |

Si noti che, essendo tutte le reti note direttamente al router, non è necessario impostare esplicitamente la tabella, a meno dell'ultima riga.
**1.3)** Ecco le configurazioni dei tre PC, ovviamente ciascuno collegato a un'interfaccia di tipo access di uno switch opportunamente configurata:

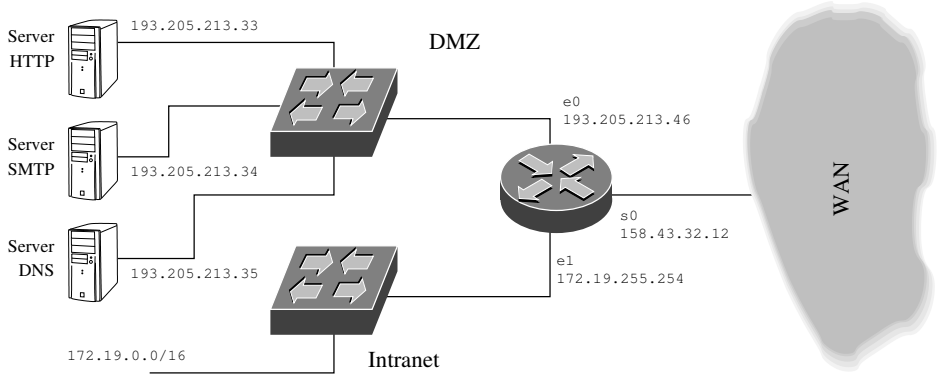
| Indirizzo IP | Maschera di rete | Gateway | Broadcast |
|---|---|---|---|
| `192.168.100.1` | `255.255.255.128` | `192.168.100.126` | `192.168.100.127` |
| `192.168.100.129` | `255.255.255.192` | `192.168.100.190` | `192.168.100.191` |
| `192.168.100.193` | `255.255.255.192` | `192.168.100.254` | `192.168.100.255` |

**Esercizio 2**

Una rete locale è costituita da una DMZ con indirizzi pubblici nella sottorete `193.205.213.32/28` e da un'intranet protetta con indirizzi privati nella rete `172.19.0.0/16`.
La DMZ ospita un server web (TCP, porta 80), un server SMTP (TCP, porta 25) e un server DNS (UDP, porta 53) su tre diverse macchine.
Si dispone di un router con due interfacce Ethernet e una interfaccia WAN con indirizzo `158.43.32.12`, netmask `/29`; il default gateway dell'ISP ha l'IP più baso in quella rete.
**2.1)** Descrivere la configurazione delle interfacce del router e la sua tabella di instradamento.
**2.2)** Descrivere la configurazione NAT (interfaccia interna ed esterna) e le ACL necessarie a garantire l'accesso esterno alla DMZ per i soli servizi elencati, e la connettività dall'intranet protetta verso l'esterno.

**Soluzione 2**

**2.1)** Supponiamo che il router disponga di almeno due interfacce ethernet, `e0` ed `e1`, dedicate rispettivamente alla DMZ e all'intranet privata, e la seriale `s0` rivolta verso la WAN. Ecco una possibile configurazione:




La rete dell'ISP ala quale il router è connesso tramite `s0` è la `158.43.32.8/29`, e l'IP del gateway, essendo il più basso di questa sottorete, è `158.43.32.9`. La tabella di instradamento corrispondente è

| Destinazione | Interfaccia | Gateway |
|---|---|---|
| `193.205.213.32/28` | `e0` | — |
| `172.19.0.0/16` | `e1` | — |
| `158.43.32.8/29` | `s0` | — |
| `0.0.0.0/0` | `s0` | `158.43.32.9` |

**2.2)** La configurazione NAT riguarderà soltanto l'intranet protetta, in quanto la DMZ dispone di indirizzi pubblici. L'interfaccia interna sarà dunque la `e1`, mentre quella esterna è ovviamente `s0`. Il pool di indirizzi disponibili per il NAT è ovviamente l'unico assegnato all'interfaccia esterna, `158.43.32.12`.
La configurazione delle ACL è meno deterministica. In generale, si vorranno bloccare tutti i pacchetti verso l'intranet a meno che non appartengano a comunicazioni iniziate dall'interno, e tutti gli accessi verso servizi della DMZ non appartenenti a quelli elencati. Ad esempio, ecco una possibile ACL applicata in ingresso all'interfaccia `s0`:

| IP Destinazione | Porta destinazione | Flags | Azione | Spiegazione |
|---|---|---|---|---|
| `193.205.213.33/32` | 80 | * | allow | Accesso a DMZ |
| `193.205.213.34/32` | 25 | * | allow | Accesso a DMZ |
| `193.205.213.35/32` | 53 | * | allow | Accesso a DMZ |
| `172.19.0.0/16` | * | ESTABLISHED | allow | Risposte a Intranet |
| `0.0.0.0/0` | * | * | deny | Default |

Essendo applicata in ingresso a `s0`, questa ACL lascia piena libertà di comunicazione fra la intranet e la DMZ: in un contesto reale, ovviamente, bisognerebbe limitare i posibili danni che un'eventuale intrusione nell'intranet potrebbe causare alla DMZ e viceversa. Infine, nel caso in cui non si ritenga l'intranet sufficientemente protetta dal NAT (che ovviamente impedisce per sua natura connessioni dall'esterno) si potrà inserire un'opportuna regola.

**Esercizio 3**

Un router con funzionalità NAT dispone di un unico indirizzo IP pubblico `125.41.33.11`, assegnato alla sua interfaccia `e1`, e gestisce una rete locale con IP privati `172.30.16.0/20` tramite la sua interfaccia `e0` con IP `172.31.16.1`.
La tabella NAT del router contiene le seguenti righe statiche:

| Protocollo | Porta router | Porta client | IP client |
|---|---|---|---|
| TCP | 80 | 80 | 172.30.18.44 |
| TCP | 81 | 80 | 172.30.21.137 |

I seguenti pacchetti IP (delle cui intestazioni riportiamo solamente alcuni campi) vengono consegnati al router:

| | |
|---|---|
| Interfaccia: | e0 |
| Protocollo: | TCP (SYN) |
| Mittente: | 172.30.20.245:36533 |
| Destinatario: | 115.32.18.7:22 |

| | |
|---|---|
| Interfaccia: | e1 |
| Protocollo: | TCP (SYN) |
| Mittente: | 41.112.29.238:57632 |
| Destinatario: | 125.41.33.11:80 |

**3.1)** Come verranno modificate dal router le intestazioni dei pacchetti? Quale nuova riga verrà aggiunta alla tabella NAT?
**3.2)** Quai sarano le intestazioni dei corrispondenti pacchetti `SYN/ACK` di ritorno dai rispettivi server? Come verranno modificate dal router?

**Soluzione 3**

**3.1)** Il primo pacchetto proviene da un host interno ed è diretto verso un IP pubblico esterno. Il router dovrà scegliere una propria porta effimera, ad esempio `55555`, e inserirla insieme al proprio indirizzo IP pubblico nel campo mittente. Il pacchetto sarà dunque:

| | |
|---|---|
| Interfaccia: | e1 (in uscita) |
| Protocollo: | TCP (SYN) |
| Mittente: | 125.41.33.11:55555 |
| Destinatario: | 115.32.18.7:22 |

La traduzione verrà registrata nella tabella NAT:

| Protocollo | Porta router | Porta client | IP client |
|---|---|---|---|
| TCP | 55555 | 36533 | 172.30.20.245 |

Il secondo pacchetto proviene da un host pubblico ed è diretto a una delle porte del router contemplate dalle regole statiche (la prima). Il router rimuoverà il proprio IP e la porta dal campo destinazione e li rimpiazzerà con i dati del client riportati nella tabella indicizzata con la porta di destinazione (`80`):

| | |
|---|---|
| Interfaccia: | e0 (in uscita) |
| Protocollo: | TCP (SYN) |
| Mittente: | 41.112.29.238:57632 |
| Destinatario: | 172.30.18.44:80 |

La regola è già presente, quindi il passaggio del pacchetto non comporta aggiunte alla tabella di traduzione.
**3.2)** Quando il primo pacchetto `SYN` viene ricevuto dall'host esterno, il corrispondente `SYN/ACK` viene rispedito invertendo i campi sorgente e destinazione:

| | |
|---|---|
| Interfaccia: | e1 |
| Protocollo: | TCP (SYN/ACK) |
| Mittente: | 115.32.18.7:22 |
| Destinatario: | 125.41.33.11:55555 |

La porta di destinazione `55555` viene usata dal router per ottenere l'host finale:

| | |
|---|---|
| Interfaccia: | `e0` (in uscita) |
| Protocollo: | TCP (`SYN/ACK`) |
| Mittente: | `115.32.18.7:22` |
| Destinatario: | `172.30.20.245:36533` |

Ugualmente, l'host interno destinatario del secondo pacchetto `SYN` risponderà col seguente pacchetto:

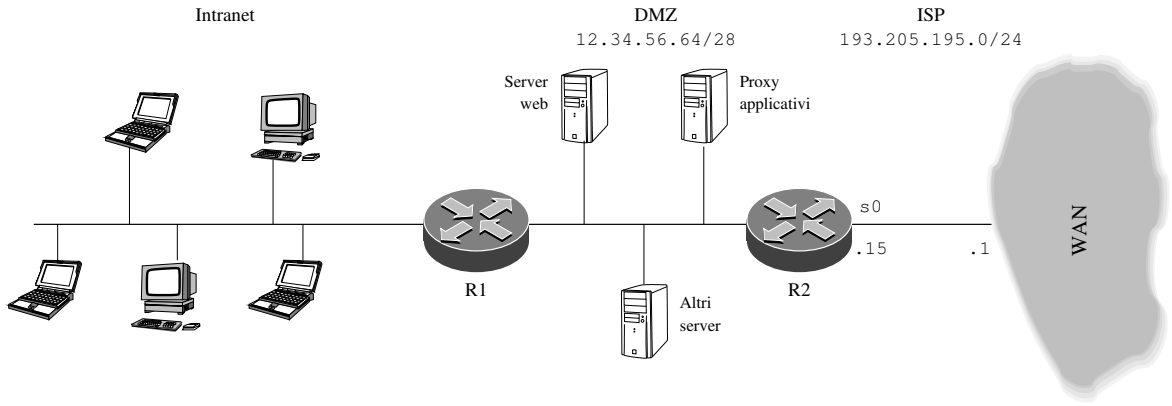
| | |
|---|---|
| Interfaccia: | `e0` |
| Protocollo: | TCP (`SYN/ACK`) |
| Mittente: | `172.30.18.44:80` |
| Destinatario: | `41.112.29.238:57632` |

Il router vedrà che la coppia IP, porta del mittente sono già registrate nella tabella (prima riga statica), quindi effettuerà la traduzione senza il bisogno di aggiornare la tabella stessa:

| | |
|---|---|
| Interfaccia: | `e1` (in uscita) |
| Protocollo: | TCP (`SYN/ACK`) |
| Mittente: | `125.41.33.11:80` |
| Destinatario: | `41.112.29.238:57632` |

**Esercizio 4**

Si consideri la seguente rete aziendale, divisa in tre rami (gli switch non sono rappresentati):




- Il router esterno, R2, si affaccia con l'interfaccia `s0` verso la rete dell'ISP `193.205.195.0/24`. All'interfaccia, l'ISP assegna l'indirizzo IP `193.205.195.15`, e il gateway della rete è `193.205.195.1`.
- Una DMZ dispone di indirizzi pubblici nella sottorete `12.34.56.64/28`. Contiene alcuni server che devono poter comunicare con l'esterno, in particolare:
  - un server web, che deve poter accettare connessioni dall'esterno;
  - un proxy applicativo che deve consentire un minimo livello di connettività alle macchine dell'intranet;
  - altri server (ad esempio database, SMTP, DNS...).
- La rete interna, "Intranet", è gestita come un'unica LAN. Deve essere mappata su una grande sottorete privata (ad esempio una `/16`). I suoi dispositivi devono poter comunicare solamente con i server della DMZ.

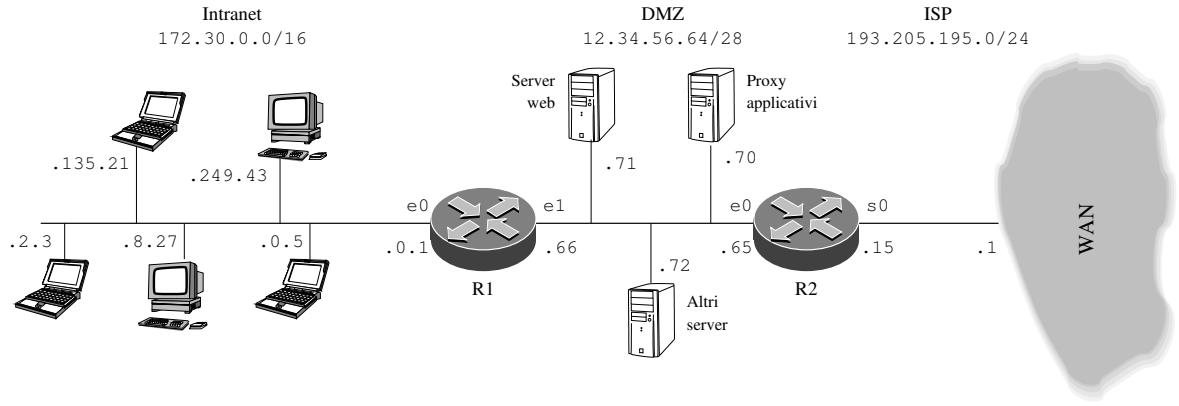
**4.1)** Si scelgano liberamente le sottoreti e gli indirizzi mancanti dallo schema. Indicare le configurazioni di base dei router R1 ed R2 per consentire le comunicazioni fra l'Intranet e la DMZ, e fra la DMZ e l'esterno. Si noti che l'intranet non ha motivo di comunicare con l'esterno.
**4.2)** Riportare la configurazione di rete di una delle macchine dell'intranet e di uno dei server della DMZ.
**4.3)** Impostare le ACL in R2 in modo che le uniche comunicazioni consentite siano:

- le connessioni da client esterni verso il server web della DMZ;
- le connessioni del proxy applicativo verso server web esterni.

Ogni altra comunicazione dev'essere bloccata.

**Soluzione 4**

Prima di iniziare, completiamo il disegno assegnando indirizzi IP ai vari dispositivi, e nomi alle interfacce fisiche dei router. Assegniamo la sottorete `172.30.0.0/16` all'intranet, e supponiamo che R1 vi sia collegato con l'interfaccia `e0` alla quale daremo il primo IP disponibile. Inoltre, assegniamo indirizzi IP ai dispositivi della DMZ:




**4.1)** Sulla base dei valori appena decisi, le interfacce del router R1 hanno la seguente configurazione:

| Interfaccia | IP | Netmask |
|---|---|---|
| e0 | 172.30.0.1 | 255.255.0.0 |
| e1 | 12.34.56.66 | 255.255.255.240 |

Il router R2 ha la seguente configurazione:

| Interfaccia | IP | Netmask |
|---|---|---|
| e0 | 12.34.56.65 | 255.255.255.240 |
| s0 | 193.205.195.15 | 255.255.255.0 |

Ecco la tabella di routing di R1:

| Destinazione | Interfaccia | Gateway | |
|---|---|---|---|
| 172.30.0.0/16 | e0 | — | Automatica |
| 12.34.56.64/28 | e1 | — | Automatica |
| 0.0.0.0/0 | e1 | 12.34.56.65 | Per il resto c'è R2 |

Si osservi che, stando al testo dell'esercizio, il router R1 deve gestire esclusivamente le comunicazioni fra la DMZ e l'Intranet. La riga di default, quindi, potrebbe anche essere omessa. La tabella di R2 è la seguente:

| Destinazione | Interfaccia | Gateway | |
|---|---|---|---|
| 12.34.56.64/28 | e0 | — | Automatica |
| 193.205.195.0/24 | s0 | — | Automatica |
| 172.30.0.0/16 | e0 | 12.34.56.66 | (*) Intranet attraverso R1 |
| 0.0.0.0/0 | s0 | 193.205.195.1 | Default attraverso il gateway dell'ISP |

In questo caso, possiamo osservare che R2 non dovrebbe avere motivo di raggiungere l'intranet, quindi la terza riga della tabella potrebbe essere omessa (ma si veda più avanti).

**4.2)** Ecco la configurazione di rete di una macchina dell'intranet:

| | |
|---|---|
| Indirizzo IP | 172.30.195.220 |
| Netmask | 255.255.0.0 |
| Default gateway | 172.30.0.1 |

Ecco la configurazione di rete di una macchina della DMZ:

| | |
|---|---|
| Indirizzo IP | 12.34.56.67 |
| Netmask | 255.255.255.240 |
| Default gateway | 12.34.56.65 |

Si osservi che questa configurazione di rete permette alle macchine della DMZ di inviare pacchetti verso la intranet solo attraverso il loro default gateway R1, a patto che la riga marcata con (*) sia presente nella routing table. Un pacchetto dalla DMZ verso l'intranet deve dunque passare per entrambi i router. In alternativa, le macchine della DMZ devono essere informate dell'esistenza di entrambi i router impostando le loro tabelle di routing (supponendo che il nome dell'interfaccia ethernet del

server sia `en0`):

| Destinazione | Interfaccia | Gateway | |
|---|---|---|---|
| `12.34.56.64/24` | `en0` | — | Per raggiungere il resto della DMZ |
| `172.30.0.0/16` | `en0` | `12.34.56.66` | Intranet attraverso R1 |
| `0.0.0.0/0` | `en0` | `12.34.56.65` | Default attraverso R2 |

**4.3)** Ecco una possibile ACL da installare in R2:

| Protocollo | Sorgente | Destinazione | Flag | Azione | |
|---|---|---|---|---|---|
| `TCP` | `0.0.0.0/0:*` | `12.34.56.71/32:80` | `*` | PASS | Mondo → web server |
| `TCP` | `12.34.56.71/32:80` | `0.0.0.0/0:*` | `ESTABLISHED` | PASS | Web server → mondo |
| `TCP` | `12.34.56.70/32:*` | `0.0.0.0/0:80` | `*` | PASS | Proxy → mondo |
| `TCP` | `0.0.0.0/0:80` | `12.34.56.70/32:*` | `ESTABLISHED` | PASS | Mondo → proxy |
| `*` | `*` | `*` | `*` | `DROP` | Proibire tutto il resto |

Le prime due righe consentono a ogni client esterno di collegarsi alla porta 80 del server web `12.34.56.71`, e a questo di rispondere (ovviamente solo se la connessione è iniziata dall'esterno). Le due righe successive consentono al proxy della DMZ `12.34.56.70` di collegarsi alla porta 80 di ogni server della rete, e a questo di rispondere. Ogni altro pacchetto è scartato (riga di default).
Naturalmente, le ACL di un caso reale prevederanno molti più casi.

**Esercizio 5**

Un'azienda desidera strutturare la propria rete interna in tre parti: una intranet per le vendite, una intranet per l'amministrazione e una DMZ per i server pubblici.
Le due intranet devono condividere la stessa infrastruttura di livello 2, e devono utilizzare sottoreti IP private, ma operano in modo diverso. L'intranet delle vendite deve poter effettuare connessioni arbitrarie verso l'esterno, mentre quella dell'amministrazione ha solamente la possibilità di accedere all'esterno tramite un proxy situato nella DMZ.
La DMZ dispone di indirizzi IP pubblici nella sottorete `185.44.23.192/28` e deve esporre su macchine diverse, oltre al già citato proxy (porta TCP 3128), anche un server web (porta TCP 80) e un server DNS (porte UDP e TCP 53). Per le due intranet si utilizzano gli IP privati nella sottorete `192.168.42.0/23`, da suddividere in parti uguali.
Il router dispone di un'interfaccia `s0` rivolta verso l'ISP; l'interfaccia ha indirizzo `135.21.44.23/16` e utilizza come default gateway l'ultimo IP della stessa sottorete. Dispone inoltre di due interfacce Ethernet, `e0` ed `e1`, da utilizzare rispettivamente per la DMZ e per le intranet.
**5.1)** Disegnare la struttura della rete (ovviamente rappresentando pochi host per ciascuna intranet).
**5.2)** Descrivere la configurazione di base del router senza considerare, per ora, la sicurezza: configurazione delle interfacce fisiche e virtuali, tabella di routing, servizio NAT.
**5.3)** Applicare delle access control list in modo che la DMZ esponga solo i servizi previsti. Ricordare che il proxy, pur essendo nella DMZ, deve essere accessibile solo dall'intranet che lo utilizza (ma deve poter comunicare liberamente con l'esterno); le due intranet non devono poter comunicare tra loro nemmeno a livello 3.

**Soluzione 5**

**5.1)** Si tratta di una soluzione three-legged. Supponiamo che `e0` gestisca la DMZ e `e1` le due intranet. Per le due intranet abbiamo bisogno di spezzare la sottorete privata in due parti, ad esempio:

- `192.168.42.0/24` per le vendite;
- `192.168.43.0/24` per l'amministrazione.

Le due sotoreti saranno affidate a due VLAN separate gestite dall'interfaccia `e1` del router in modalità trunk e spezzata in due interfacce virtuali, `e1.10` (VLAN 10, vendite) ed `e1.20` (VLAN 20, amministrazione). Gli switch dell'intranet saranno collegati tra loro e con il router in modalità trunk, mentre esporranno interfacce access verso gli host.
**5.2)** Scegliamo di usare gli indirizzi più alti per il gateway delle diverse sottoreti. Allora le interfacce del router avranno la seguente configurazione:

| Interfaccia fisica | Interfaccia virtuale | IP | |
|---|---|---|---|
| `s0` | — | `135.21.44.23/16` | Indirizzo assegnato dall'ISP |
| `e0` | — | `185.44.23.206/28` | Indirizzo più alto della sottorete DMZ |
| `s1` | `e1.10` | `192.168.42.254/24` | Indirizzo più alto VLAN 10 (vendite) |
| | `e1.20` | `192.168.43.254/24` | Indirizzo più alto VLAN 20 (amministrazione) |

Il default gateway dell'ISP richiesto dall'esercizio ha indirizzo IP `135.21.255.254` (l'IP più alto della sottorete `135.21.0.0/16`). La tabella di routing è dunque la seguente:

| Destinazione | Interfaccia | Gateway | |
|---|---|---|---|
| `185.44.23.192/28` | `e0` | — | Connessione diretta alla DMZ |
| `192.168.42.0/24` | `e1.10` | — | Connessione diretta alla VLAN 10 |
| `192.168.43.0/24` | `e1.20` | — | Connessione diretta alla VLAN 20 |
| `135.21.0.0/16` | `s0` | — | Connessione diretta alla rete ISP |
| `0.0.0.0/0` | — | `135.21.255.254` | Default |

Infine, per quanto riguarda la configurazione del NAT, questo è richiesto solamente fra la VLAN 10 delle vendite e l'interfaccia esterna, quindi avrà come interfaccia interna `e1.10`, come interfaccia esterna `s0`, e come pool di indirizzi pubblici l'unico indirizzo assegnato dall'ISP, `135.21.44.23`.
**5.3)** Sono possibili varie soluzioni. Ad esempio, le seguenti regole poste in uscita dalla porta `e0` dovrebbero bloccare tutto il traffico non desiderato verso la DMZ:

| Prot. | Dest. | Porta | Flag | Esito | |
|---|---|---|---|---|---|
| TCP | `185.44.23.193/32` | `80` | * | PASS | Connessioni al server web |
| * | `185.44.23.194/32` | `53` | * | PASS | Connessioni al server DNS |
| TCP | `185.44.23.195/32` | `3128` | * | PASS | Connessioni al proxy |
| TCP | `185.44.23.192/28` | * | ESTABLISHED | PASS | Connessioni aperte dall'interno |
| * | * | * | * | DROP | Tutto il resto non passa |

In più, alcune regole in ingresso alla porta `s0` bloccano l'accesso al proxy:

| Prot. | Dest. | Porta | Flag | Esito | |
|---|---|---|---|---|---|
| TCP | `185.44.23.195/32` | `3128` | * | DROP | Blocca le richieste al proxy |
| * | `185.44.23.192/28` | * | * | PASS | Connessioni alla DMZ (filtrate in `e0`) |
| TCP | `135.21.44.23/32` | * | ESTABLISHED | PASS | Connessioni aperte da NAT |
| * | * | * | * | DROP | Tutto il resto non passa |

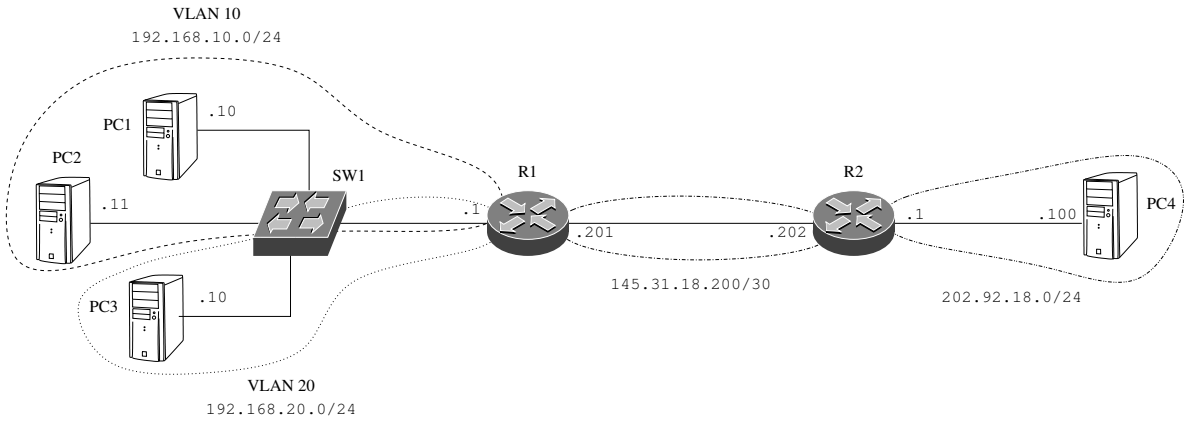
Per quanto riguarda i rapporti fra le intranet e la DMZ, possiamo imporre la seguente ACL per i pacchetti in entrata alla porta `e1.10`:

| Prot. | Dest. | Porta | Flag | Esito | |
|---|---|---|---|---|---|
| TCP | `185.44.23.195/32` | `3128` | * | DROP | Blocca le richieste al proxy |
| * | `185.44.23.192/28` | * | * | PASS | Connessioni alla DMZ (filtrate in `e0`) |
| * | `192.168.43.0/24` | * | * | DROP | Non si vede l'altra intranet |
| * | * | * | * | PASS | Tutto il resto passa (NAT) |

Ovviamente, molte soluzioni sono possibili.

**Esercizio 6**

La figura descrive tre LAN fisiche connesse fra loro da due router. Una delle LAN è ulteriormente suddivisa in due VLAN.




Tutte le interfacce sono di tipo Ethernet. I collegamenti tra PC1, PC2, PC3 e SW1 sono di tipo access sulle rispettive VLAN; il collegamento tra SW1 e R1 è di tipo trunk. Le due VLAN utilizzano indirizzi IP privati che non possono transitare al di fuori delle VLAN stesse; le altre LAN utilizzano IP pubblici.
**6.1)** Descrivere le tabelle di instradamento dei due router. I nomi delle interfacce fisiche dei router sono `e1` verso sinistra ed `e2` verso destra; ipotizzare dei nomi significativi per le eventuali interfacce logiche (ad esempio, `e2.30` per un'interfaccia virtuale basata su `e2` che serve la VLAN 30).
**6.2)** Descrivere il tragitto e la composizione a livello 2 (data link), 3 (rete) e 4 (trasporto) per i seguenti pacchetti:

1. un pacchetto UDP da PC1 (porta effimera) a PC3 (porta 57);
2. un pacchetto TCP con flag SYN da PC2 (porta effimera) a PC4 (porta 80);
3. il corrispondente pacchetto di risposta da PC4 a PC2.

**Soluzione 6**

**6.1)** L'interfaccia `e1` di R1 serve due VLAN, quindi va divisa in due interfacce virtuali. La tabella di routing di R1 è la seguente:

| Rete | Interfaccia | Gateway | |
|---|---|---|---|
| `192.168.10.0/24` | `e1.10` | — | VLAN 10 |
| `192.168.20.0/24` | `e1.20` | — | VLAN 20 |
| `145.31.18.200/30` | `e2` | — | Connessione con R2 |
| `202.92.18.0/24` | — | `145.31.18.202` | R2 è il gateway per arrivare alla rete di PC4 |

L'ultima riga è importante, altrimenti R1 non può sapere come far arrivare i pacchetti a PC4, che non si trova in nessuna delle reti da esso controllate. La tabella di R2 è più semplice:

| Rete | Interfaccia | Gateway | |
|---|---|---|---|
| `202.92.18.0/24` | `e2` | — | LAN più a destra |
| `145.31.18.200/30` | `e1` | — | Connessione con R1 |

In questo caso non è necesario citare le due VLAN, perché R2 non deve sapere nulla a loro riguardo: i loro IP sono privati, e dovranno essere NATtati da R1 per poter circolare.
**6.2)** Il pacchetto UDP da PC1 a PC3, pur restando confinato nella stessa LAN, si muove su due sottoreti diverse. Quindi PC1, quando lo trasmette, deve farlo arrivare al default gateway (che per lui è il router R1). Il pacchetto uscente da PC1 e diretto a SW1 è dunque:

| | | |
|---|---|---|
| Livello data link | Destinatario | Indirizzo MAC della porta `e1.10` di R1 |
| | Mittente | Indirizzo MAC di PC1 |
| Livelli 3 e 4 | IP e porta di destinazione | `192.168.20.10:57` |
| | IP e porta mittente | `192.168.10.10:12345` |

Quando SW1 riceve il frame, che appartiene alla VLAN 10 in quanto ricevuto attraverso una porta access, lo inoltra verso il destinatario di livello 2 (R1); dovendo però iniettare il frame in un collegamento di tipo trunk, deve arricchirlo con l'opportuno tag 802.1Q. Il frame diviene dunque:

| | | |
|---|---|---|
| Livello data link | Destinatario | Indirizzo MAC della porta `e1.10` di R1 |
| | Mittente | Indirizzo MAC di PC1 |
| | Tag 802.1Q | 10 |
| Livelli 3 e 4 | IP e porta di destinazione | `192.168.20.10:57` |
| | IP e porta mittente | `192.168.10.10:12345` |

Il router R1 riceve il frame attraverso l'interfaccia virtuale `e1.10`; consulta la tabella di routing e destina il pacchetto verso la VLAN 20; il frame che esce da R1 verso SW1 è dunque:

| | | |
|---|---|---|
| Livello data link | Destinatario | Indirizzo MAC di PC3 |
| | Mittente | Indirizzo MAC della porta `e1.20` di R1 |
| | Tag 802.1Q | 20 |
| Livelli 3 e 4 | IP e porta di destinazione | `192.168.20.10:57` |
| | IP e porta mittente | `192.168.10.10:12345` |

Infine, SW1 riceve il frame e lo inoltra attraverso l'opportuno collegamento access, togliendo dunque il tag 802.1Q:

| | | |
|---|---|---|
| Livello data link | Destinatario | Indirizzo MAC di PC3 |
| | Mittente | Indirizzo MAC della porta `e1.20` di R1 |
| Livelli 3 e 4 | IP e porta di destinazione | `192.168.20.10:57` |
| | IP e porta mittente | `192.168.10.10:12345` |

Si noti come, in tutto il passaggio, la parte di livello 3 non sia mai stata modificata.
Per quanto riguarda il pacchetto TCP, ecco la sequenza. Da PC2 a SW1:

| | | |
|---|---|---|
| Livello data link | Destinatario | Indirizzo MAC della porta `e1.10` di R1 |
| | Mittente | Indirizzo MAC di PC2 |
| Livelli 3 e 4 | IP e porta di destinazione | `202.92.18.100:80` |
| | IP e porta mittente | `192.168.10.11:54321` |

Da SW1 a R1:

| | | |
|---|---|---|
| Livello data link | Destinatario | Indirizzo MAC della porta `e1.10` di R1 |
| | Mittente | Indirizzo MAC di PC2 |
| | Tag 802.1Q | 10 |
| Livelli 3 e 4 | IP e porta di destinazione | `202.92.18.100:80` |
| | IP e porta mittente | `192.168.10.11:54321` |

R1 estrae il pacchetto IP, consulta la tabella di routing e decide di inoltrare il pacchetto verso l'interfaccia `e1` di R2. Inoltre, visto che tra le VLAN e l'esterno è attivo il servizio NAT, sostituisce l'indirizzo mittente col proprio e utilizza una porta mittente di propria iniziativa:

| | | |
|---|---|---|
| Livello data link | Destinatario | Indirizzo MAC della porta `e1` di R2 |
| | Mittente | Indirizzo MAC della porta `e2` di R1 |
| Livelli 3 e 4 | IP e porta di destinazione | `202.92.18.100:80` |
| | IP e porta mittente | `145.31.18.201:55555` |

Infine, R2 estrae il pacchetto IP, consulta la propria tabella di routing e inoltra il pacchetto chiudendolo in un frame diretto verso il destinatario finale:

| | | |
|---|---|---|
| Livello data link | Destinatario | Indirizzo MAC di PC4 |
| | Mittente | Indirizzo MAC della porta `e2` di R2 |
| Livelli 3 e 4 | IP e porta di destinazione | `202.92.18.100:80` |
| | IP e porta mittente | `145.31.18.201:55555` |

Il pacchetto SYN/ACK di ritorno seguirà il percorso inverso, con l'inversione degli indirizzi mittente e destinatario in tutti i passaggi.

**Nota bene** — la soluzione presentata riporta tutti i dettagli di qualche utilità; molte soluzioni parziali sono comunque state valutate al massimo dei punti.

**Esercizio 7**

Il reparto IT di un'azienda deve gestire le seguenti sottoreti:

- una DMZ da 5 host, dotati di indirizzi pubblici;
- una server farm di 50 host, dotati di indirizzi pubblici;
- due intranet da 100 host l'una, basate su IP privati.

Le regole di comunicazione sono le seguenti:

- le quattro reti devono poter comunicare tra loro a livello 3 senza restrizioni;
- gli host della DMZ devono essere accessibili senza restrizioni anche da Internet;
- gli host della server farm devono essere accessibili anche da Internet, ma alle sole porte `22` e `80`;
- le due intranet devono poter accedere a Internet tramite NAT in uscita.

Per realizzare il sistema, il reparto IT ha a disposizione:

- la sottorete IP pubblica `195.41.64.128/25` da ripartire fra la DMZ e la server farm, con l'indicazione di minimizzare le dimensioni delle sottoreti utilizzate, riservando quanti più indirizzi possibili per usi futuri;
- un router dotato di due porte Ethernet e una porta WAN (per comunicare con l'ISP), con funzionalità di NAT, ACL e interfacce virtuali;
- un numero sufficiente di switch a 24 porte con funzionalità VLAN;
- per la connessione a Internet, l'ISP ha messo a disposizione l'indirizzo IP `123.45.67.89` con gateway `123.45.67.90` e netmask `/30`;
- per le intranet è possibile utilizzare qualunque sottorete IP privata.

**7.1)** Descrivere l'allocazione delle sottoreti IP alle varie reti richieste.
**7.2)** Disegnare una possibile configurazione fisica della rete aziendale (con tutti gli switch e pochi host d'esempio) e descrivere le configurazioni degli switch: VLAN, porte access e trunk.
**7.3)** Descrivere la configurazione del router: configurazione delle porte (fisiche e virtuali), tabelle di routing, ACL e NAT.

**Soluzione 7**

**7.1)** Per la DMZ (5 host) servono 3 bit di host, quindi basta una rete `/29`; per la server farm servono 6 bit, quindi una rete `/26`. Una possibilità è dunque spezzare ricorsivamente la sottorete pubblica a nostra disposizione:

- Dalla sottorete `195.41.64.128/25` otteniamo le due sottoreti `195.41.64.128/26` e `195.41.64.192/26`. Usiamo la prima per la server farm.
- Spezziamo la seconda in due parti: `195.41.64.192/27` e `195.41.64.224/27`. Mettiamo da parte la seconda per usi futuri.
- Spezziamo la prima in due parti: `195.41.64.192/28` e `195.41.64.208/28`. Mettiamo da parte la seconda per usi futuri.
- Spezziamo la prima in due parti: `195.41.64.192/29` e `195.41.64.200/29`. Usiamo la prima per la DMZ e mettiamo da parte la seconda per usi futuri.

Infine, per le intranet dovremo usare due reti `/25`, facendo attenzione a scegliere degli IP privati; ad esempio, `192.168.1.0/25` e `192.168.1.128/25`. Riassumendo, ecco le reti utilizzate:

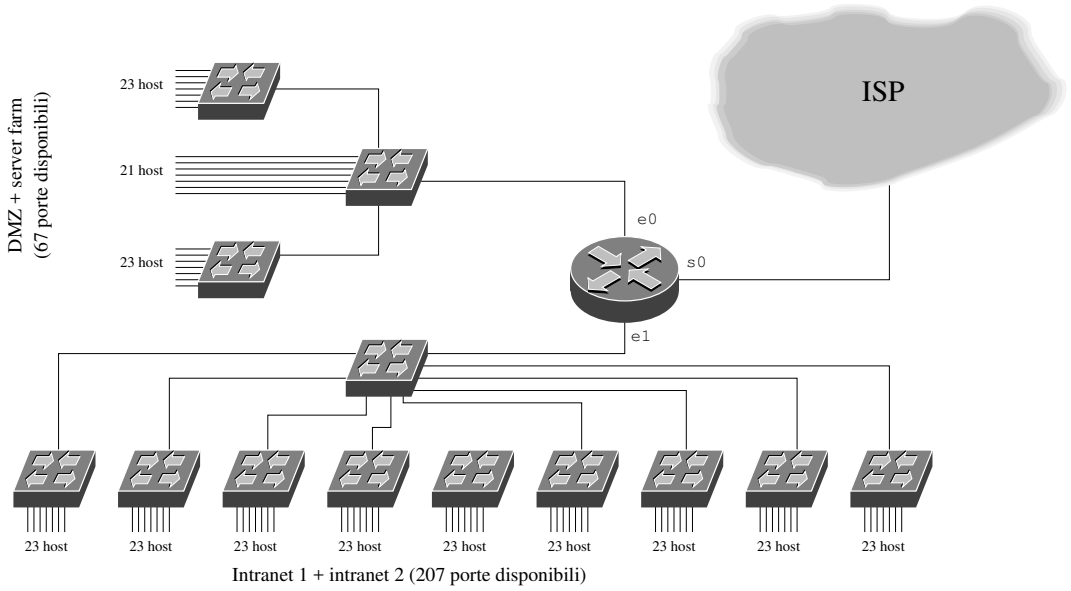
| Rete | IP | Gateway | Netmask | Broadcast |
|---|---|---|---|---|
| Server farm | `195.41.64.128/25` | `195.41.64.129` | `255.255.255.128` | `195.41.64.191` |
| DMZ | `195.41.64.192/29` | `195.41.64.193` | `255.255.255.248` | `195.41.64.199` |
| Intranet 1 | `192.168.1.0/25` | `192.168.1.1` | `255.255.255.128` | `192.168.1.127` |
| Intranet 2 | `192.168.1.128/25` | `192.168.1.129` | `255.255.255.128` | `192.168.1.255` |

Le sottoreti pubbliche avanzate per usi futuri sono dunque:

| IP | Netmask | Broadcast |
|---|---|---|
| `195.41.64.200/29` | `255.255.255.248` | `195.41.64.207` |
| `195.41.64.208/28` | `255.255.255.240` | `195.41.64.223` |
| `195.41.64.224/27` | `255.255.255.224` | `195.41.64.255` |

Ovviamente, molti di questi dettagli non sono richiesti dall'esercizio e sono presentati solo per completezza.

**7.2)** Dato che il router possiede solo due interfacce Ethernet, mentre le reti locali da gestire sono quattro, supporremo di far uso di reti virtuali (VLAN). In particolare, possiamo decidere che la DMZ e la server farm (55 host in tutto) condividano la stessa LAN fisica, e così le due intranet (200 host in tutto); altre suddivisioni sono ovviamente possibili, ad esempio a partire da considerazioni sul bilanciamento dei carichi. Dato che gli switch sono a 24 porte, una possibile configurazione è quella riportata in figura:




Le porte che collegano gli switch tra loro e col router andranno ovviamente impostate in modalità trunk, mentre le porte rivolte verso gli host saranno in modalità access sulla VLAN assegnata all'host in questione. Definiremo le VLAN nel modo seguente:

| Mnemonico | ID |
|---|---|
| `DMZ` | 10 |
| `server_farm` | 20 |
| `intranet1` | 30 |
| `intranet2` | 40 |

**7.3)** Per accomodare le VLAN, le due porte Ethernet andranno sdoppiate in due porte logiche ciascuna, con l'incapsulamento 802.1Q opportuno. Ovviamente, la porta `s0` va dedicata al collegamento WAN con l'ISP, quindi si usa direttamente l'interfaccia fisica:

| Porta fisica | Porta logica | ID 802.1Q | IP | netmask |
|---|---|---|---|---|
| `e0` | `e0.0` | 10 | `195.41.64.193` | `255.255.255.248` |
| | `e0.1` | 20 | `195.41.64.129` | `255.255.255.128` |
| `e1` | `e1.0` | 30 | `192.168.1.1` | `255.255.255.128` |
| | `e1.1` | 40 | `192.168.1.129` | `255.255.255.128` |
| `s0` | — | — | `123.45.67.89` | `255.255.255.252` |

La tabella di instradamento elenca le sottoreti collegate e il percorso di default:

| Destinazione | Prossimo passo | Interfaccia | |
|---|---|---|---|
| `195.41.64.192/29` | — | `e0.0` | DMZ |
| `195.41.64.128/26` | — | `e0.1` | Server farm |
| `192.168.1.0/25` | — | `e1.0` | Intranet 1 |
| `192.168.1.128/25` | — | `e1.1` | Intranet 2 |
| `123.45.67.88/30` | — | `s0` | ISP |
| `0.0.0.0/0` | `123.45.67.90` | — | Default |

Per quanto riguarda il NAT, questo andrà configurato in modo che le due interfacce `e1.0` ed `e1.1` vengano tradotte sull'unico IP disponibile su `s0`.
La politica di firewall è molto permissiva. Possiamo supporre di imporre soltanto delle limitazioni in ingresso su `s0`, per bloccare gli accessi indesiderati sulla server farm. Le due intranet hanno indirizzi privati, quindi non risulteranno comunque accessibili dall'esterno, e non ci sarebbe bisogno di esplicitarle sull'ACL.

| Destinazione | Flag | Azione | |
|---|---|---|---|
| `195.41.64.128/26 : 22,80` | — | Allow | Porte ammesse su server farm |
| `195.41.64.128/26 : *` | Established | Allow | Connessioni originate da server farm |
| `195.41.64.128/26 : *` | — | Deny | Altro su server farm è bloccato |
| `192.168.1.0/24 : *` | — | Deny | Intranet bloccate |
| `0.0.0.0/0 : *` | — | Allow | Default: DMZ libera |

Si osservi che, dato che l'ACL è installata in ingresso su `s0`, tutte le comunicazioni a livello 3 fra le reti locali sono possibili, come richiesto dall'esercizio.

**Esercizio 8**

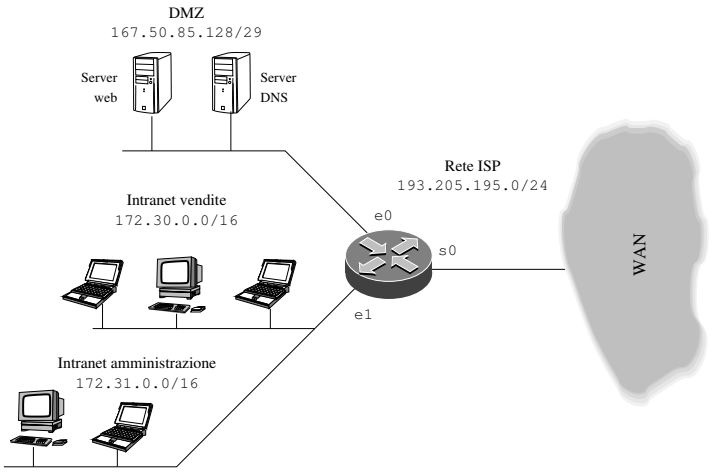
Vogliamo realizzare una rete composta da una DMZ e due Intranet (vendite e amministrazione).

- Requisiti della DMZ:
  - utilizza una sottorete IP pubblica;
  - contiene un server web (porta 80 TCP) e un server DNS (porta 53 UDP) su due macchine distinte;
  - le intranet devono poter accedere liberamente alle macchine della DMZ;
  - dall'esterno, solo i due servizi indicati devono essere accessibili.
- Requisiti delle due intranet:
  - ciascuna deve utilizzare una sottorete IP privata con netmask `255.255.0.0`;
  - possono accedere liberamente alla DMZ;
  - accedono a internet tramite NAT;
  - non devono essere contattabili dalla DMZ.
- Risorse disponibili:
  - un router con due interfacce Ethernet `e0` ed `e1`, e un'interfaccia WAN `s0`, dotato di funzionalità NAT e firewalling;
  - la sottorete IP pubblica `167.50.85.128/29` per la DMZ;
  - l'indirizzo fornito dall'ISP all'interfaccia `s0` del router e il corrispondente default gateway sono rispettivamente l'IP più basso e quello più alto utilizzabili nella sottorete `108.100.199.224/28`;
  - tutti gli switch necessari.

Ogni dato non fornito nell'esercizio può essere stabilito arbitrariamente (se possibile, motivando la scelta).
**8.1)** Disegnare uno schema di massima della rete evidenziando la posizione delle varie reti, fisiche e virtuali.
**8.2)** Indicare la configurazione (indirizzo IP, netmask, default gateway) dei due server della DMZ, e di un terminale per ciascuna delle intranet.
**8.3)** Indicare la configurazione delle interfacce del router: eventuale suddivisione in interfacce logiche, indirizzo IP e netmask.
**8.4)** Indicare la configurazione interna del router: tabelle di instradamento, NAT, ACL.

**Soluzione 8**

**8.1)** — schema di massima
Scegliamo arbitrariamente di assegnare l'interfaccia `e0` del router alla DMZ; l'interfaccia `e1` andrà assegnata alle due intranet, quindi in seguito dovremo organizzare due VLAN. Per le intranet scegliamo le reti `172.30.0.0/16` e `172.31.0.0/16`.
Lo schema risultante è il seguente; decideremo in seguito altri dettagli:




**Possibili varianti**

Non ci sono particolari vincoli nella scelta degli IP per le due intranet, a patto che vengano scelte nei blocchi CIDR privati: `10.0.0.0/8`, `172.16.0.0/12`, `192.168.0.0/16`. Anche se l'esercizio richiede due reti di tipo `/16`, il testo potrebbe essere interpretato diversamente, ad esempio utilizzando una singola rete `/16` da ripartire in due. Si è scelto di accettare ogni interpretazione, purché le due reti siano separate, entrambe private, e con delle netmask corrette.

**8.2)** — Configurazioni terminali

Per quanto riguarda la DMZ, la sottorete è fornita dal testo, e gli IP utilizzabili per le interfacce sono quelli nell'intervallo `167.50.85.129` — `167.50.85.134`. Scegliamo, arbitrariamente, di utilizzare come IP dei default gateway quelli più alti; una possibile configurazione è la seguente:

| Terminale | Indirizzo IP | Netmask | Default gateway |
|---|---|---|---|
| Server web | `167.50.85.129` | `255.255.255.248` | `167.50.85.134` |
| Server DNS | `167.50.85.130` | `255.255.255.248` | `167.50.85.134` |
| Terminale vendite | `172.30.34.213` | `255.255.0.0` | `172.30.255.254` |
| Terminale amministrazione | `172.31.218.112` | `255.255.0.0` | `172.31.255.254` |

**Possibili varianti** — Gli indirizzi IP e i default gateway sono completamente arbitrari, ma tutti devono essere indirizzi IP validi all'interno delle rispettive sottoreti.

**8.3)** — Interfacce del router

Disponendo di una sola interfaccia per la gestione delle due intranet, il router deve suddividere `e1` in due interfacce logiche, ad esempio `e1.30` per le vendite ed `e1.31` per l'amministrazione. Gli indirizzi dele interfacce ethernet devono corrispondere ai default gateway scritti nel punto precedente (servono esattamente a questo!).

L'esercizio dice che l'interfaccia `s0` deve avere l'indirizzo più basso disponibile nel blocco CIDR fornito dall'ISP, che copre gli indirizzi `108.100.199.225` — `108.100.199.238`.

| Interf. fisica | Interf. logica | ID 802.1Q | IP | Netmask |
|---|---|---|---|---|
| `s0` | | | `108.100.199.225` | `255.255.255.240` |
| `e0` | | | `167.50.85.134` | `255.255.255.248` |
| `e1` | `e1.30` | 30 | `172.30.255.254` | `255.255.0.0` |
| | `e1.31` | 31 | `172.31.255.254` | `255.255.0.0` |

**Possibili varianti** — A parte i nomi delle interfacce logiche e gli ID delle VLAN, il resto dei valori è già stato determinato nei passi precedenti.

**8.4)** — Configurazione avanzata del router

In base alle richieste dell'esercizio, il default gateway imposto dall'ISP è l'indirizzo più alto ammissibile dal blocco CIDR, quindi la tabella di instradamento è completamente determinata dai passi precedenti:

| Destinazione | Interfaccia | Gateway | |
|---|---|---|---|
| `167.50.85.89/29` | `e0` | — | DMZ |
| `172.30.0.0/16` | `e1.30` | — | Vendite |
| `172.31.0.0/16` | `e1.31` | — | Amministrazione |
| `108.100.199.224/28` | `s0` | — | Rete dell'ISP |
| `0.0.0.0/0` | `s0` | `108.100.199.238` | Default |

Per quanto riguarda il NAT, la DMZ non ne ha bisogno (ha IP pubblici). Le due intranet, invece, possono uscire solo grazie ad esso:

- interfaccia esterna ("outside"): `s0`;
- interfacce interne ("inside"): `e1.30` ed `e1.31`;
- pool di indirizzi pubblici: `108.100.199.225` (unico IP pubblico disponibile su `s0`);
- pool di indirizzi privati mappabili:
  - `172.30.0.1`–`172.30.255.253`
  - `172.31.0.1`–`172.31.255.253`

Le ACL possono essere organizzate in molti modi. Una forma minima, nell'ipotesi che vi sia un'unica tabella centralizzata, può essere la seguente:

| Prot. | Sorgente | Destinazione | Flag | Azione | |
|---|---|---|---|---|---|
| * | `172.30.0.0/15:*` | * | * | PASS | Prov. Intranet |
| * | * | `172.30.0.0/15:*` | Established | PASS | Dest. Intranet |
| TCP | * | `167.50.85.129/32:80` | * | PASS | Dest. server web |
| TCP | `167.50.85.129/32:80` | * | Established | PASS | Prov. server web |
| UDP | * | `167.50.85.130/32:53` | * | PASS | Dest. server DNS |
| UDP | `167.50.85.130/32:53` | * | Established | PASS | Prov. server DNS |
| UDP | `167.50.85.130/32:*` | `0.0.0.0/0:53` | * | PASS | Query DNS |
| * | * | * | * | DROP | Default |

Le prime due righe lasciano libertà d'azione alle due intranet (si noti che il blocco CIDR `172.30.0.0/15` le copre entrambe), che sono comunque protette dal NAT. Le due righe successive regolano i restanti rapporti del server web, che può solo essere contattato alla porta 80. Idem per il DNS, con una riga aggiuntiva che permette al server di inoltrare query all'esterno. Per default, ovviamente, si blocca tutto.

**Possibili varianti e punti di attenzione** — Attenzione alle ultime due righe della tabella di instradamento: spesso si dimentica la riga corrispondente alla rete dell'ISP.

L'indicazione delle interfacce coinvolte dal NAT è sufficiente, anche senza l'elenco degli indirizzi. Sono possibili molte varianti di ACL.

**Esercizio 9**

Alice e Bob avviano il protocollo Diffie-Hellman per la creazione di una chiave condivisa di sessione. Data la consueta notazione ($p$ primo, $g$ base, $a$ segreto di Alice, $A = g^a \pmod p$, $b$ segreto di Bob, $B = g^b \pmod p$),

- Alice spedisce a Bob la terna $p = 11$, $g = 5$, $A = 3$;
- Bob spedisce ad Alice $B = 4$.

**9.1)** Qual è la chiave segreta condivisa?
**9.2)** Tralasciando la lunghezza della chiave, gli altri requisiti di sicurezza del protocollo sono stati rispettati?

**Soluzione 9**

**9.1)** Dobbiamo calcolare almeno $a$ oppure $b$. Visto che il numero $p$ è piccolo, calcoliamo tutte le potenze di $g$ modulo $p$, ciascuna ottenuta dalla precedente moltiplicando il risultato per $g$ e trovando il resto della divisione per $p$:

$$\begin{array}{lll} 5^1 = 5 & 5^4 = 9 & 5^7 = 3 \\ 5^2 = 3 & 5^5 = 1 & 5^8 = 4 \\ 5^3 = 4 & 5^6 = 5 & 5^9 = 9 \\ & & 5^{10} = 1 \end{array}$$

Guardando la tabella, scegliamo $a = 2$ e $b = 3$. Si noti che la scelta non è univoca, segno che $g$ non è stato scelto con la dovuta cautela (lo vedremo al punto successivo). Possiamo scegliere se preferiamo calcolare $K = A^b = 3^3 = 5 \pmod{11}$, oppure $K = B^a = 4^2 = 5 \pmod{11}$. I due risultati, ovviamente, coincidono per le proprietà delle potenze nei campi.
**9.2)** Il numero $p = 11$ è primo; come abbiamo visto prima, però, le potenze di $g$ modulo $p$ si ripetono e non coprono l'intero di interi modulo $p$. Un attacco di forza bruta è quindi più semplice del necessario (basta calcolare le prime 5 potenze).

**Esercizio 10**

Per la creazione di una chiave condivisa basata sul protocollo di Diffie-Hellman, Bob riceve da Alice il messaggio pubblico ($p = 17, g = 3, A = 11$); di seguito, genera il proprio valore segreto $b = 14$.
**10.1)** Verificare che $g$ è una radice prima modulo $p$.
**10.2)** Che messaggio spedisce Bob ad Alice?
**10.3)** Quanto vale la chiave condivisa?

**Soluzione 10**

**10.1)** Verifichiamo che le prime $p-1$ potenze di $g$ modulo $p$ sono tutte diverse:

$$\begin{array}{cccc} 3^1 = 3 & 3^5 = 5 & 3^9 = 14 & 3^{13} = 12 \\ 3^2 = 9 & 3^6 = 15 & 3^{10} = 8 & 3^{14} = 2 \\ 3^3 = 10 & 3^7 = 11 & 3^{11} = 7 & 3^{15} = 6 \\ 3^4 = 13 & 3^8 = 16 & 3^{12} = 4 & 3^{16} = 1 \end{array}$$

**10.2)** Dato il valore segrato di Bob, il valore pubblico è ottenuto esaminando la tabella del punto precedente:

$$B = g^b = 3^{14} \equiv 2 \pmod{p}.$$

**10.3)** La chiave può essere calcolata come $K = A^b = 11^{14} \pmod{p}$, oppure più comodamente come $K = B^a \pmod{p}$. Il valore di $a$ può essere ricavato dalla tabella al punto 1 sapendo che $3^a = A = 11$, da cui $a = 7$. Di conseguenza, $K = 2^7 = 128 \equiv 9 \pmod{17}$.

**Esercizio 11**

Alice deve inviare a Bob una sequenza di tre numeri $m_1, m_2, m_3$. Per farlo in modo sicuro stabilisce con Bob una chiave condivisa utilizzando il protocollo di Diffie-Hellman, poi "cifra" i numeri sommando loro la chiave (ovviamente in $\mathbb{Z}$, non in aritmetica modulare) prima di spedirli.
Charlie vede passare i seguenti messaggi:

- Da Alice a Bob: $p = 11$, $g = 3$, $A = 5$;
- Da Bob a Alice: $B = 4$;
- Da Alice a Bob: I valori cifrati sono: $c_1 = 27$, $c_2 = 48$, $c_3 = 90$.

**11.1)** Descrivere in che modo Charlie può determinare la chiave e quindi i valori originali attraverso un attacco di forza bruta.
**11.2)** La scelta di $p$ e $g$ da parte di Alice è corretta?

**Soluzione 11**

**11.1)** Charlie deve dedurre uno dei numeri segreti, quindi comincia a calcolare le potenze di 3 (modulo 11) finché non ottiene uno dei due numeri pubblici $3^a \equiv A \pmod{11}$ o $3^b \equiv B \pmod{11}$ (d'ora in poi, uso il segno di equivalenza "$\equiv$" per indicare l'uguaglianza in $\mathbb{Z}_{11}$:

$$3^1 \equiv 3, \qquad 3^2 \equiv 3 \cdot 3 \equiv 9, \qquad 3^3 \equiv 9 \cdot 3 \equiv 27 \equiv 5 = A.$$

Charlie ha dunque scoperto il segreto di Alice: $a = 3$. Passa dunque a calcolare la chiave esattamente come farebbe Alice:

$$K \equiv B^a \equiv 4^3 \equiv 64 \equiv 9.$$

A questo punto sa che i codici di Alice sono ottenuti sommando la chiave ai valori reali: $c_i = m_i + K$:

$$m_1 = c_1 - K = 27 - 9 = 18; \qquad m_2 = c_2 - K = 48 - 9 = 39; \qquad m_3 = c_3 - K = 90 - 9 = 81.$$

**11.2)** Le ipotesi perché la chiave sia robusta sono:

1. $p = 11$ dev'essere primo (soddisfatta);
2. $g = 3$ dev'essere una radice prima modulo $p$, ovvero le sue potenze devono generare tutti i valori da 1 a $p - 1$. Abbiamo già calcolato le prime tre potenze di $g$; vediamo le successive:

   $$3^4 \equiv 5 \cdot 3 \equiv 15 \equiv 4, \qquad 3^5 \equiv 4 \cdot 3 \equiv 12 \equiv 1.$$

   Quindi la quinta potenza di $g$ è tornata a valere 1, e da qui il ciclo non può che ripetersi: solo i numeri 1, 3, 9, 5 e 4 vengono generati da $g$, che quindi non è radice prima.

**Esercizio 12**

Come misura di contrasto alla pirateria, un sistema per lo scambio di file musicali prevede che un utente possa cedere un proprio brano musicale a un altro utente solo dopo averlo firmato. La firma consiste nel calcolare un hash crittografico a 48 bit del file e nel cifrarlo con la propria chiave privata. Alice e Bob amano la musica classica. Charlie, venuto in possessso di un brano di musica da camera barocca firmato da Alice, vuole screditarla agli occhi di Bob usando la stessa firma su un brano di acid house music.

**12.1)** Discutere la fattibilità del disegno criminoso nell'ipotesi che Charlie sia in grado di calcolare un miliardo di funzioni hash al secondo.

**12.2)** Che cosa cambia se Charlie ha a disposizione dieci, cento, oppure mille brani firmati da Alice?

Suggerimento — *Ricordare l'approssimazione* $2^{10} \approx 10^3$.

**Soluzione 12**

**12.1)** Trattandosi di un contesto in cui dev'essere possibile verificare una firma digitale, possiamo assumere che l'algoritmo di hashing e la chiave pubblica di Alice siano noti a tutti, anche a Charlie. L'unico dato in mano di Charlie è una firma di Alice. Detto $f_A$ il file di Alice, $H(f_A)$ il suo valore hash, $\mathrm{RSA}_{K_{sA}}(H(f_A))$ la firma di Alice, l'unica possibilità da parte di Charlie è produrre una variante del proprio file $f_C$ tale che $H(f_C) = H(f_A)$. Purtroppo, la funzione hash è crittografica, quindi non permette di generare collisioni molto facilmente. Una variante casuale del file di Charlie ha probabilità $2^{-48}$ di avere lo stesso hash di $f_A$, quindi mediamente dovrà generare $2^{48}$ varianti. Se può eseguire $10^9 \approx 2^{30}$ controlli al secondo, allora impiegherà mediamente $2^{48-30} = 2^{18} = 2^{20}/4 \approx 10^6/4 = 250000$ secondi, cioè circa quattro giorni.

**12.2)** Se Charlie ha a disposizione $n$ brani, allora la collisione diviene più probabile. Supponendo per semplicità che gli hash di tutti i brani di Alice siano diversi, la probabilità diventa $2^{-48}n$, quindi il tempo atteso va diviso per $n$ (rispettivamente 25000, 2500 e 250 secondi).

Si noti che Charlie non ha nemmeno bisogno di conoscere firme vere: può benissimo generare un valore casuale $s'$, trovare il valore hash $h' = \mathrm{RSA}_{K_{pA}}(s')$ di cui esso sarebbe firma (il passaggio da firma a hash è facile per tutti) e trovare un brano $f'$ tale che $h' = H(f')$.