Sicurezza dei sistemi e delle reti[1]

Mattia Monga

Dip. di Informatica
Università degli Studi di Milano, Italia
mattia.monga@unimi.it

a.a. 2014/15

Lezione I: Introduzione alla sicurezza delle reti

Di cosa parleremo

La sicurezza dei sistemi in rete

- I protocolli TCP/IP dal punto di vista della sicurezza
- Analisi delle possibilità di connessione e del traffico di rete
- La sicurezza "perimetrale"
- Rilevamento delle intrusioni
- Protezione di servizi critici e dell'infrastruttura Internet

Di cosa parleremo

La sicurezza nelle reti

- Misure di protezione in una rete untrusted
- VPN
- Anonimato
- Specificità delle reti wireless

Riferimenti

- Il sito del corso:
  https://mameli.docenti.di.unimi.it/sicureti
  SHA-1 fingerprint:
  0E:06:D3:0B:63:71:6A:5D:1C:7D:2F:DC:41:17:4B:89:E6:DE:46:3D
- Testi di riferimento:
  - "Security Engineering" R. Anderson, Wiley 2008 (Disponibile anche gratuitamente: http://www.cl.cam.ac.uk/~rja14/book.html)
  - "The Tao of Network Security Monitoring – Beyond Intrusion Detection" R. Bejtlich Pearson Education Inc., 2004
  - "Silence on the Wire. A Field Guide to Passive Reconnaissance and Indirect Attacks", M. Zalewski, No Starch Press, 2005
- Articoli scientifici indicati a lezione (e sul sito)

Modalità d'esame

- prova scritta + prova di laboratorio: il voto finale 75% scritto + 25% laboratorio (entrambe devono essere sufficienti).
- Scritto: domande e svolgere esercizi sul programma trattato a lezione e gli approfondimenti indicati.
- Laboratorio: esercizio da risolvere con i tool trattati a lezione

Riassumendo

Il titolo del corso è assai vasto... Ci concentreremo su:

- Reti TCP/IP
- Le misure classiche di protezione "perimetrale"
- La sicurezza nelle reti

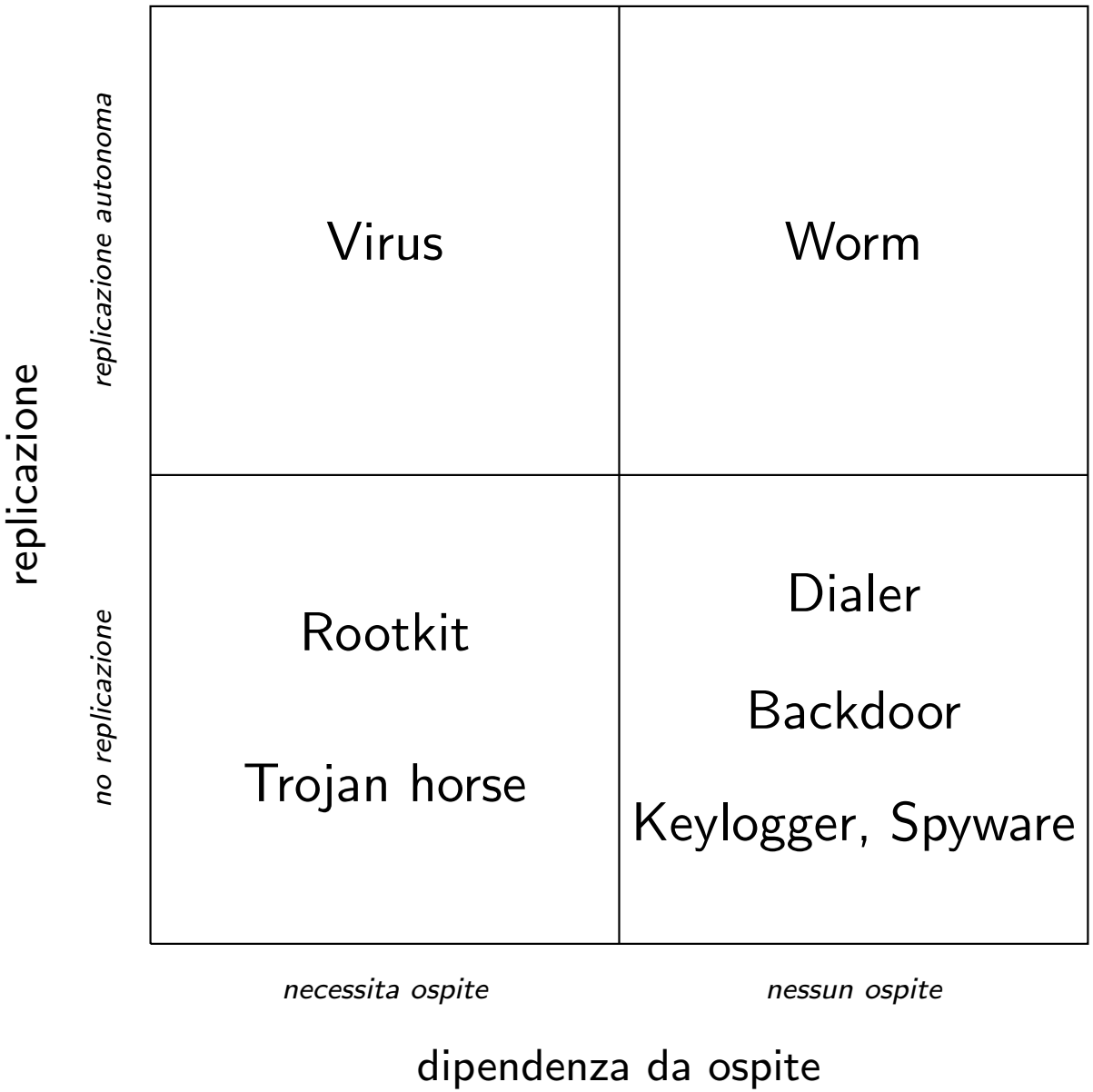
Sicurezza & reti

Storicamente il grande pubblico ha iniziato a parlare di sicurezza informatica in congiunzione con la diffusione delle reti. L'evento simbolo Rete/sicurezza dei sistemi è l'Internet Worm (2 novembre 1988). Colpí qualche migliaio di macchine ed è considerato il giorno della *perdita dell'innocenza di Internet*.

In realtà il problema era già ben noto: la legge grazie alla quale R. Morris fu condannato era del 1986.
Letture obbligatorie:

- Joyce Reynolds; *The Helminthiasis of the Internet;* RFC 1135; Dec. 1989.

Tipologie di malware



malware: codice progettato per danneggiare intenzionalmente un sistema, alterandone funzionalità o dati.

Symantec Threat Report 2013

http://www.symantec.com/threatreport/

- 42% increase in targeted attacks in 2012.
- 6787 (2012: 5291) new vulnerabilities discovered in 2013, 127 (2012: 416) of them on mobile operating systems.
- Spam volume continued to decrease, with 66% of all email being spam (-3%)
- Number of bots detected (2.3M) decreasing (-33%)

Chi ha interesse a colpire un sistema?

Secondo *Verizon 2014 Data Breach Investigations Report* (contiene anche i dati dell'USSS, e di varie polizie) 63437 incidenti + 1367 *data breaches*. Il report è molto cambiato: nel 2013 conteneva solo 621 incidenti.

2010	2011	2012	2013	
70%	92%	98%	92%	causato da agenti esterni
48%	17%	4%	14%	causato da interni
11%	$< 1\%$	$< 1\%$	1%	causato da business partner
40%	17%	21%	25%	attacchi mirati
38%	49%	69%	40%	causato da malware

19% 'state-affiliated'...

Geni del crimine?

Altri dati interessanti (dal report 2013): 99% delle compromissioni iniziali non è da considerarsi *molto difficile* e la grande maggioranza (78%) usa tecniche di base o strumenti automatici...

Lettura obbligatoria

Verizon Data Breach Investigations Report
http://www.verizonenterprise.com/DBIR/2014/

Un'altra data simbolo?

Una nuova data simbolo: il 7 giugno 2013 Edward J. Snowden rivela che NSA ha raccolto informazioni riguardanti comunicazioni private con l'aiuto di Microsoft, Yahoo, Google, Facebook, PalTalk, YouTube, Skype, AOL, Apple (programma PRISM). Seguono decine di altre rivelazioni:
http://www.lawfareblog.com/catalog-of-the-snowden-revelations/.
Per approfondire: "No place to hide" Glenn Greenwald (in italiano: "Sotto controllo", Rizzoli).

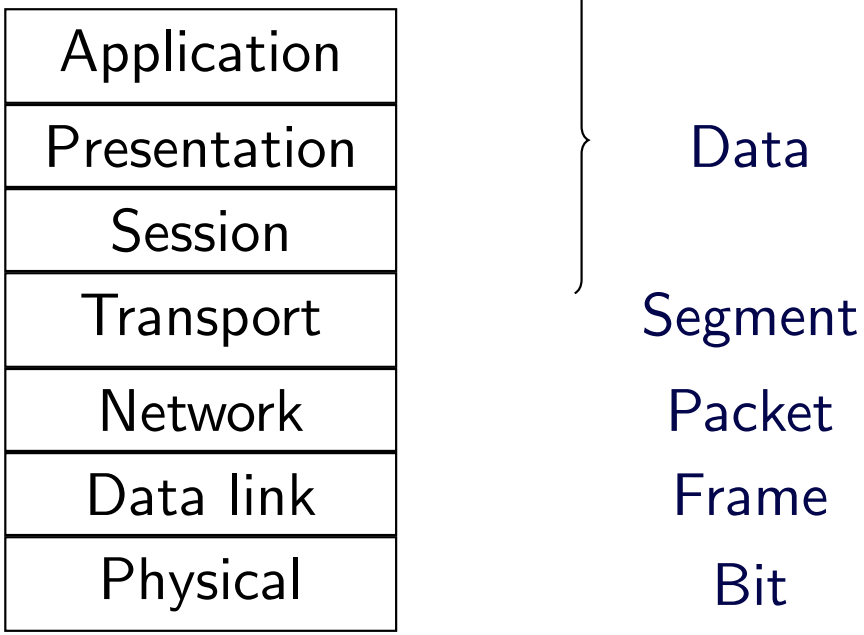
Riassumendo

Sicurezza informatica è sempre piú un problema con risvolti sociali, oltre che tecnologici

- Non solo da attacchi mirati, ma comunque attacchi con obiettivi commerciali
- Malware “di massa”
- Spionaggio industriale e attacchi alle libertà civili

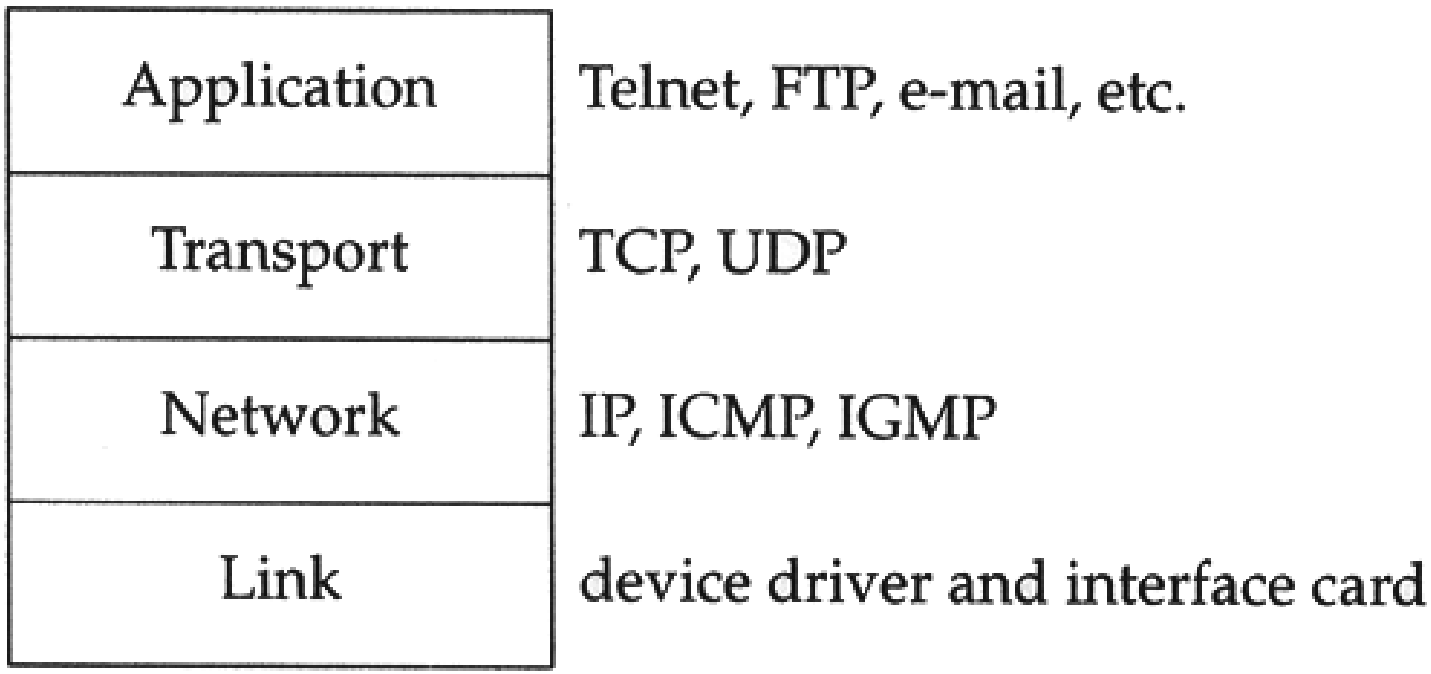
Lezione II: Il modello di riferimento

Il modello di riferimento OSI



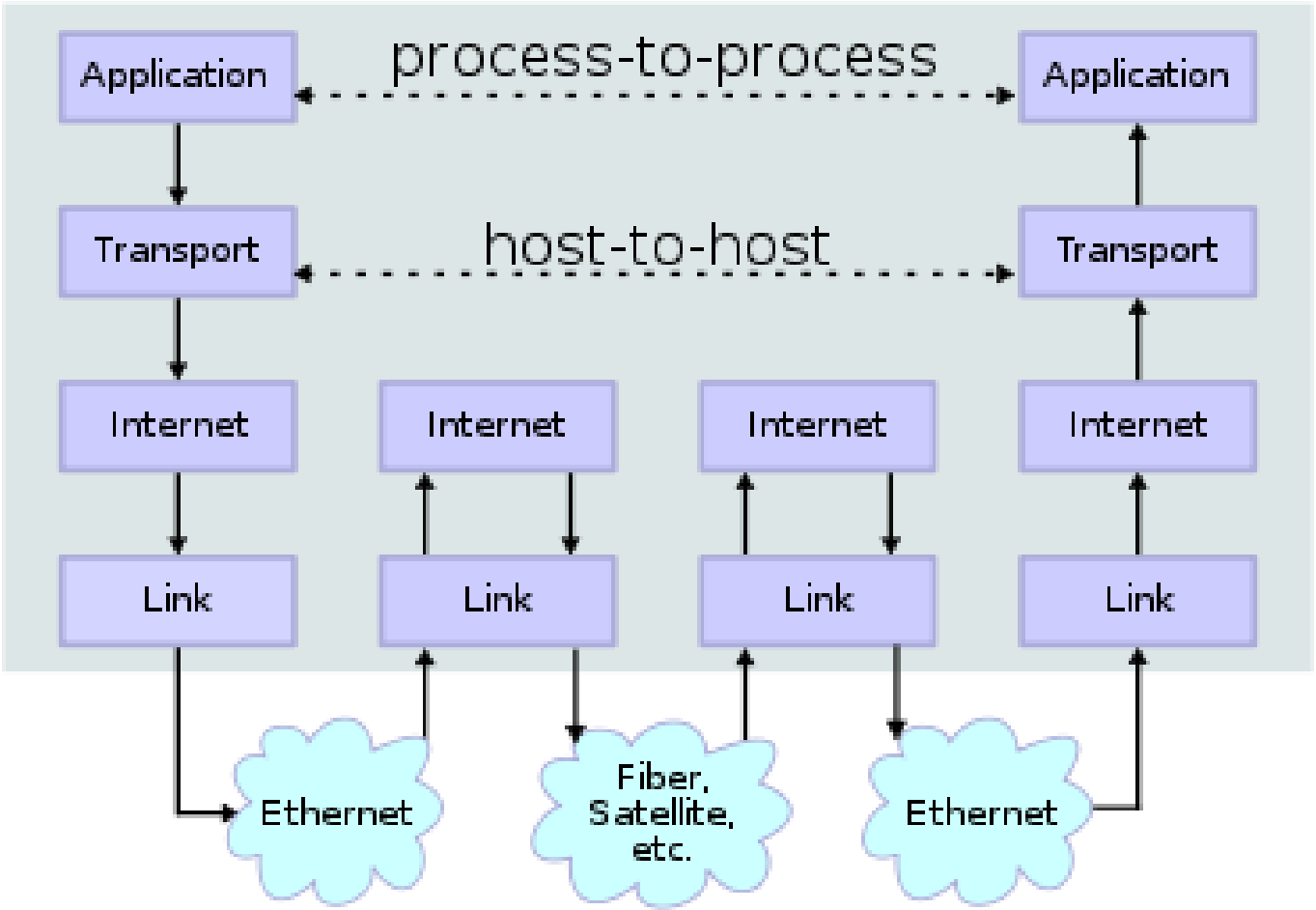
Stack dei protocolli Internet

Un modello semplificato (*TCP/IP Illustrated*, W. Stevens)

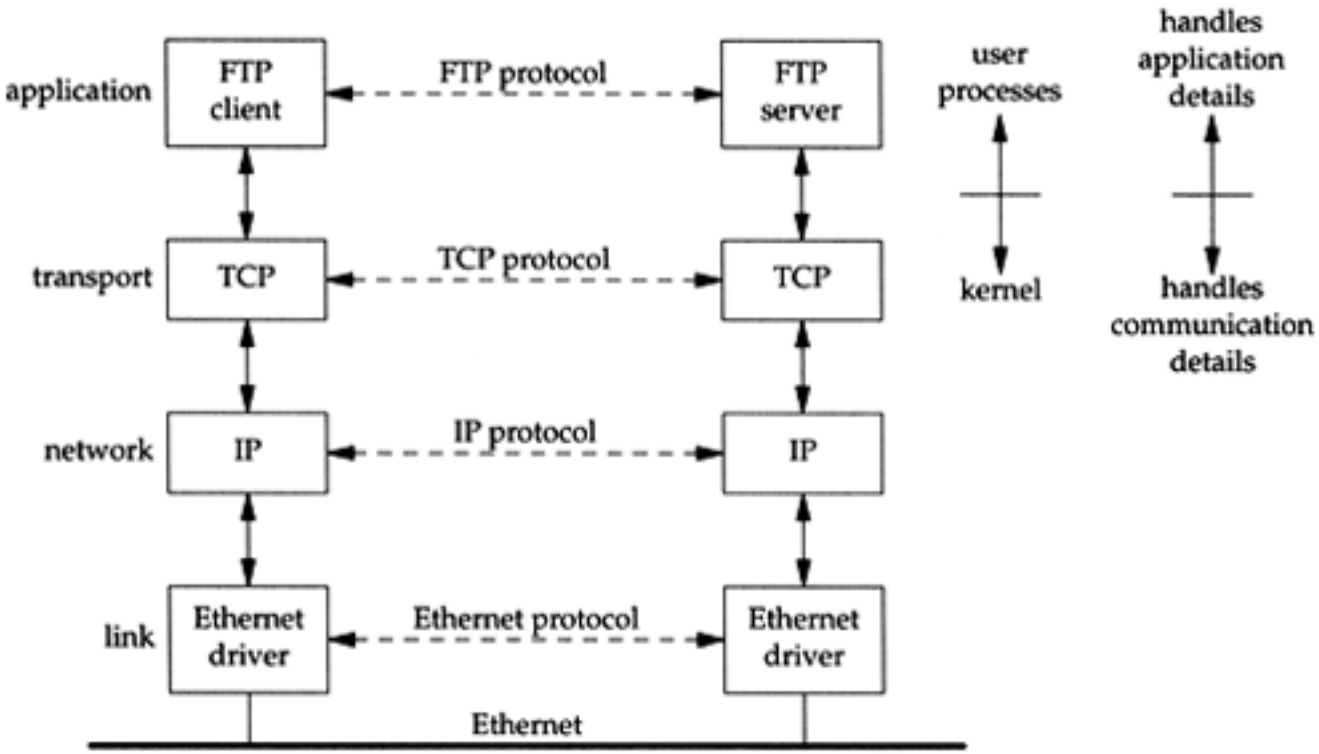


È uno stack narrow waist: lo strato IP è molto difficile da cambiare.

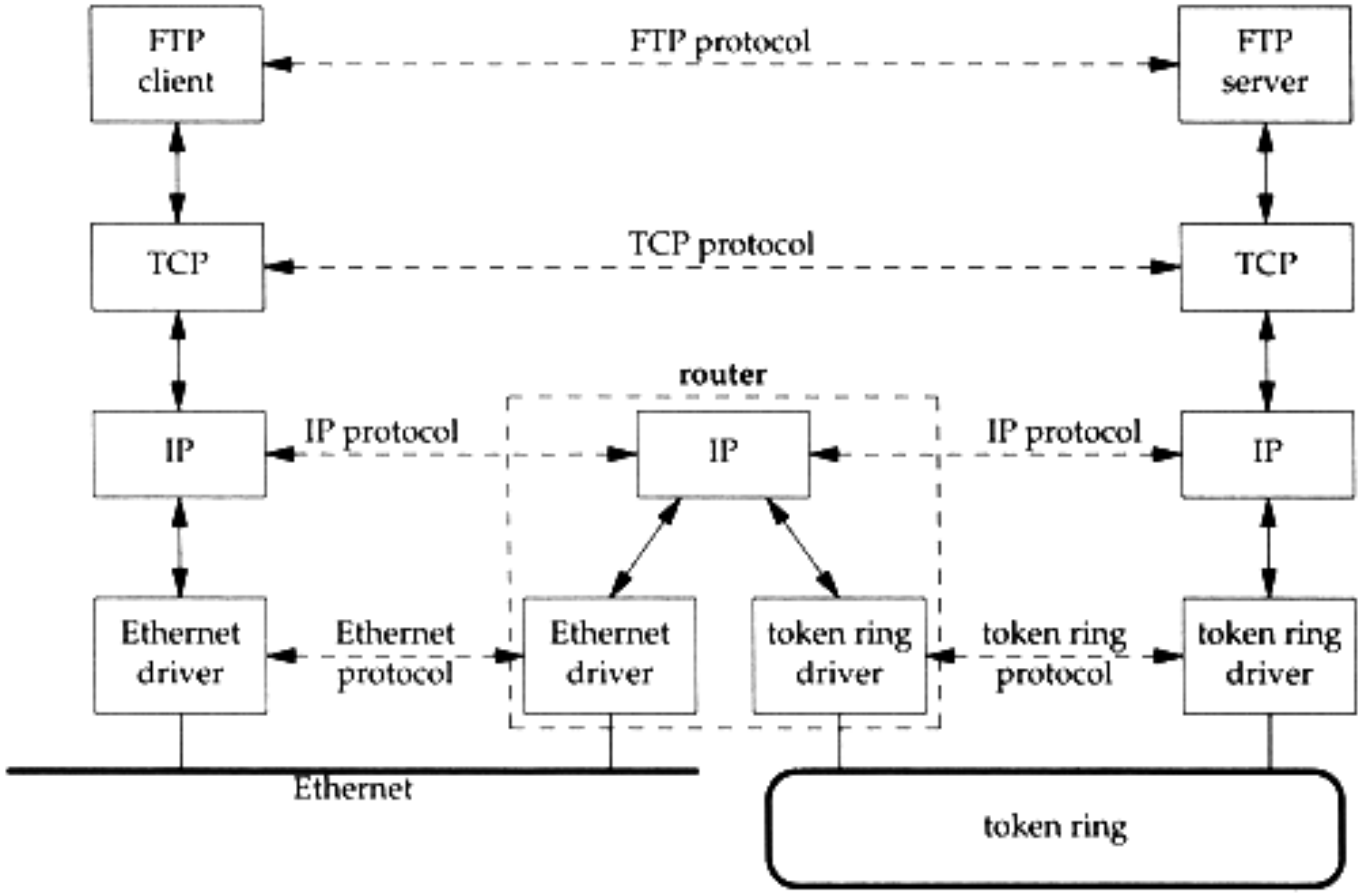
Stack dei protocolli Internet



Stack dei protocolli Internet



Stack dei protocolli Internet



Principi architetturali

end-to-end principle L'*intelligenza* ai vertici della rete, che trasmette i dati nella maniera piú efficiente;

robustness approach Conservatori nel mandare, liberali nel ricevere.

Lettura obbligatoria: E. Allman. *The robustness principle reconsidered.* CACM 54, 8 (August 2011), 40–45.

Riassumendo

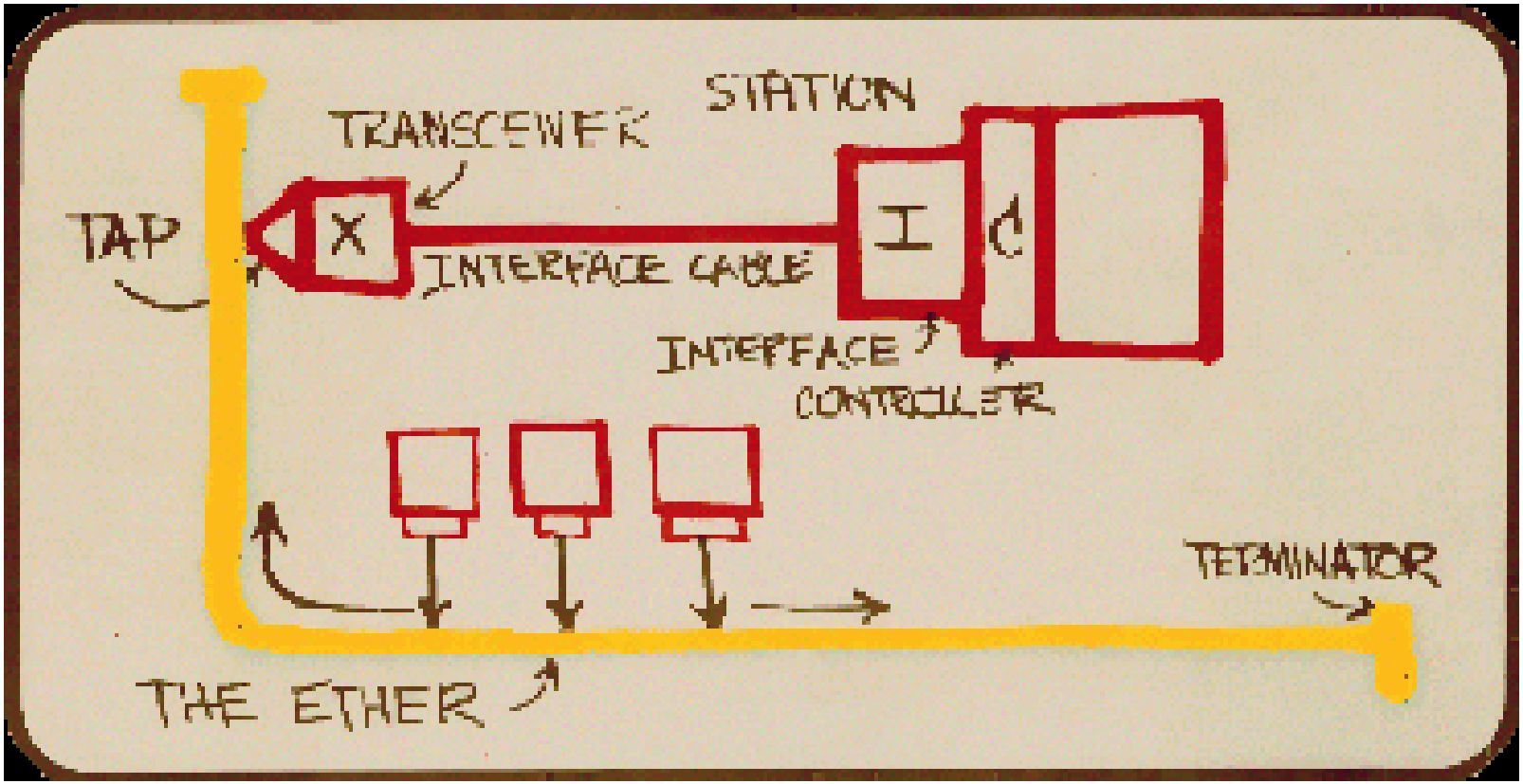
La suite TCP/IP è basata su:

- 4 livelli: *link*, *network*, *transport*, *application*
- *end-to-end principle*
- *robustness approach*

Lezione III: Dal livello link a quello di trasporto

Ethernet

comunicare tramite un medium condiviso (analogo al famigerato *etere*)



Caratteristiche del protocollo

- Carrier Sense, Multiple Access with Collision Detection
- indirizzi a 48 bit
- maximum transmission unit (MTU): 1500 bytes
- . . . ma c'è anche una dimensione minima: 46 byte (ciò costringe al *padding*)

Dal punto di vista della sicurezza

Tutti i nodi connessi (LAN) ricevono tutti i frame: scartano quelli non diretti a loro.

Hub e switch

L'estensione della rete si amplia con:

Hub semplici ripetitori di segnale (tutt'al piú aiutano nella *collision detection* producendo i *jam* frame)

Hub e switch

L'estensione della rete si amplia con:

Switch definiscono diversi *collision domain*: logicamente LAN differenti, che non condividono fra loro il medium

MAC

Le schede di rete sono identificate da un numero seriale, che viene utilizzato come indirizzo all'interno della LAN.

- 48 bit, i primi 24 identificano il produttore
- notazione esadecimale (`MAC: 00:23:a2:d6:f2:15 (Motorola Mobility, Inc.)`)

Affidabilità dei MAC number

Avendo i privilegi adeguati, è quasi sempre possibile (e facile) cambiare il numero MAC usato nella produzione dei frame
Quindi: conoscendo il MAC di una macchina assente, è immediato *impersonarla*.

Sicurezza con gli switch

La separazione dei collision domain è, in definitiva, solo logica.

- La separazione è ottenuta tramite una CAM (*content addressable memory*) che contiene le associazioni MAC-porta dello switch

MAC flooding

Se la tabella è generata dinamicamente (molto comodo: basta attaccare i nodi allo switch, l'amministratore divide i collision domain per porta) è possibile saturarla.
La tabella satura non viene utilizzata!

Riassumendo

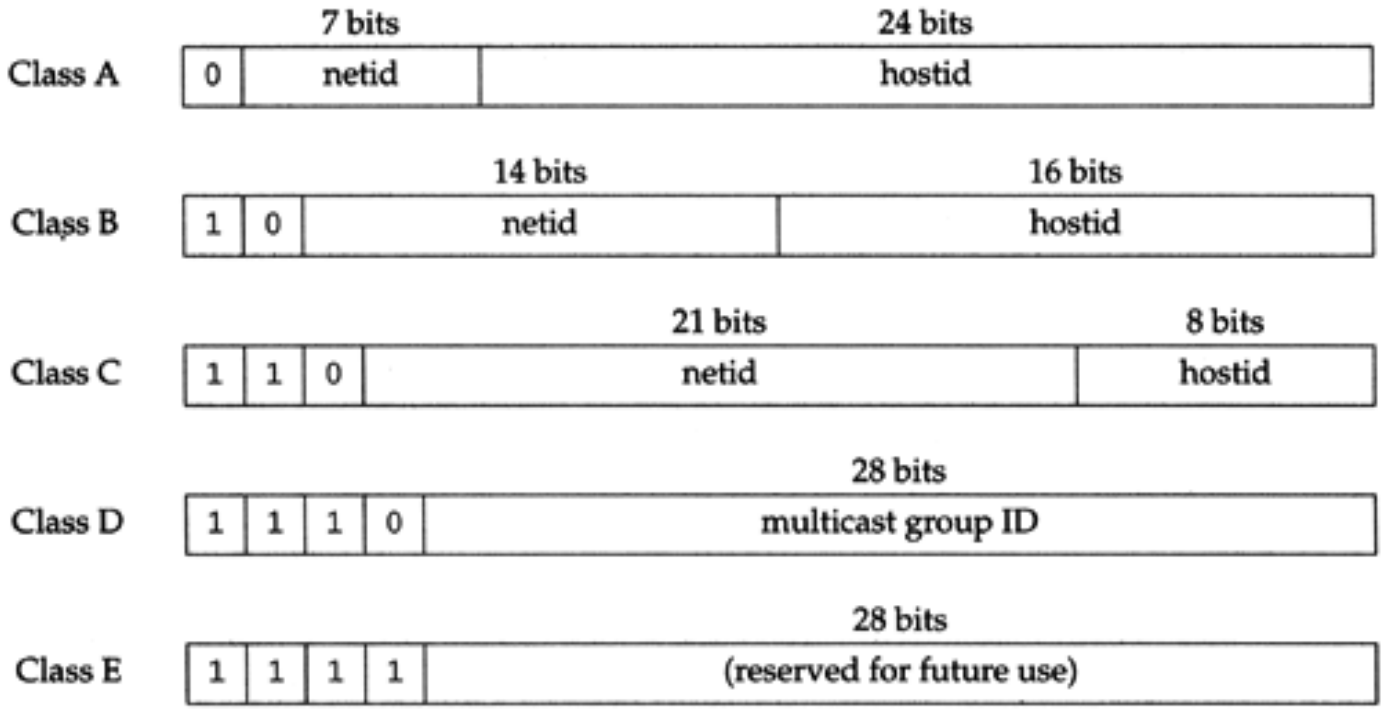
- Le reti locali assumono che i nodi collegati condividano una relazione di fiducia
- I numeri MAC sono un identificatore debole
- MAC flooding: permette di violare i collision domain imposti dagli switch

IP

Occorre istradare i pacchetti fra media differenti.

- Ogni nodo è identificato da un numero IP da 32 bit (IPv4), tradizionalmente scritto come 4 ottetti (notazione in base 256)
- L'istradamento (*routing*) avviene tramite nodi gateway che si interfacciano con due o piú LAN

Classi di indirizzo



Classi di indirizzo

Classe	intervallo	uso
A	0.0.0.0–127.255.255.255	reti tradizionali
B	128.0.0.0–191.255.255.255	reti tradizionali
C	192.0.0.0–223.255.255.255	reti tradizionali
D	224.0.0.0–239.255.255.255	multicast
E	240.0.0.0–255.255.255.255	altri usi speciali

Classi di indirizzo

http://xkcd.com/195/

Classi di indirizzo

Classe	intervallo	uso
A	10.0.0.0–10.255.255.255	intranet
B	172.16.0.0–172.31.255.255	intranet
C	192.168.0.0–192.168.255.255	intranet

Secondo le specifiche i router devono *scartare* (o manipolare...) i pacchetti contrassegnati con questi indirizzi.

Sottoreti e netmask

La netmask è una sequenza di 32 bit che identifica quali bit sono comuni negli IP all'interno di una LAN (sottorete)

`01110111 01110111 01110111 11110111` 119.119.119.247

7 bit per i nodi $2^7 = 128$

CIDR

Normalmente si usano i primi bit (non obbligatorio), quindi è comoda la notazione CIDR (Classless InterDomain Routing)

`159.149.30.0/24` 24 bit per le sottoreti, $32 - 24 = 8$ per gli host

Riassumendo

- Il protocollo IP definisce il formato degli indirizzi dei nodi e delle reti in cui essi si trovano
- Il numero IP contiene entrambi

Lezione IV: Dal livello link a quello di trasporto

ARP

In una rete locale, il numero IP è *superfluo:* è sufficiente (e necessario) il numero MAC.

- ARP (Address Resolution Protocol): numero MAC da un numero IP

Come funziona

- Ogni nodo mantiene una tabella (ARP cache) in cui ci sono le associazioni già note
- altrimenti si chiede a tutti i nodi della rete locale chi ha un certo numero IP

ARP cache poisoning

L'assunzione di trust nella LAN...

1. Chi ha il numero IP 192.168.0.2?
2. Sono io: 00:23:a2:d6:f2:15
3. Le comunicazioni dirette a 192.168.0.2 vanno a chi riceve i frame destinati a 00:23:a2:d6:f2:15

In realtà funziona con arp reply (o anche request!) anche non sollecitate.

Che fare?

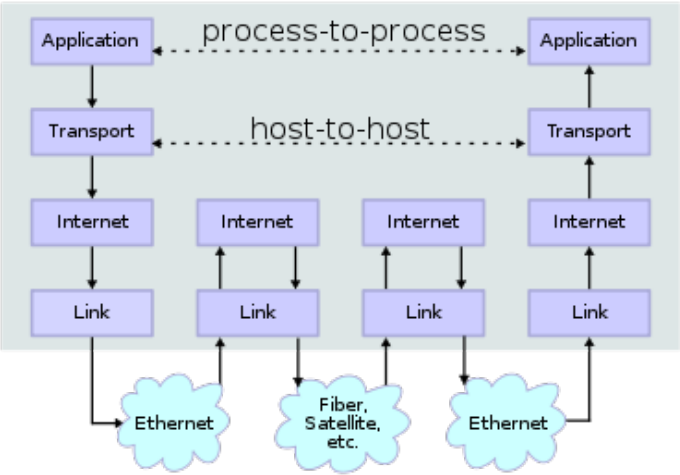
Una possibile difesa è l'uso di tabelle ARP statiche.

Attenzione: l'ARP poisoning ha anche usi perfettamente legittimi: p.es. per ridondanza o per fare convergere il primo collegamento verso un server di autenticazione.

Riassumendo

- Le reti locali assumono che i nodi collegati condividano una relazione di fiducia
- ARP poisoning: permette di *impersonare* uno o piú nodi della LAN

Il livello di trasporto



Poiché a livello applicativo la comunicazione avviene fra processi, a livello trasposto occorre identificare nodi e processi.

Porte

Un segmento di scambio fra due processi necessita di 4 numeri (*socket pair*)

$$\langle ip_1, n_1 : ip_2, n_2 \rangle$$

Port

Port

n_1, n_2 (0–65536) si dicono porte: quelle lato server devono essere note al client e rappresentano quindi il punto *d'accoglienza*.

Nota: il client è il nodo che inizia la connessione con il server.

Porte ben note

da http://www.iana.org/assignments/port-numbers

```
discard 9/tcp sink null
discard 9/udp sink null
ftp-data 20/tcp
ftp 21/tcp
ssh 22/tcp # SSH Remote Login Protocol
ssh 22/udp
telnet 23/tcp
smtp 25/tcp mail
domain 53/tcp # name-domain server
domain 53/udp
finger 79/tcp
www 80/tcp http # WorldWideWeb HTTP
pop3 110/tcp pop-3 # POP version 3
nntp 119/tcp readnews untp # USENET News Transfer Protocol
ntp 123/udp # Network Time Protocol
irc 194/tcp # Internet Relay Chat
https 443/tcp # http protocol over TLS/SSL
printer 515/tcp spooler # line printer spooler
# ...
```

Porte ben note

```
# Non fissate da IANA
socks 1080/tcp # socks proxy server
openvpn 1194/tcp
openvpn 1194/udp
rmiregistry 1099/tcp # Java RMI Registry
# ...
```

Porte convenzionali

Ricordare sempre che le porte sono numeri convenzionali (concordate con IANA per i numeri ≤ 1024)

- in generale non identificano un servizio, ma la possibilità di stabilire una connessione.

Uso delle porte a scopi di sicurezza

- vietare l'uso della porta destinazione 22 non significa vietare SSH, ma impedire che client e server possano accordarsi sull'uso di tale porta.
- il divieto può funzionare solo se l'amministratore controlla il server: se gestisce solo la rete il divieto è aggirabile.

Riassumendo

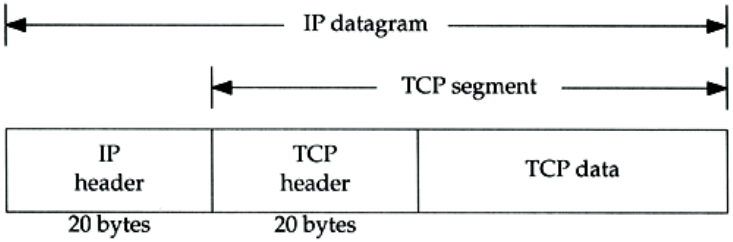
- Una connessione è identificata da 4 numeri $\langle ip_1, n_1 : ip_2, n_2 \rangle$
- Le porte sono semplicemente una convenzione stabilita fra client e server.

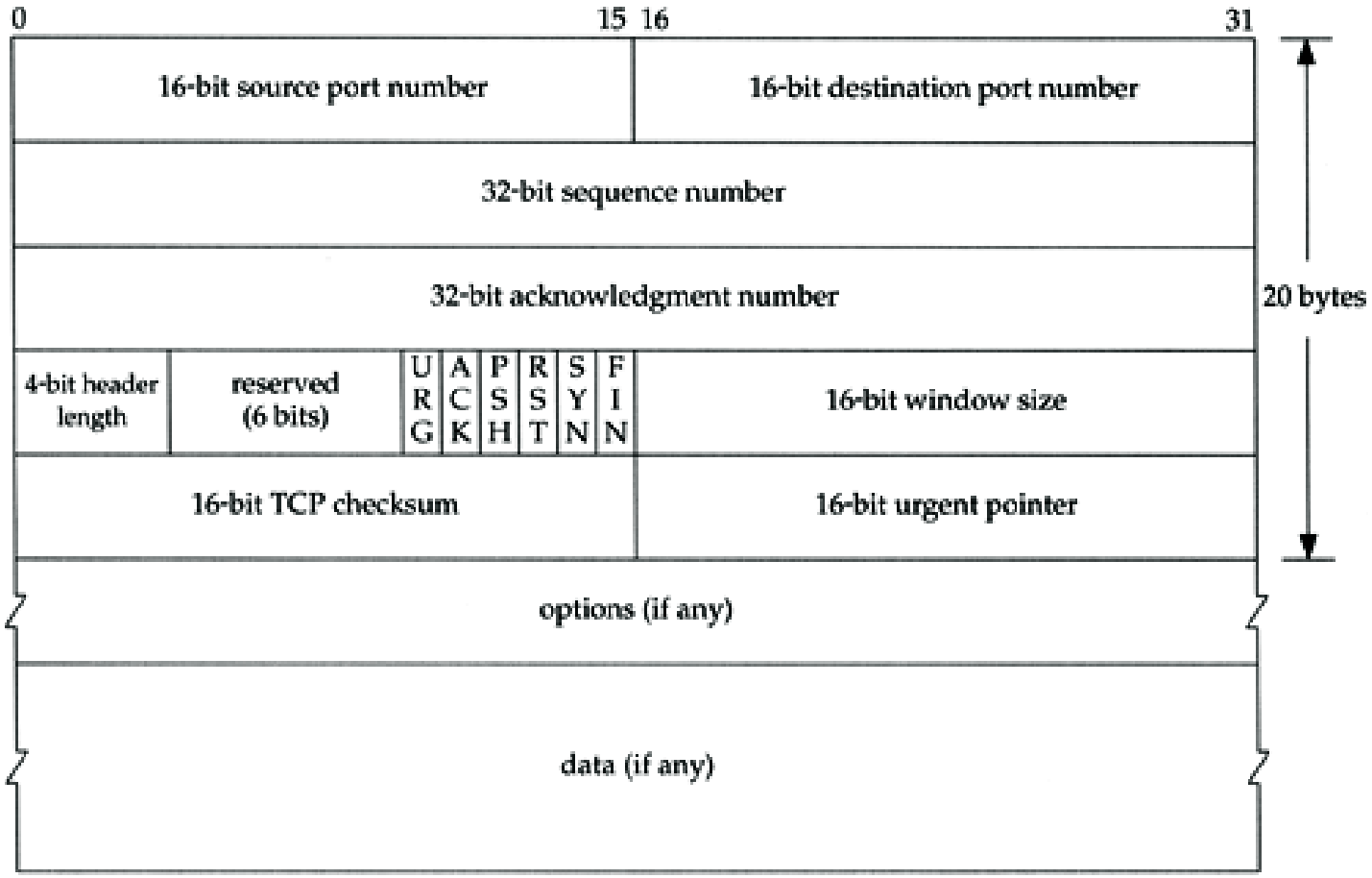
TCP

Transmission Control Protocol

- connection-oriented: è necessario uno handshake preliminare
- full-duplex
- lo "stato" è conservato interamente nei nodi (+ timer)

TCP segment





Flag

SYN richiesta di connessione, sempre il primo pacchetto di una comunicazione

FIN indica l'intenzione del mittente di terminare la sessione in maniera concordata

ACK conferma del pacchetto precedente, sia esso dati, SYN o FIN

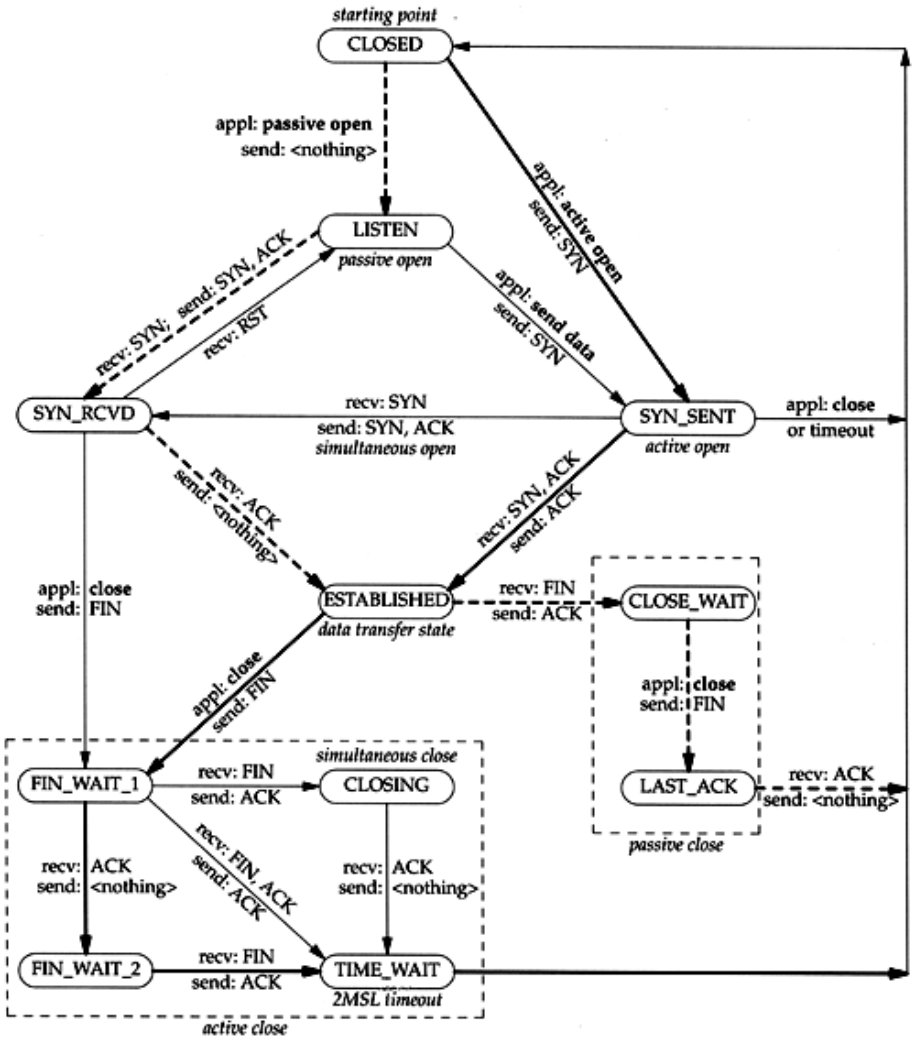
Flag

RST reset della sessione

PSH operazione di push, i dati che vengono inviati al destinatario non dovrebbero essere bufferizzati

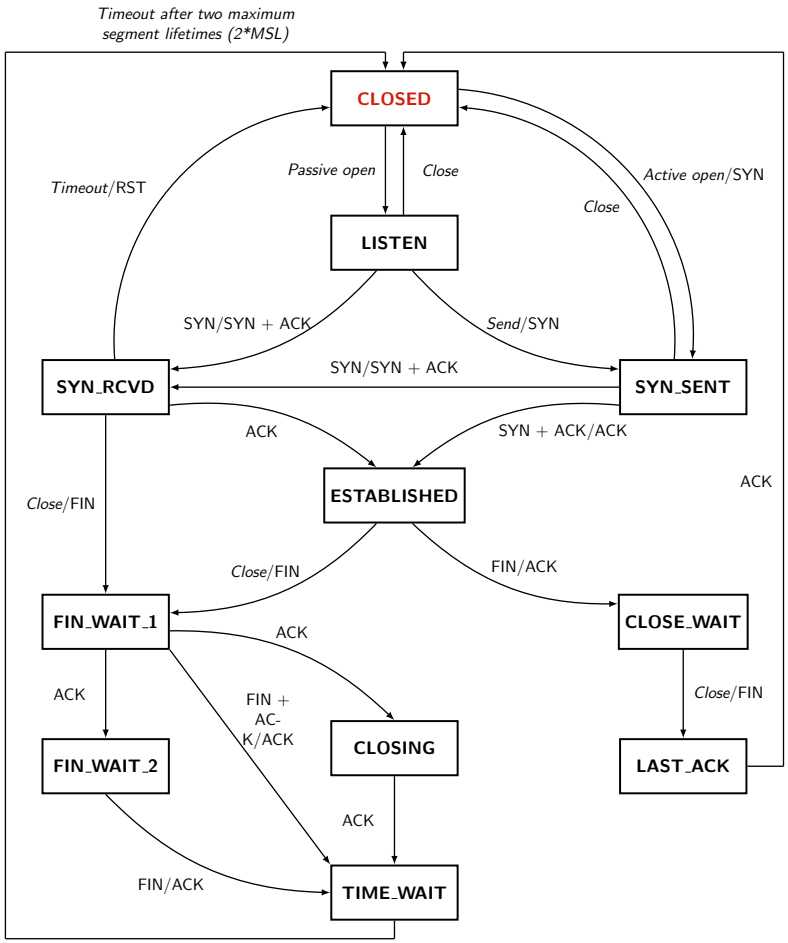
URG dati urgenti (es. CTRL+C) vengono inviati con precedenza sugli altri

State diagram

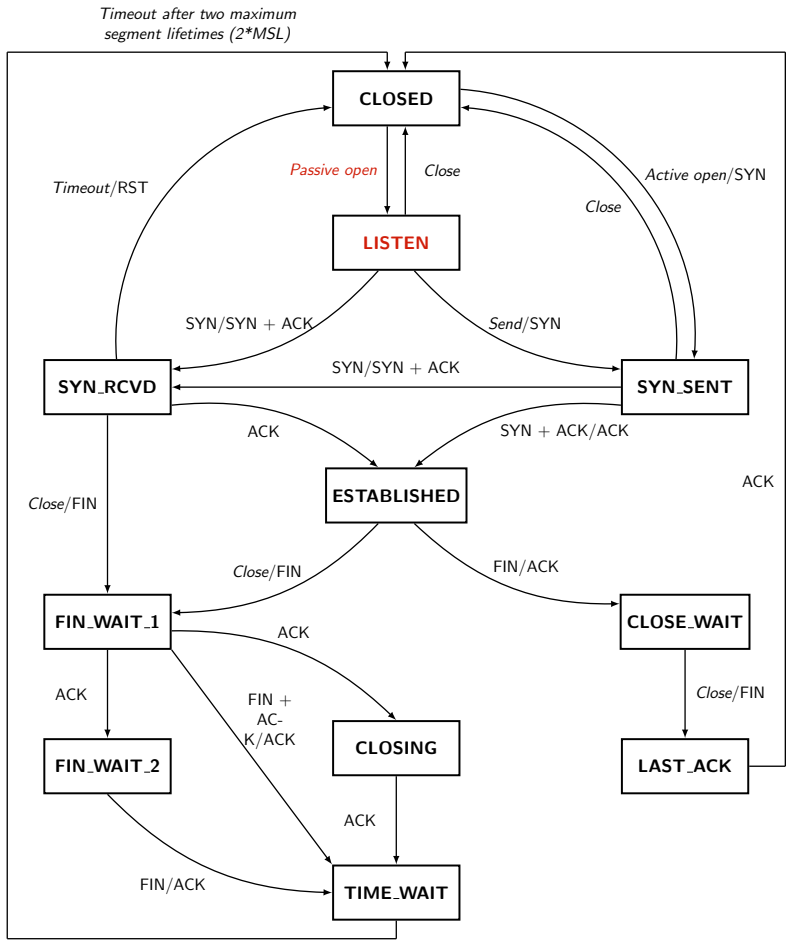


TCP state diagram

Client

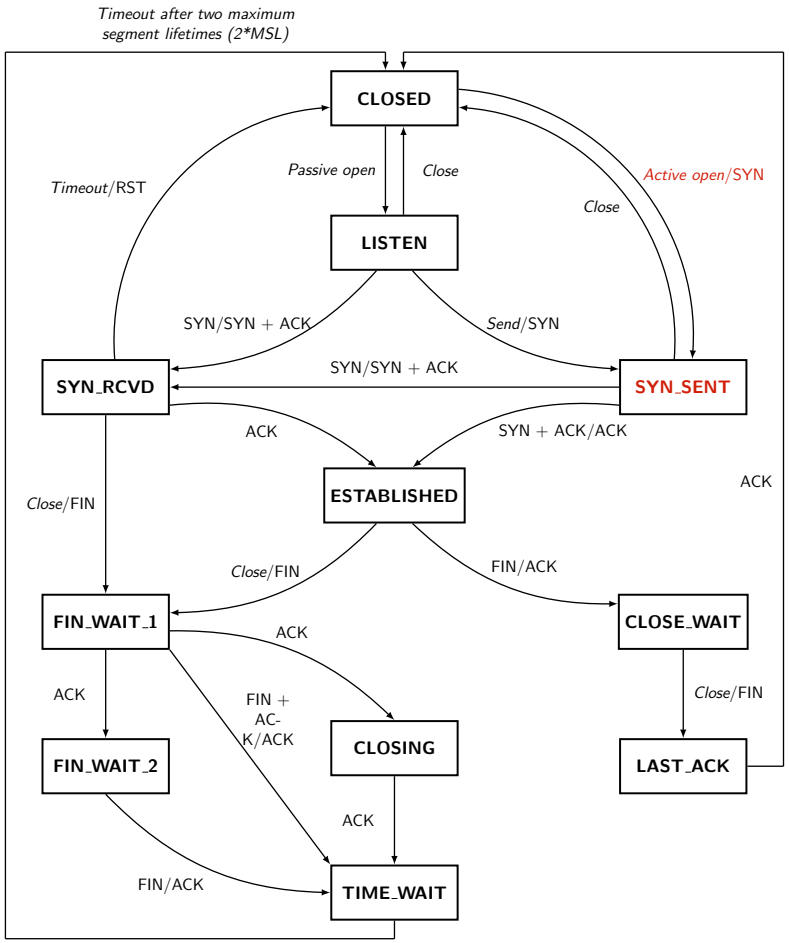


Server

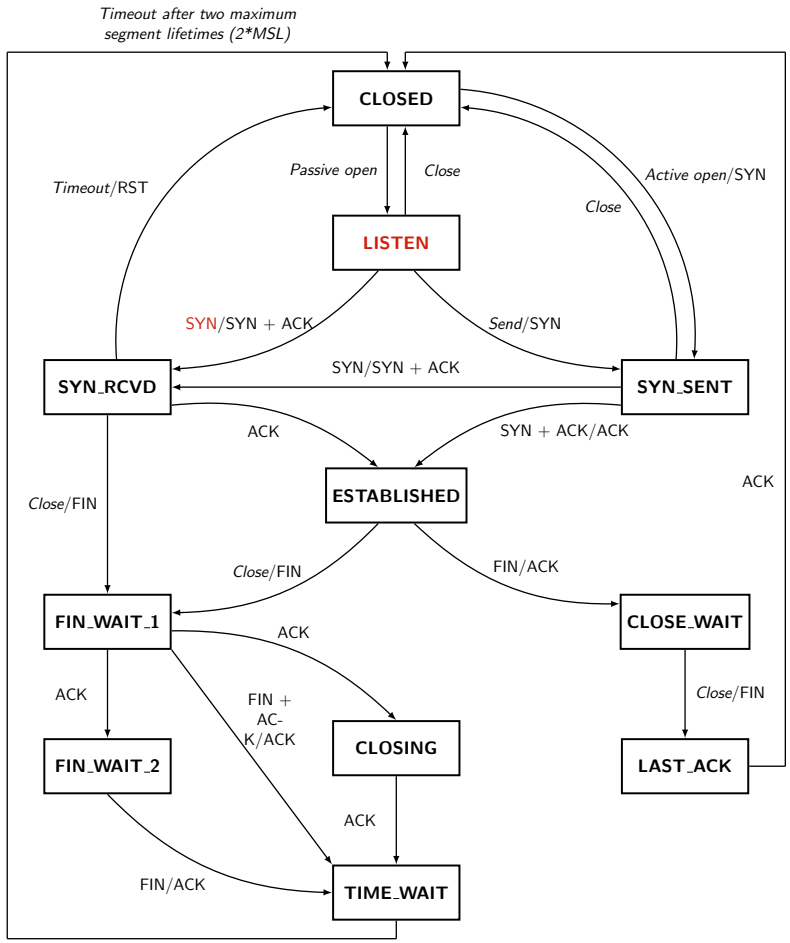


TCP state diagram

Client

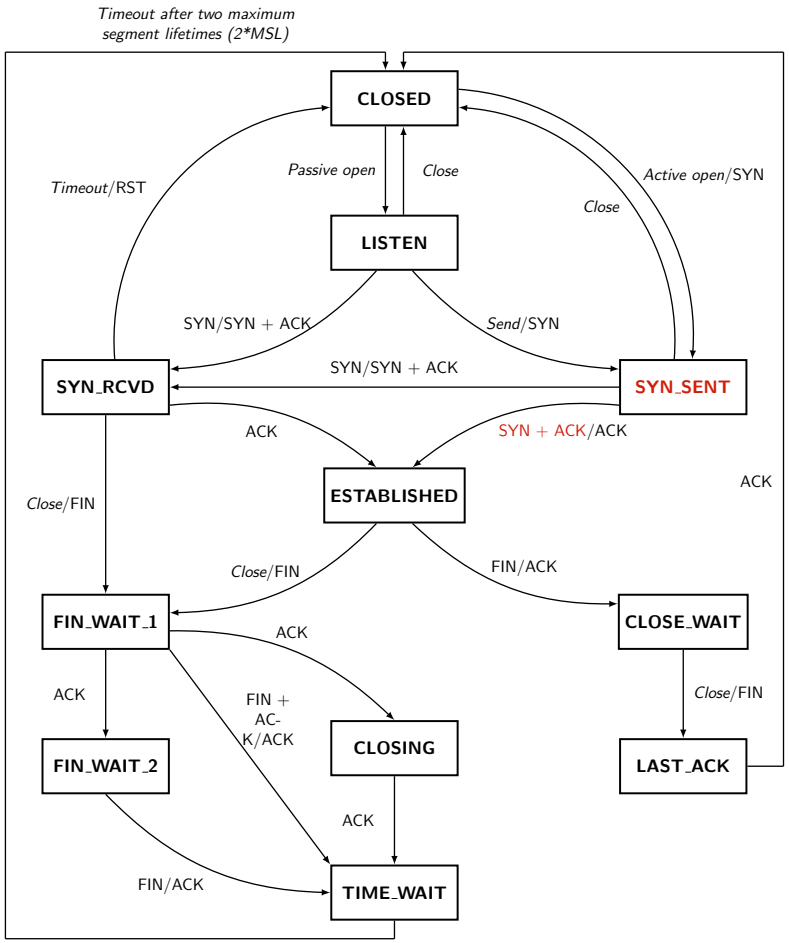


Server

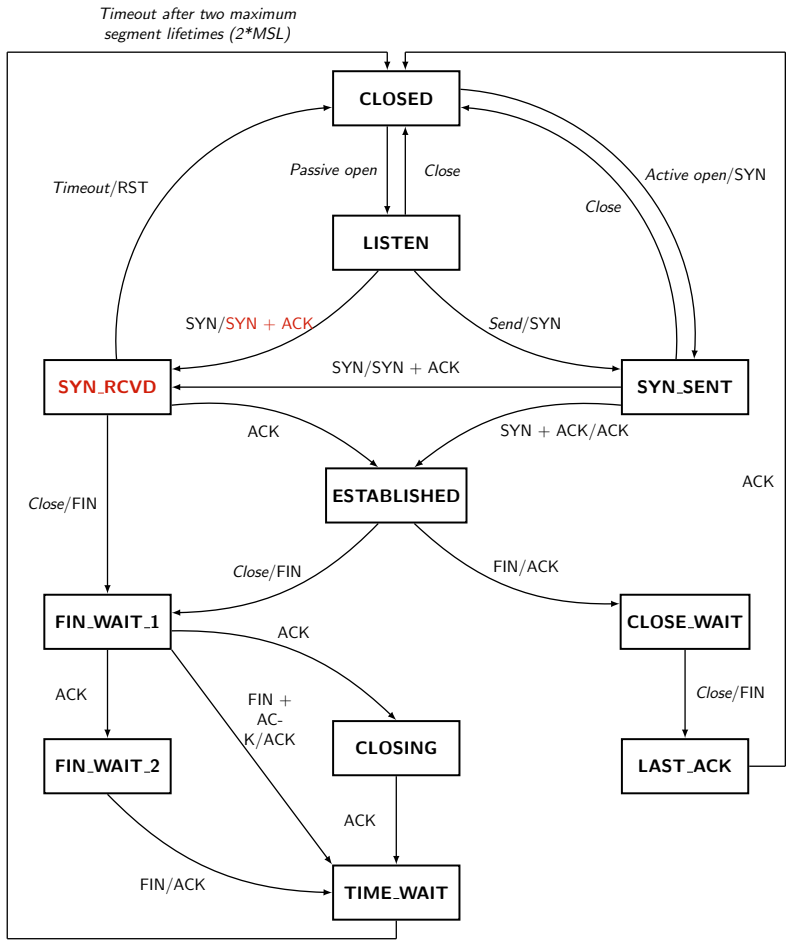


TCP state diagram

Client

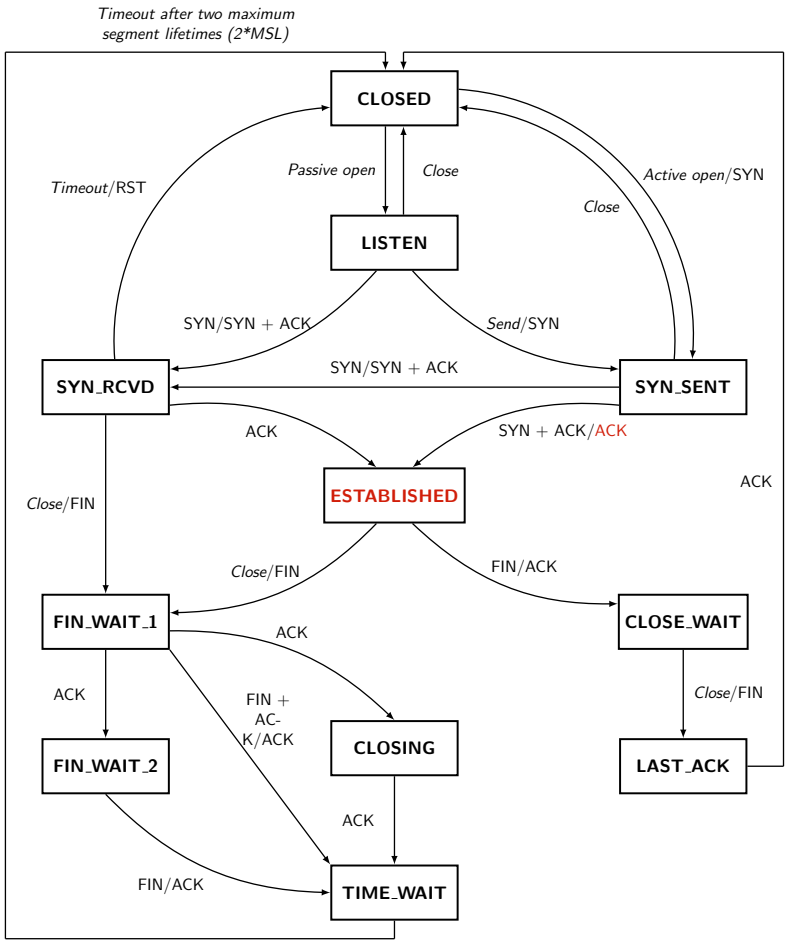


Server

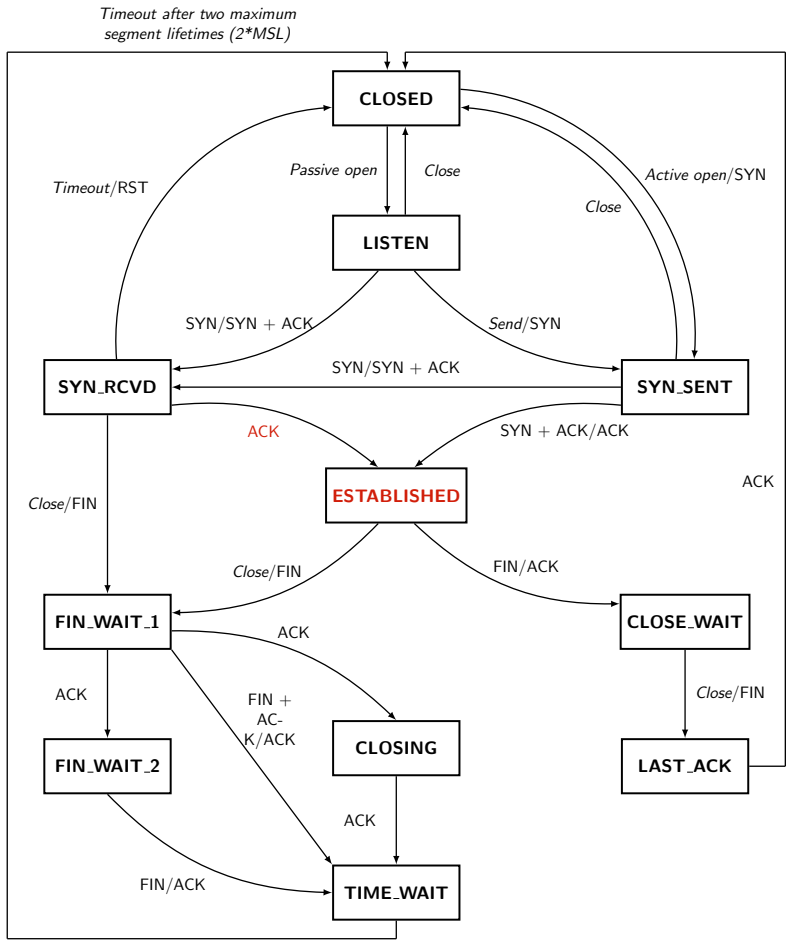


TCP state diagram

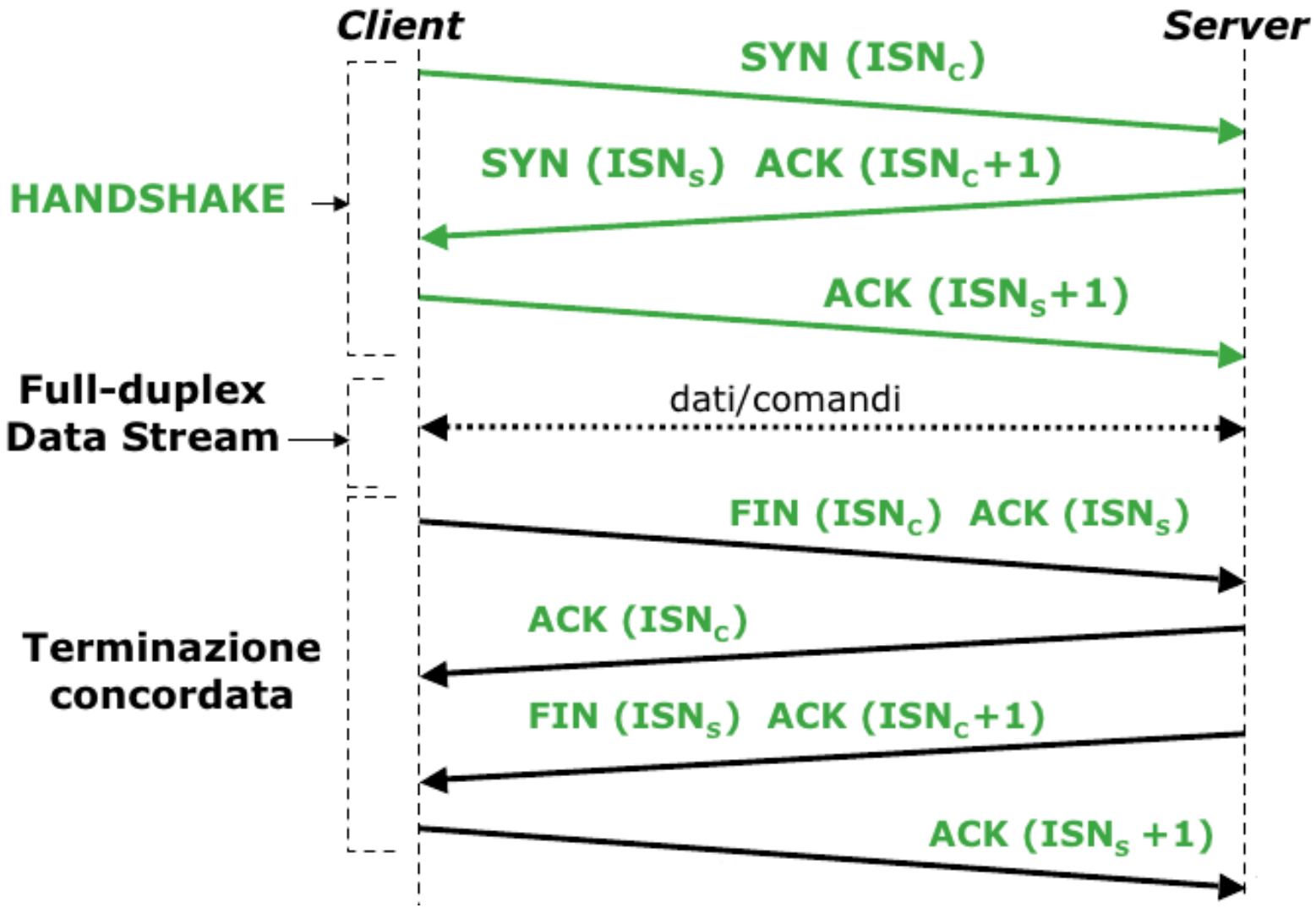
Client



Server



Sequence diagram

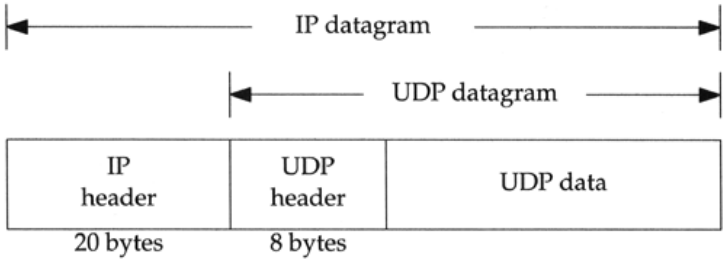


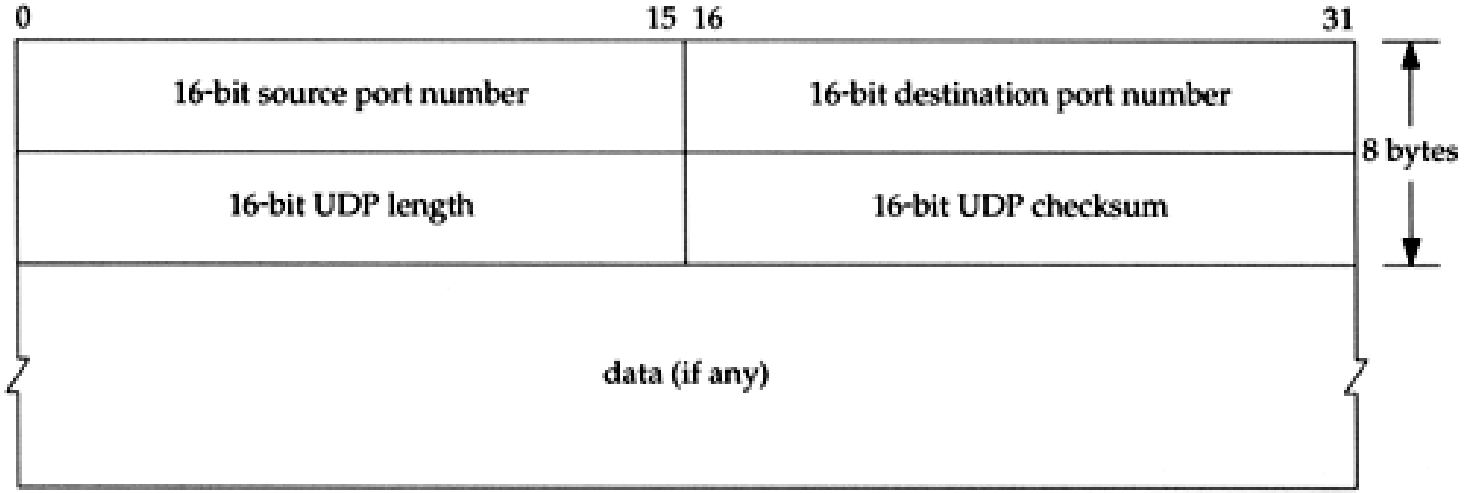
UDP

User Datagram Protocol

- Protocollo di trasporto “minimo”, senza connessione, senza stato
- minimo overhead (TCP: +20B, UDP: +8B)

UDP segment





Checksum

Sia TCP che UDP portano nei segmenti un checksum.
Attenzione: ha lo scopo di proteggere solo dagli errori di trasmissione, non dalle alterazioni maligne!

Riassumendo

- TCP: connessione tramite 3-way handshake, stato mantenuto dai nodi
- UDP: minimo overhead rispetto a IP, nessuno stato
- Protocolli senza particolari caratteristiche di sicurezza (confidenzialità o integrità)

Problemi intrinseci in TCP/IP

- Non c'è autenticazione fra le parti
- I controlli d'integrità sono banali
- Si difende la disponibilità della rete dalla congestione, ma non la possibilità di connettersi ad un determinato nodo

IP spoofing

Il campo `SRC` dello header IP è falsificabile senza particolari difficoltà.

- Le autenticazioni basate su indirizzi IP sono insicure, soprattutto all'interno di una rete locale.
- Fra l'altro la presenza di numeri IP duplicati può causare *denial of service*

Dal punto di vista dell'attaccante

Se l'IP sorgente è falso

- le risposte andranno al vero nodo titolare
- "spoofare" l'IP non è sufficiente per inserirsi in una connessione TCP

Spoofing in connessioni TCP

Per una connessione serve lo handshaking

1. $C \rightarrow S : SYN, ISN_C$
2. $S \rightarrow C : SYN, ISN_S, ACK(ISN_C)$
3. $C \rightarrow S : ACK(ISN_S)$

Se ISN_S è imprevedibile è difficile (2^{-32}) per X farsi passare per C (e se C è *up*, manderà un RST).

Initial sequence number

RFC793: ISN va incrementato circa 1 volta ogni 4 microsecondi per evitare confusioni con connessioni duplicate. Alcune implementazioni ancora piú prevedibili (famose le kick-off war con IRC e stack vulnerabili come quelli di alcune versioni di Windows)

ISN sicuri

Non può essere completamente casuale. RFC1948 (ora RFC6528) propone:

$$ISN = M + F_S(localhost, localport, remotehost, remoteport)$$

con F_S funzione hash crittografica, non prevedibile da un attaccante e M un contatore incrementato ogni 4 microsecondi.

Riassumendo

- I campi dei pacchetti sono facilmente falsificabili
- Il numero di sequenza è un parametro particolarmente delicato

Frammentazione

I segmenti TCP sono spesso frammentati e riassemblati dal destinatario.
Un *man-in-the-middle* può alterare i frammenti: in questo caso non serve indovinare i sequence number. I checksum sono facili da *aggiustare* perché semplici controlli d'errore di trasmissione.

SYN flooding

Quando un host S riceve una richiesta SYN, tiene traccia per un certo tempo (spesso 75s) della connessione in una coda.

- La coda ha lunghezza finita: talvolta addirittura 5
- SYN cui non segue un ACK possono portare a DoS

I SYN-cookie (D. J. Bernstein) usano gli ISN per evitare il flooding.

Fingerprinting

Dall'esame (non intrusivo) dei pacchetti di rete è possibile identificare molti dettagli utili negli attacchi...

- p.es. `p0f` è in grado di riconoscere molte implementazioni di stack TCP/IP

Topologia della rete

È possibile studiare la topologia della rete esaminando il TTL

- p.es. Windows TTL=128, Linux TTL=64
- TTL==80 $\Rightarrow$ Windows, e il nodo è distante 48 hop

Letture obbligatorie

- Steven M. Bellovin. *A Look Back at "Security Problems in the TCP/IP Protocol Suite".* In Proceedings of the 20th Annual Computer Security Applications Conference (ACSAC '04). 229-249.
- Steven M. Bellovin, *Defending Against Sequence Number Attacks*, February 2012, RFC6528

Riassumendo

- Il controllo d'integrità è lasco
- Il DoS può essere ottenuto abbastanza facilmente
- Gli header dei pacchetti rivelano molte informazioni ai potenziali attaccanti

Lezione V: Scansioni

Attività di ricognizione

Per progettare difese o attacchi occorre partire da attività di ricognizione delle reti obiettivo.
Il difensore *dovrebbe* conoscere la "Cartografia di reti e servizi", ma non sempre è cosí...
L'attaccante:

- Social engineering, WHOIS, DNS, Google
- Scanning

Socket programming

Server

```
1  int sd, sd_current;
2          socklen_t size;
3  struct sockaddr_in sin, pin;
4
5  if ((sd = socket(AF_INET, SOCK_STREAM, 0)) /* TCP */
6                  == -1) {perror("socket");exit(1);}
7
8  memset(&sin, 0, sizeof(sin));
9  sin.sin_family = AF_INET;
10 sin.sin_addr.s_addr = INADDR_ANY;
11 sin.sin_port = htons(PORT);
12
13 if (bind(sd, (struct sockaddr *) &sin, sizeof(sin))
14                 == -1){perror("bind");exit(1);}
15
16 if (listen(sd, 5)
17                 == -1) {perror("listen");exit(1);}
18
19 if (sd_current =
20                 accept(sd, (struct sockaddr *) &pin, &size)
21                 == -1) {perror("accept");exit(1);};
22
23 /* send/recv */
24
25 close(sd_current); close(sd);
```

Client

```
1
2  int sd;
3  struct sockaddr_in sin, pin;
4
5  memset(&pin, 0, sizeof(pin));
6  pin.sin_family = AF_INET;
7  if (inet_aton(argv[1], &pin.sin_addr)
8      == 0) {perror("inet_aton");exit(1);};
9  pin.sin_port = htons(PORT);
10
11 if ((sd = socket(AF_INET, SOCK_STREAM, 0))
12     == -1) {perror("socket");exit(1);}
13
14 if (connect(sd,(struct sockaddr *) &pin,
15                 sizeof(pin)) == -1) {perror("connect");exit(1);}
16
17 /* send/recv */
18
   close(sd);
```

Ping

- ICMP: protocollo per scambiare messaggi di controllo e diagnostici. ping manda pacchetti ICMP che chiedono una risposta.
- Esistono programmi per ping (non solo ICMP) massivi (`hping`, `fping`, `nmap`).
- Spesso ICMP viene filtrato per evitare questo tipo di attività.

Traceroute

- traceroute usa i TTL per analizzare una rete.
- Inizia mandando un pacchetto (ICMP o UDP) con `TTL==1` e si aspetta una risposta ICMP `TTL exceeded`: il mittente sarà un router a distanza 1 hop.
- Si ripete con TTL crescenti finché non si riceve un reply dalla destinazione finale.

Port scanning

La conoscenza di quali *porte* sono accessibili (TCP o UDP) identifica i possibili canali di comunicazione:

- quali applicazioni monitorare
- quali canali sono utilizzabili in un attacco

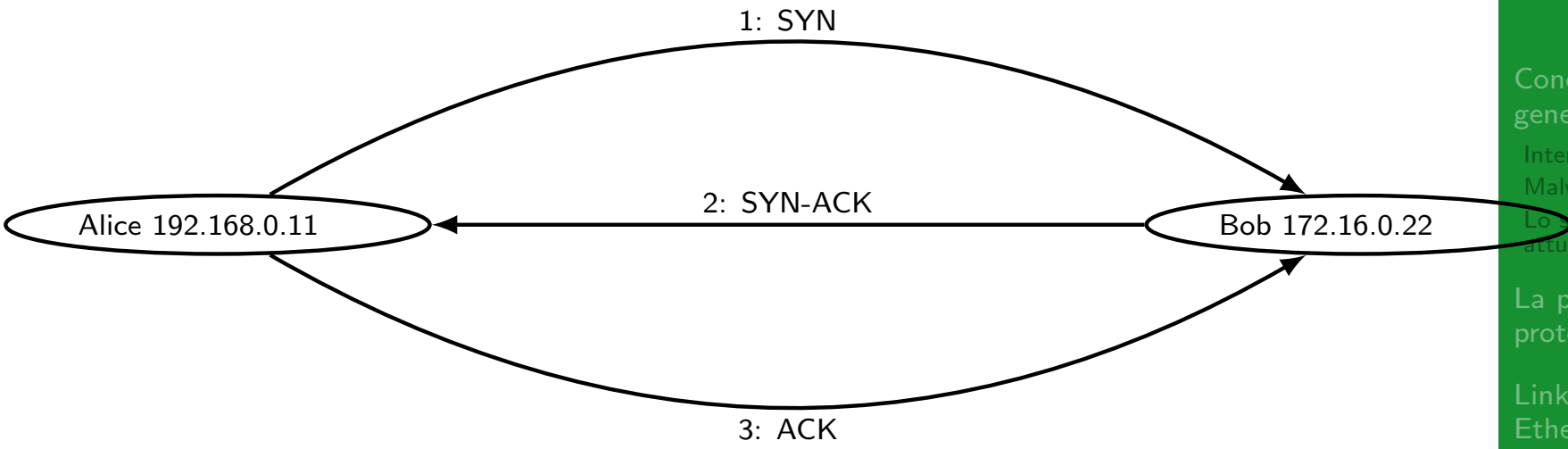
Stato di una porta

open Possibilità di connessione con un'applicazione (non necessariamente quella standard!)

closed Accessibile, ma non c'è nessuna applicazione in ascolto

filtered Appare *closed* ($\neg$ *open*) per filtraggio (del router, firewall, ecc.)

Nel caso di TCP



- Un SYN a porta chiusa $\rightarrow$ RST
- Un SYN-ACK $\rightarrow$ RST
- Un RST viene ignorato

UDP scan

UDP privo di *handshake*: un po' piú complicato

- lo stato è segnalato tramite ICMP
- lento, e sostanzialmente basato su timeout
- non molto affidabile, perché spesso ICMP è filtrato (p.es., solo x al minuto)

NMap

Nmap Scan	Command Syntax	Requires Privileged Access	Identifies TCP Ports	Identifies UDP Ports
TCP SYN Scan	`-sS`	YES	YES	NO
TCP connect() Scan	`-sT`	NO	YES	NO
FIN Scan	`-sF`	YES	YES	NO
Xmas Tree Scan	`-sX`	YES	YES	NO
Null Scan	`-sN`	YES	YES	NO
Ping Scan	`-sP`	NO	NO	NO
Version Detection	`-sV`	NO	NO	NO
UDP Scan	`-sU`	YES	NO	YES
IP Protocol Scan	`-sO`	YES	NO	NO
ACK Scan	`-sA`	YES	YES	NO
Window Scan	`-sW`	YES	YES	NO
RPC Scan	`-sR`	NO	NO	NO
List Scan	`-sL`	NO	NO	NO
Idlescan	`-sI`	YES	YES	NO
FTP Bounce Attack	`-b`	NO	YES	NO

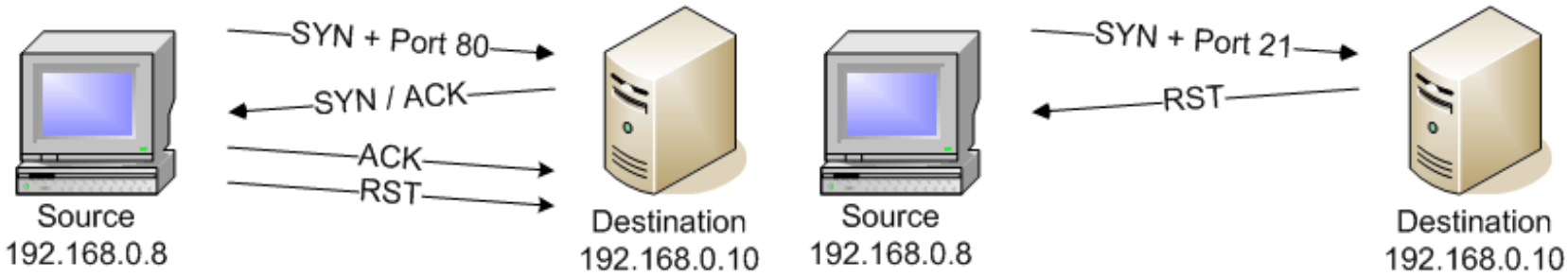
Riassumendo

- La conoscenza dei nodi e dei canali di comunicazione disponibili è fondamentale per attaccanti e difensori
- Documentazione, social engineering, WHOIS, DNS, Google...
- Scanning. Con Zmap (2013) è possibile esaminare l'intero spazio IPv4 in meno di un'ora (singola porta).

Connect

La modalità piú semplice è tentare una connessione (`connect()`)

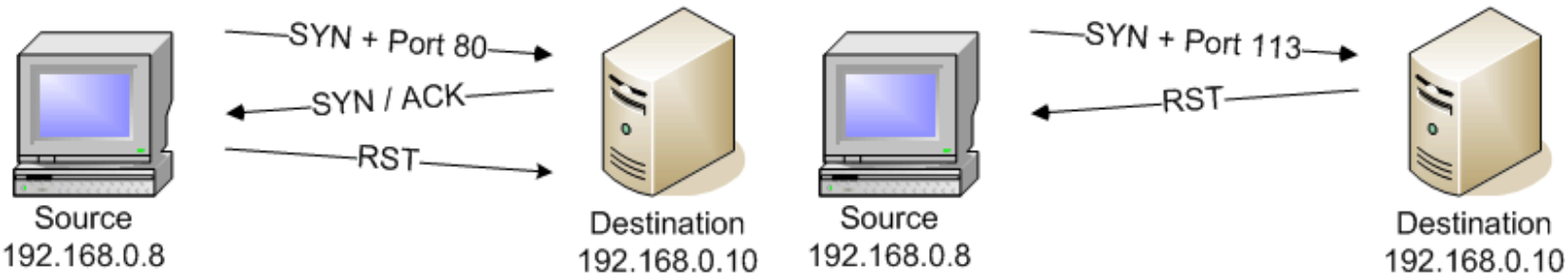
- non richiede privilegi particolari
- molto spesso l'evento viene registrato (e se la connessione avviene con lo stack standard il numero IP è quello reale)



SYN scan (half open)

Si risponde al SYN-ACK con un RST.

- È il metodo piú usato: veloce ed efficace
- Richiede i privilegi di root (non si può usare lo stack TCP standard)
- Piú difficile da "loggare"



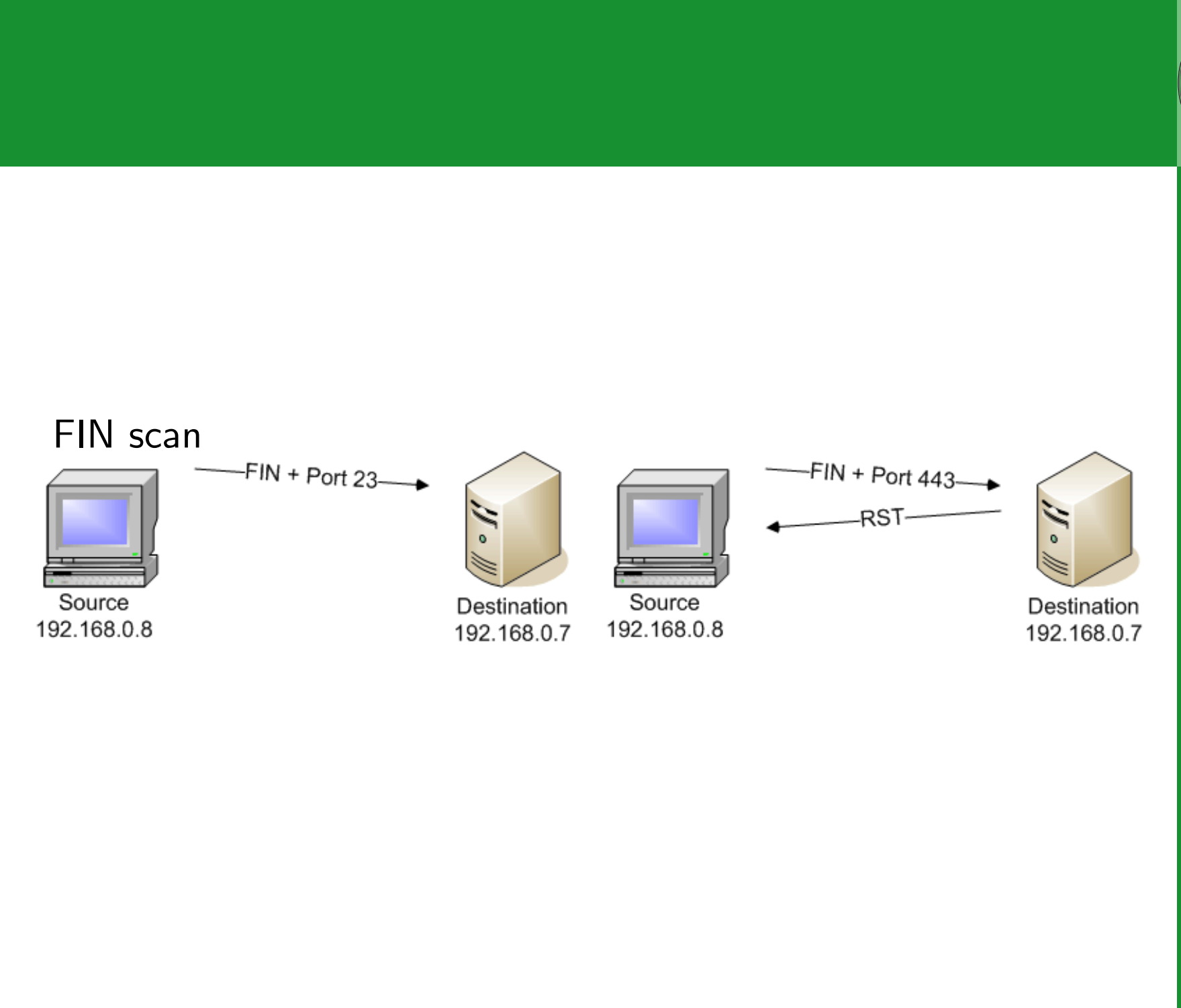
TCP NULL, FIN, Xmas scan

Si usano i flag in modo “creativo”: invece di SYN, tutti gli altri in varie combinazioni; una porta chiusa risponde con un RST, una aperta invece li scarta (aspetta solo i SYN).

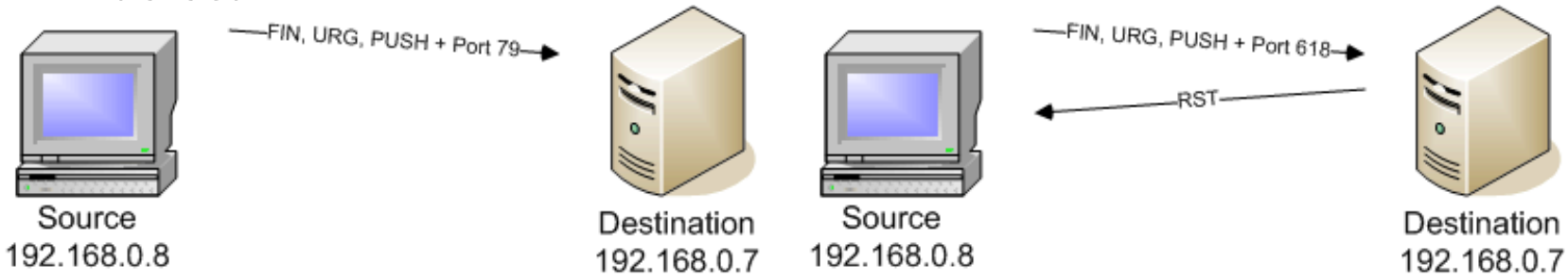
- Analoghi al SYN
- richiedono i privilegi di root
- ma ancora meno probabile una registrazione dell’evento

TCP NULL, FIN, Xmas scan

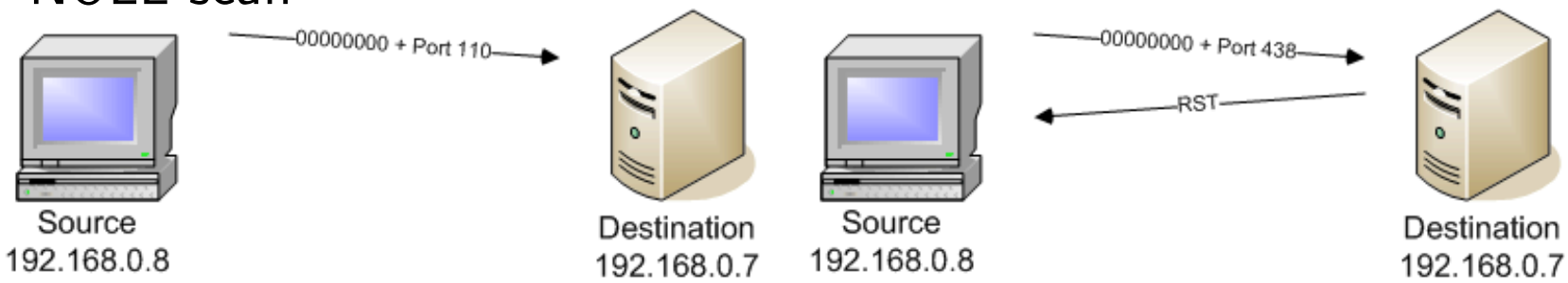
Attenzione però, se lo stack destinazione non è esattamente RFC 793 compliant, potrebbe agire in modo anomalo facendo apparire tutto chiuso


FIN scan
FIN + Port 23
Source
192.168.0.8
Destination
192.168.0.7
FIN + Port 443
RST
Source
192.168.0.8
Destination
192.168.0.7

Xmas scan



NULL scan



Maimon scan: FIN-ACK; È possibile provare differenti combinazioni dei flag.

ACK scan, Window scan

- Serve a determinare se c'è filtraggio.
- ACK: se non c'è filtraggio open e closed $\rightarrow$ RST
- se non c'è risposta o ICMP: filtered
- Window sfrutta la window size del RST ricevuto per distinguere fra open e closed (diversa in alcune implementazioni)

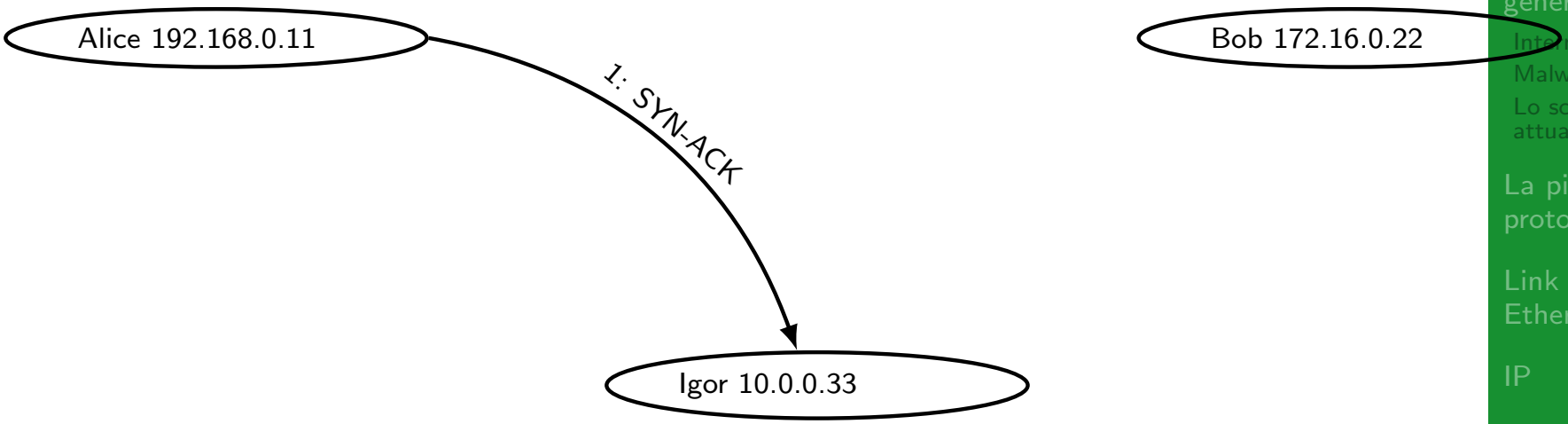
Idle scan

Lo scan viene compiuto da un nodo inconsapevole sfruttando il meccanismo di generazione degli ISN pacchetti , che talvolta è banalmente sequenziale.

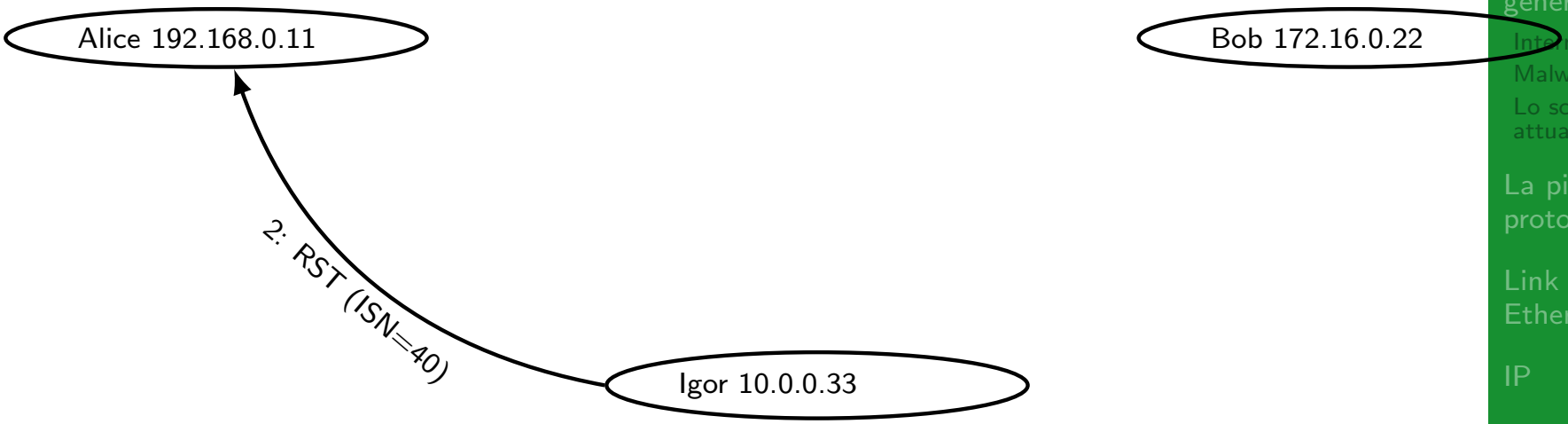
Idle scan

- Un log conterrà l'IP della macchina "prestanome" (non è *spoofing* perché il nodo esiste e ha operato nel modo registrato)
- Il nodo deve essere idle, cioè non produrre traffico di rete suo durante lo scan
- Lo stack TCP deve incrementare banalmente gli ISN

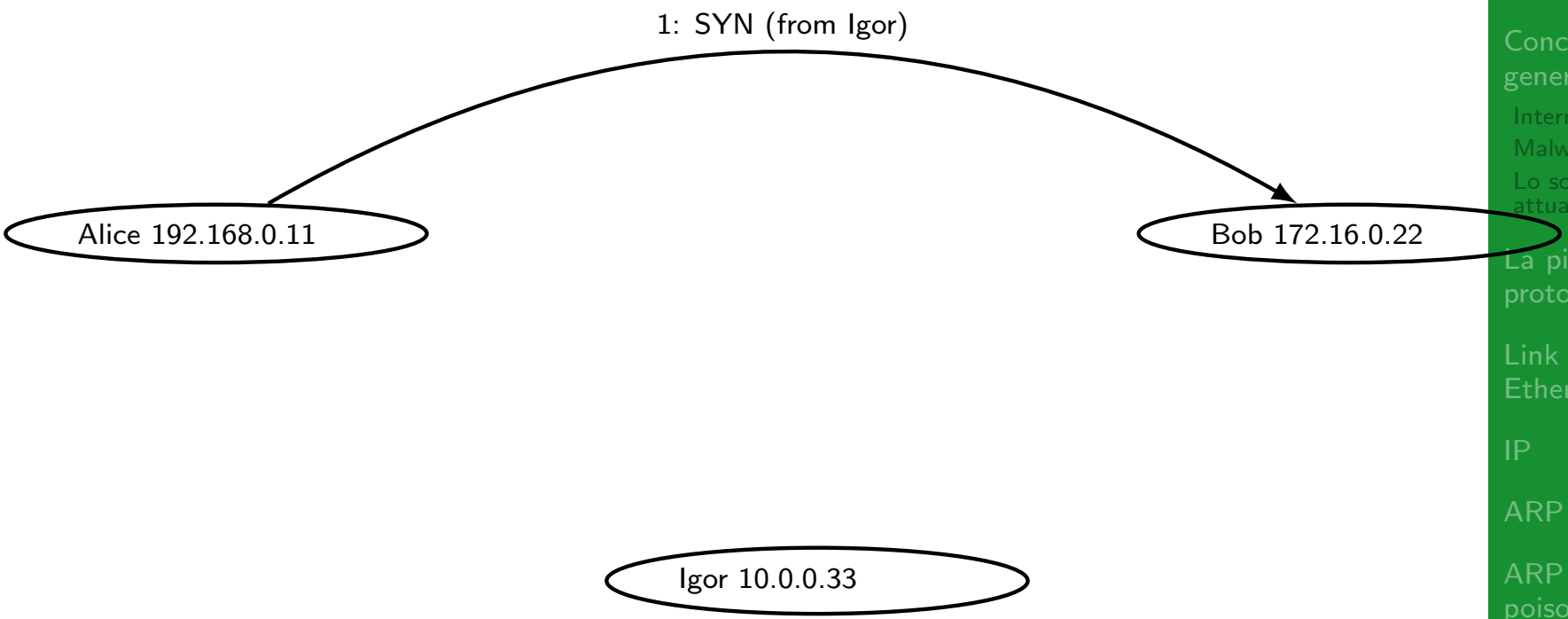
Idle scan con porta aperta



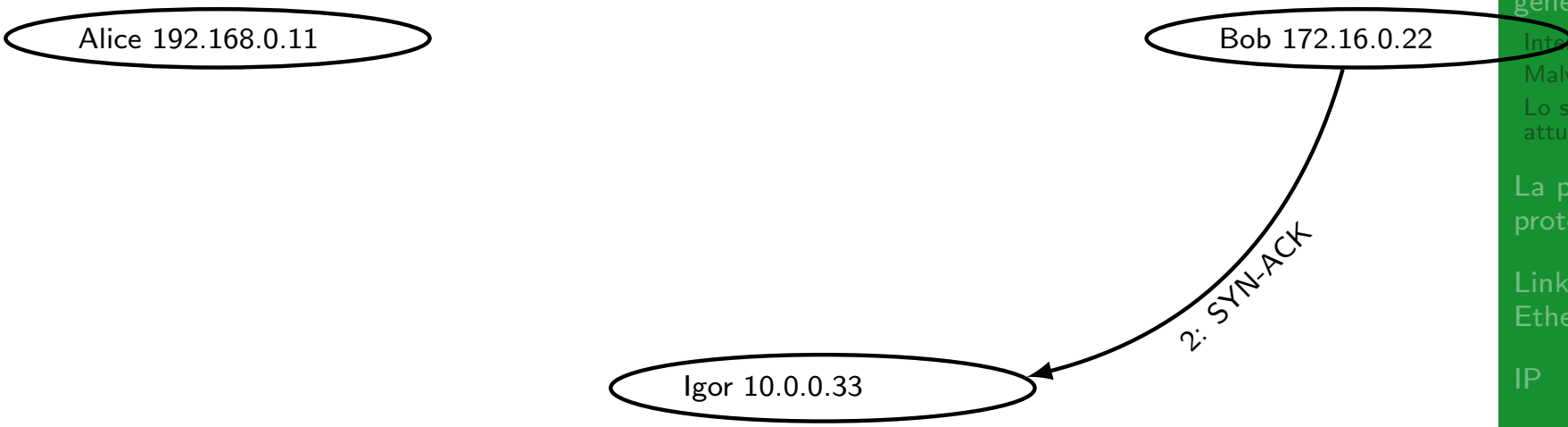
Idle scan con porta aperta



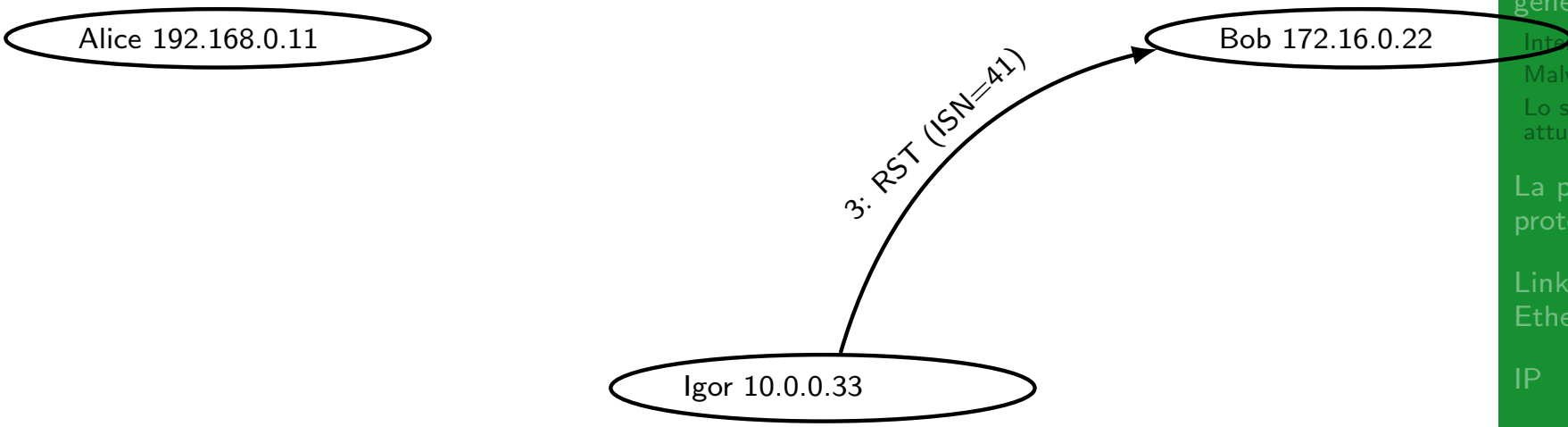
Idle scan con porta aperta



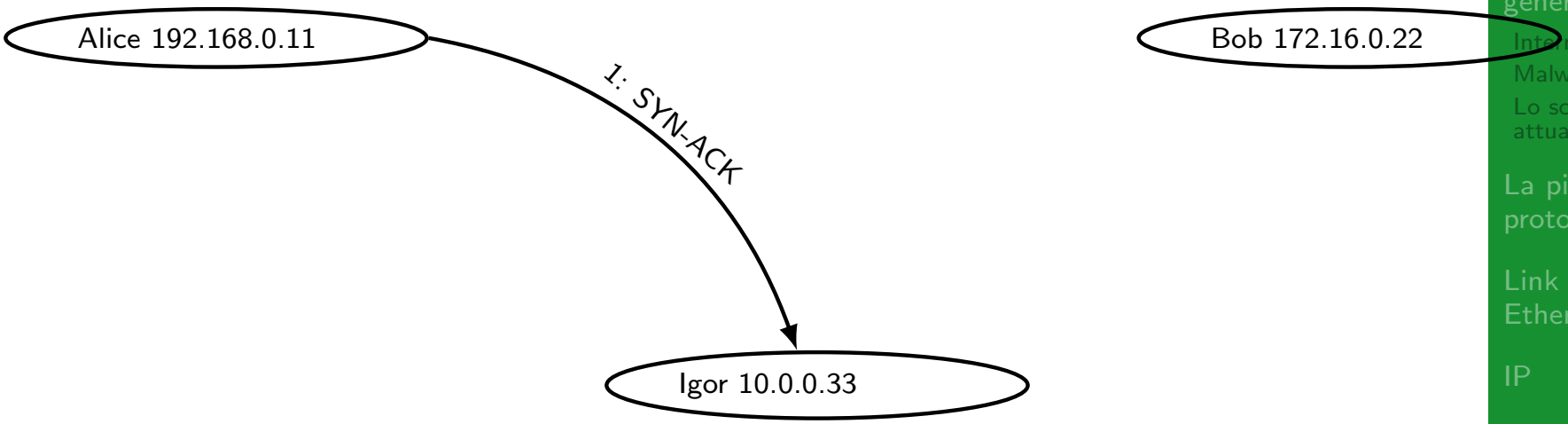
Idle scan con porta aperta



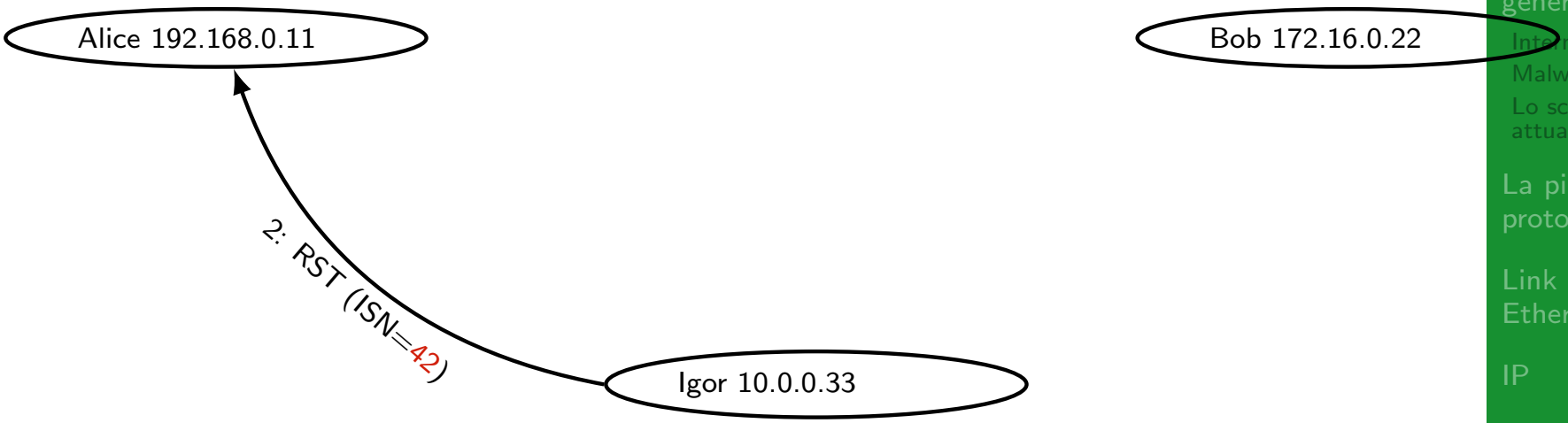
Idle scan con porta aperta



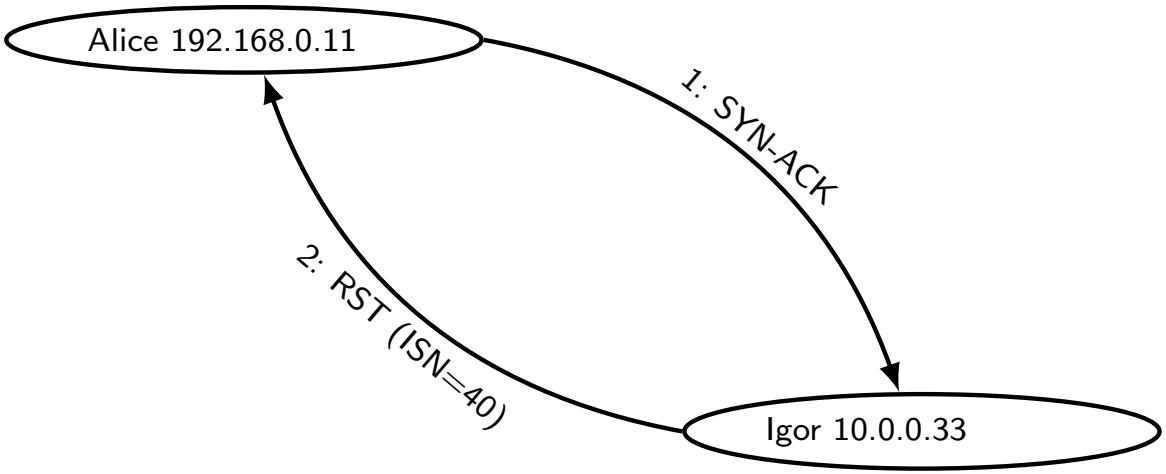
Idle scan con porta aperta



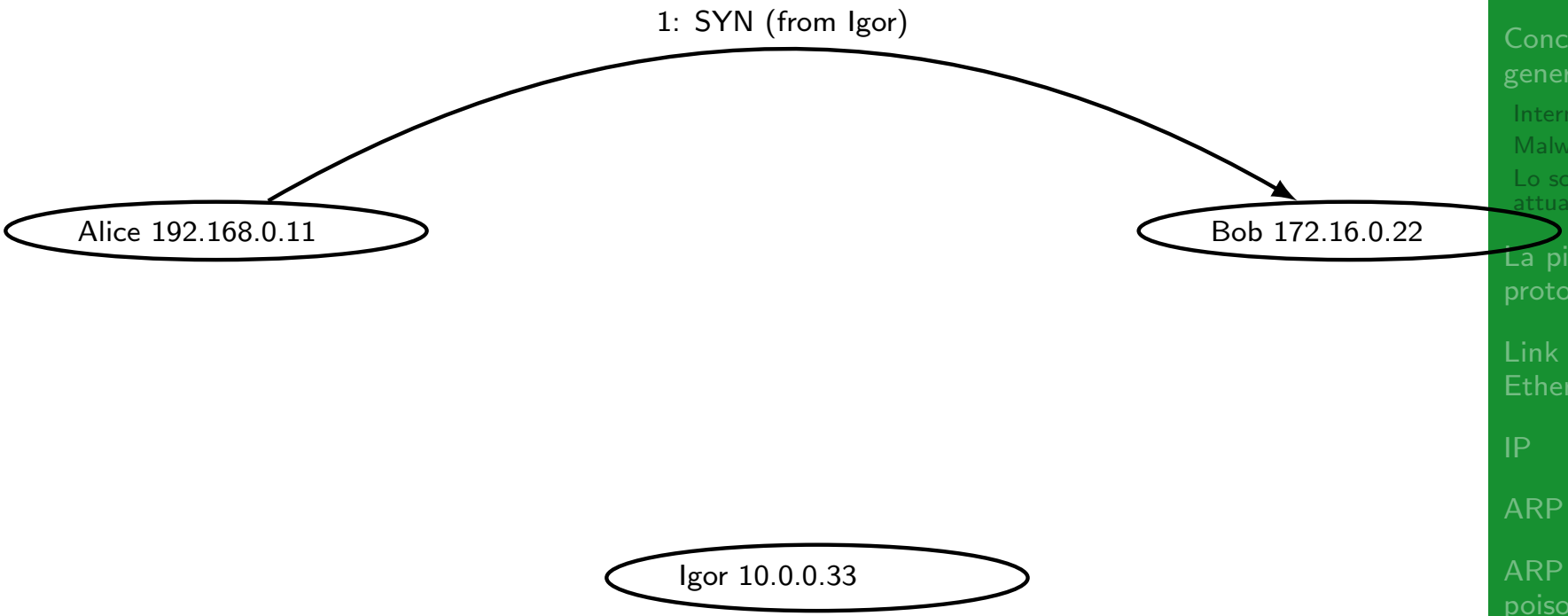
Idle scan con porta aperta



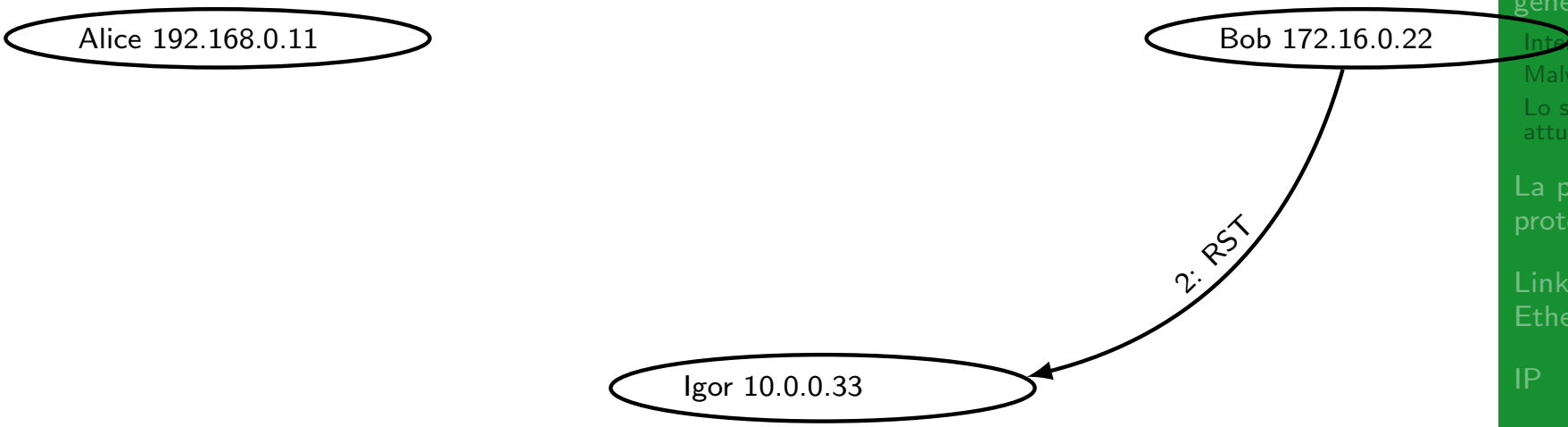
Idle scan con porta chiusa



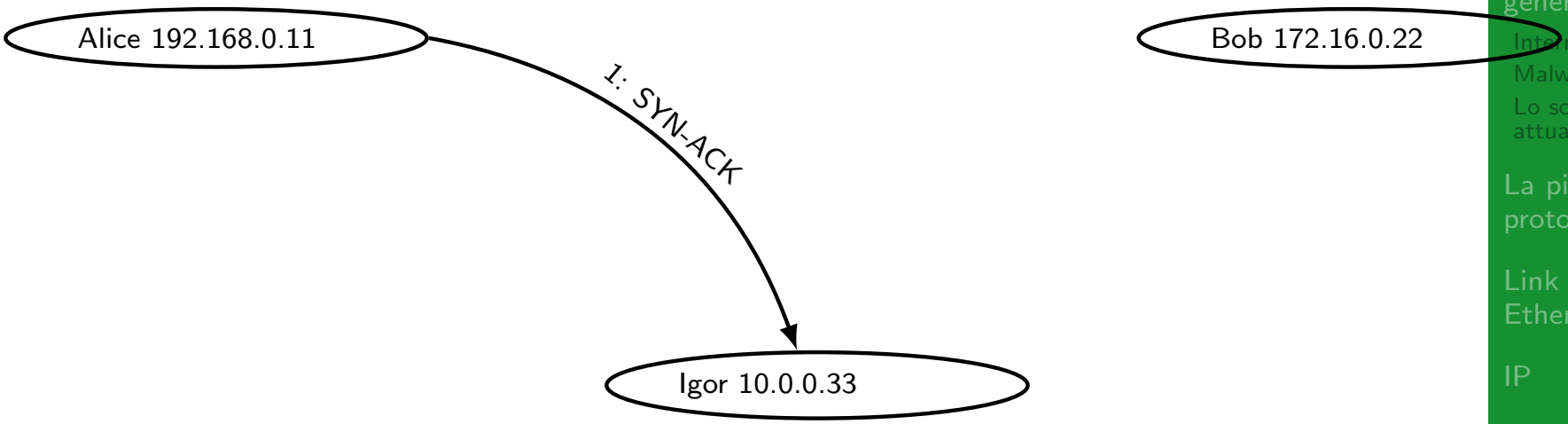
Idle scan con porta chiusa



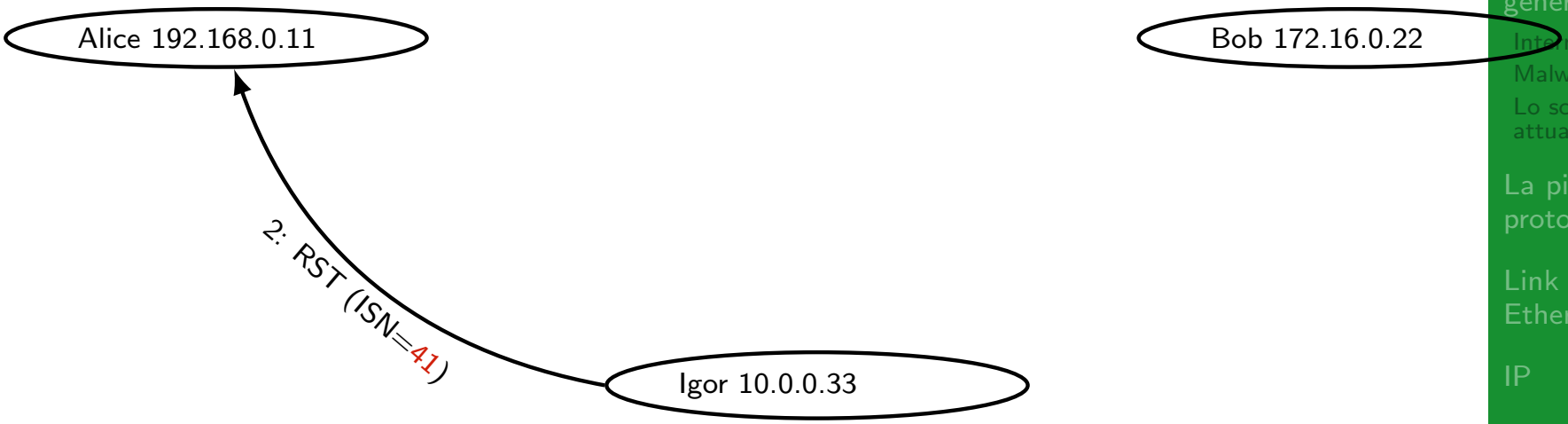
Idle scan con porta chiusa



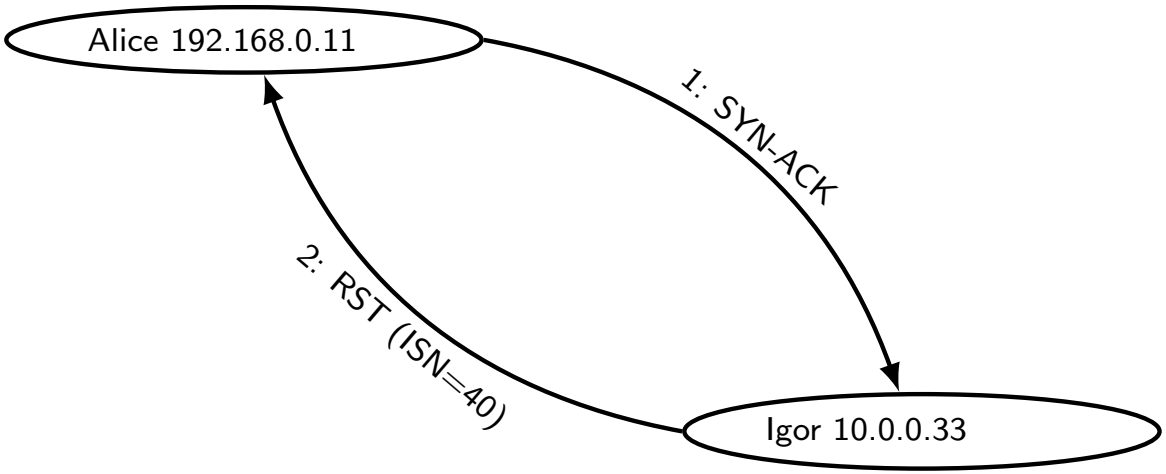
Idle scan con porta chiusa



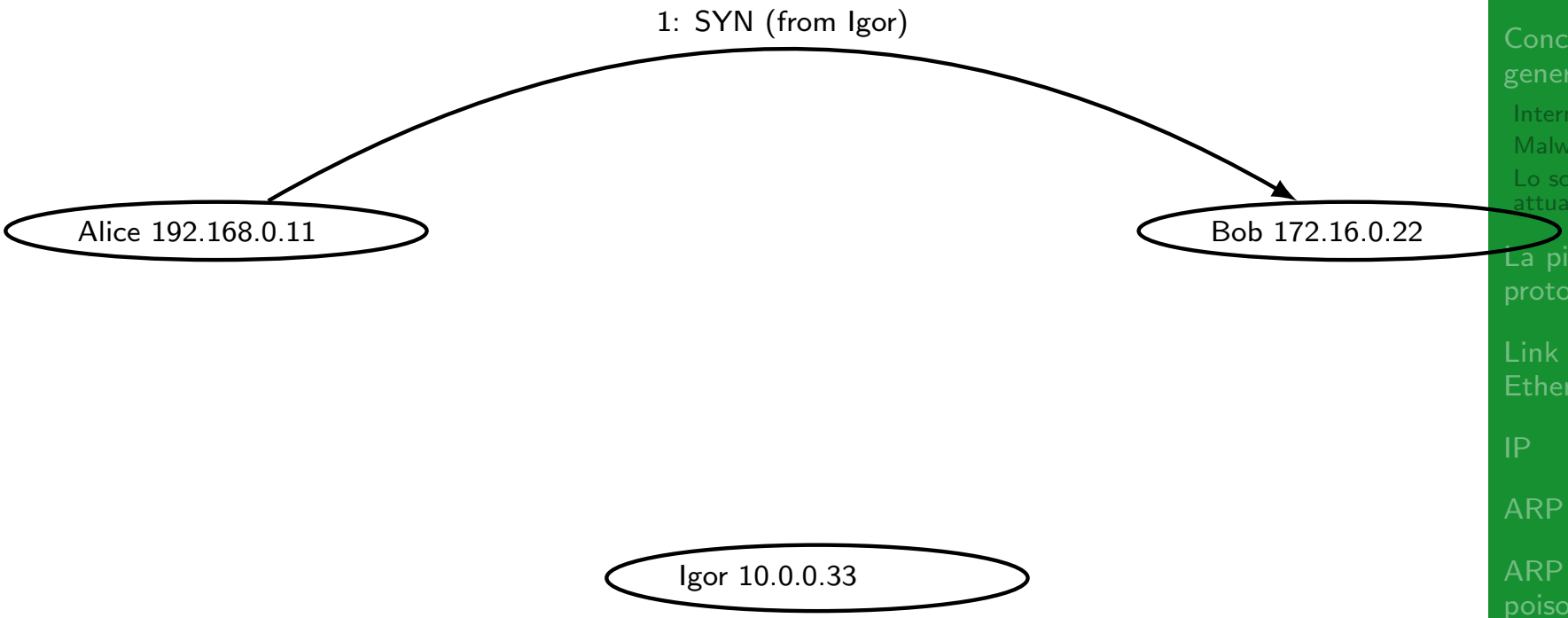
Idle scan con porta chiusa



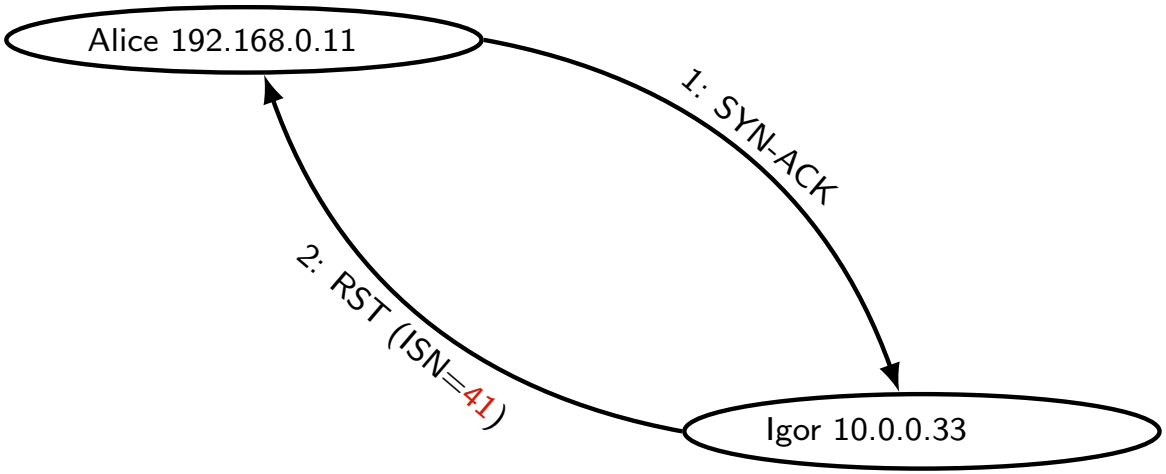
Idle scan con porta filtrata



Idle scan con porta filtrata



Idle scan con porta filtrata



Riassumendo

- Sono note diverse tecniche per rilevare se una porta TCP è aperta
  - Semplice connessione
  - Pacchetti creati ad hoc
  - Idle scan

Lezione VI: IPsec

La suite TCP/IP

La suite TCP/IP non è progettata con particolari misure di difesa per la confidenzialità o integrità dei dati dalle manomissioni.

- Lo scenario di riferimento: nodi per lo piú cooperativi (accademici)
- e qualcuno sostiene che NSA fu contraria all'inserimento di tecniche crittografiche in una rete pubblica

IPsec

IPsec specifica come crittare, autenticare e scambiare chiavi con IP.

- Basato su IP (in maniera differente IPv4 e IPv6)
- Obbligatorio supportarlo per gli stack IPv6, facoltativo in IPv4

Servizi di sicurezza offerti

- Controllo dell'accesso alla comunicazione
- Autenticazione dell'origine dei dati
- Integrità dei dati
- Confidenzialità dei dati
- Protezione da *replay*

Protocolli IPsec

Si tratta in realtà di piú specifiche protocollari

- Authentication Header (AH) per l'autenticazione e integrità del datagramma
- Encapsulating Security Payload (ESP) per la confidenzialità

Entrambi presuppongono una Security Association (SA), per lo scambio di credenziali.

AH

- Serve per autenticare l'origine del pacchetto e l'integrità dei campi immutabili.
- Un security parameter index identifica la SA
- Identifica replay di pacchetti con una tecnica "sliding window" e un contatore che per essere inizializzato necessita una nuova SA

Sliding window

Il nodo destinazione tiene un array di $SW[1 : w] = 0$ elementi per ogni SA

1. Primo datagramma contatore n: $SW[w] = n$
2. Datagramma contatore i

   $n - w + 1 \leq i \leq n \wedge OK(sig)$ controlla se $SW[i + w - n] > 0$ (replay!), altrimenti $SW[i + w - n] = i$

   $i \leq n - w$ vecchio

   $i > n \wedge OK(sig)$ sposta la finestra

ESP

- Serve per crittare il contenuto dei pacchetti
- Un security parameter index identifica la security association
- Due modalità
  1. transport protocolli superiori vengono crittati end-to-end
  2. tunnel i pacchetti IPsec contengono (crittati) pacchetti IP

Security association

Ogni conversazione IPsec è abbinata ad una Security association (SA) frutto di una negoziazione dei parametri di sicurezza e delle credenziali.

- IP destinazione
- Una SA per AH e una per ESP
- Statiche o dinamiche (ISAKMP: Internet Security Association Key Management Protocol, IKE: Internet Key Exchange)

problemi IPsec

- La configurazione dei firewall per permettere i protocolli IPsec non è banale
- Ogni volta che una comunicazione comporta la manipolazione dei pacchetti IP (proxy e NAT) occorre adottare misure speciali, con successive security association.

Riassumendo

- IPsec introduce autenticazione, integrità e confidenzialità
- Protezione da *replay*
- Necessita di un certo overhead amministrativo e computazionale

TLS/SSL

Un'altra possibilità è introdurre misure di sicurezza sopra il livello di trasporto TCP.

- 1993–1995, Netscape rilascia un Secure Socket Layer SSL (2.0) pensato per proteggere la navigazione web.
- SSL 3.0, standardizzato da IETF come TLS Transport Layer Security

Obiettivi di sicurezza

- cifratura end-to-end
- protezione dell'integrità
- autenticazione del server (il client rimane anonimo)
- efficienza adeguata alle connessioni HTTP, brevi e stateless

sessione TLS

I nodi mantengono lo stato della sessione per gestire la cifratura del traffico.

- TLS handshake protocol
- TLS record layer
- una sessione può gestire piú connessioni per ridurre l'overhead

TLS handshake

1. C richiede la connessione, elencando quali cipher suite (CS) conosce
2. S sceglie CS compatibile e spedisce un digital certificate (DC) firmato da una CA
3. C controlla DC e manda criptata una chiave di sessione (K) random

Integrazione con le applicazioni

Tre strategie:

1. Creare un nuovo servizio (es. SSH2)
2. Aggiungere TLS ad un servizio noto (es. HTTPS)
3. Estendere un servizio noto affinché usi TLS (es. ESMTP)

Riassumendo

- TLS permette cifratura e autenticazione dei server (tramite CA) a livello di trasporto
- La gestione delle sessioni è progettata per essere efficiente in presenza di connessioni ripetute
- Molto diffuso perché facile da integrare nelle applicazioni

Livello di trasporto

IPsec e TLS possono essere piuttosto penalizzanti dal punto di vista delle prestazioni (Dal punto di vista delle performance del server, TLS può arrivare ad essere fino a 82 volte piú lento di una connessione TCP).
tcpcrypt è una proposta recente (2010) piú efficiente (3 volte piú lento di TCP)

Altri limiti di TLS

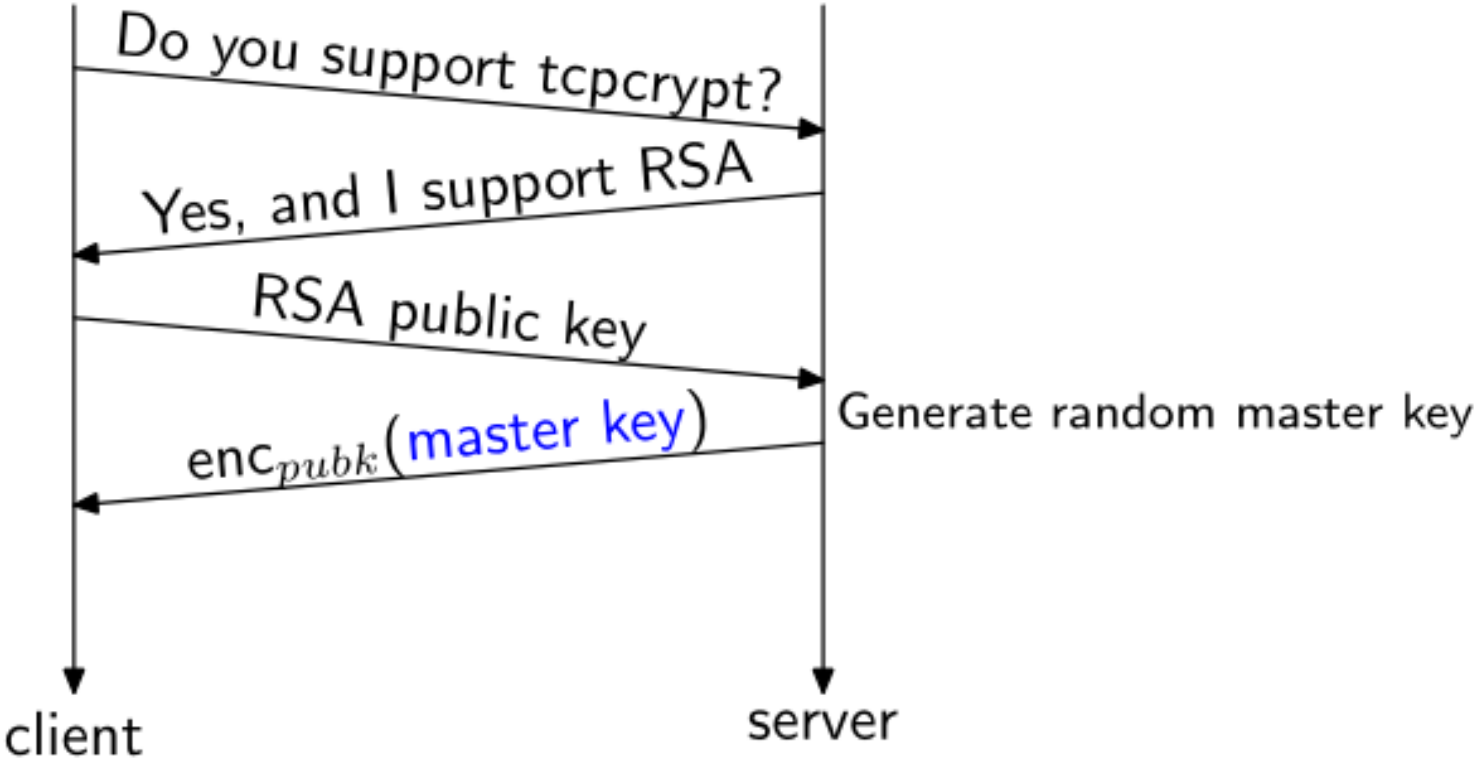
La cifratura dipende dall'autenticazione del server, a sua volta garantita dall'autorità certificatrice.

- Se l'autenticazione è falsa, la cifratura non è molto utile (ma l'overhead rimane)

tcpcrypt

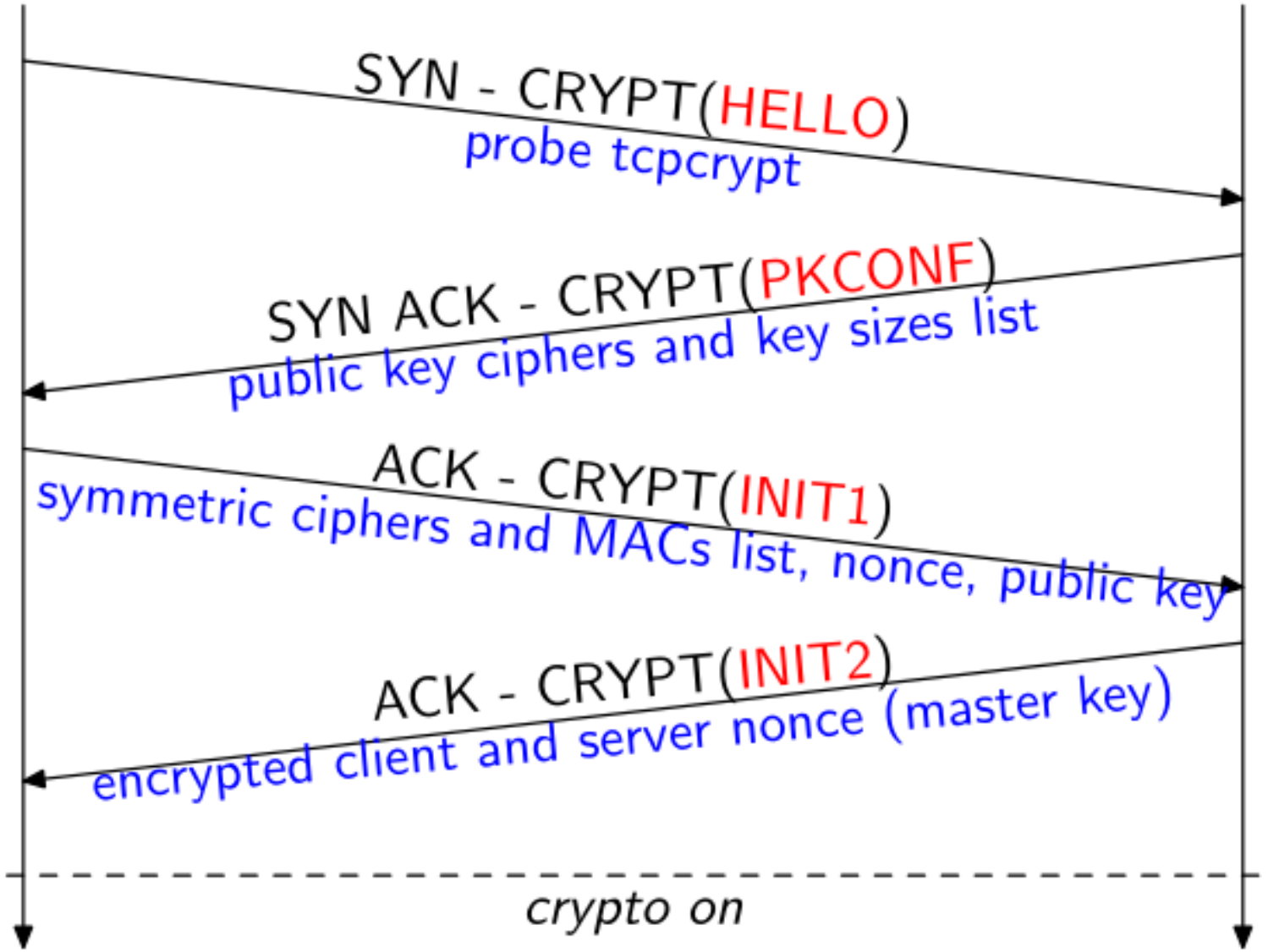
- Estensione di TCP
- Il carico computazionale crittografico è per lo piú spostato sui client
- Opportunistic encryption: attiva solo se supportata da entrambi (attenzione agli attacchi attivi!)

tcpcrypt handshake



36 volte piú veloce di TLS

tcpcrypt handshake



Autenticazione

Non c'è autenticazione del server con CA come nel caso di TLS, ma un session ID probabilisticamente unico (anche quando uno dei nodi è malevolo).
Un segreto condiviso k può essere usato cosí

1. $C \rightarrow S : HASH(k, C|SessionID)$
2. $S \rightarrow C : HASH(k, S|SessionID)$

Se anche S è malevolo (e k non generabile da un dizionario), non potrà riusare k (non estraibile da $HASH(k, C|SessionID)$) né $HASH(k, C|SessionID)$ perché il $SessionID$ sarà diverso.

Riassumendo

- tcpcrypt è un'estensione di TCP, che permette di cifrare il livello di trasporti
- è molto piú efficiente di TLS perché il carico crittografico è per lo piú spostato sui client
- Il Session ID permette di costruire protocolli di autenticazione a livello applicativo

Lezione VII: Introduzione al laboratorio

Live CD

- Useremo un Live CD: Debian GNU/Linux (http://live.debian.net/)
- Personalizzato per il corso, contiene:
  - busybox
  - bind9-host
  - openssh-client
  - dropbear (ssh-server)
  - nmap
  - tcpdump
  - tshark
  - iptables
  - Virtualizzazione della rete (vde2 + umview)
- Tutti programmi *console-based* per risparmiare spazio e permetterne l'uso anche in condizioni di risorse limitate

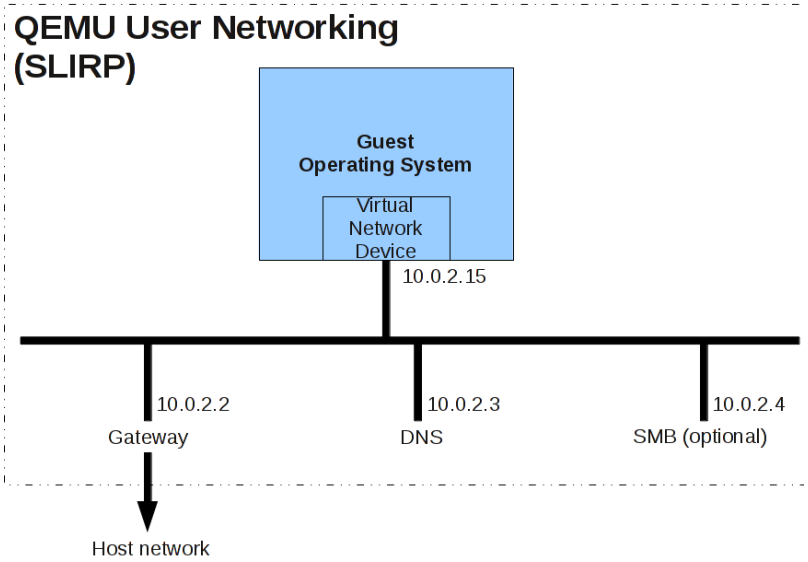
Qemu

- Il Live CD è utilizzabile nativamente o con una macchina virtuale qualsiasi (VirtualBox, VMware, ecc.)
- Gli esercizi però sono pensati per l'uso con Qemu (http://wiki.qemu.org)
  - i440FX host PCI bridge and PIIX3 PCI to ISA bridge
  - Several video card (VGA)
  - PS/2 mouse and keyboard
  - 2 PCI IDE interfaces with hard disk and CD-ROM support
  - Floppy disk
  - Several network adapters (Intel e1000)
  - Serial ports
  - PCI UHCI USB controller and a virtual USB hub.

Qemu virtual network

Qemu fornisce una modalità *user networking*

- Gli indirizzi IP possono essere assegnati a mano o tramite il DHCP server automaticamente attivato da 10.0.2.2
- È possibile redirigere porte *host* su porte *guest* p.es.:




```
qemu -cdrom sicureti.iso -net nic,model=e1000 -net user,hostfwd=tcp::6666-:22
```

Virtual Square

Virtual Square è un progetto di software libero per virtualizzazione dell'Università di Bologna (Renzo Davoli, http://wiki.virtualsquare.org).

- Ben integrato con Qemu (e VirtualBox)
- Vari componenti: virtualizzazione della rete (VDE2, LWIPV6) e dell'interfaccia del sistema operativo (UMview)
- User mode
- Solo in ambienti Unix-like (e grande enfasi sul software libero)

VDE2

Virtual Distributed Ethernet

- vde_switch realizza uno *switch* virtuale
- wire qualsiasi cosa sia capace di fornire uno *stream* di dati può essere un *wire*
  plug un terminale cui è attaccato un *wire* e finisce in uno *switch*
  cable è un *wire* con due *plug* e connette i nodi della rete virtuale

UMview

È lo strumento principale di View OS un approccio alla virtualizzazione in cui ogni processo "vede" una versione personalizzate delle chiamate di sistema

- umview bash
- L'idea è che si programmano moduli in cui si ridefinisce la semantica delle *system call*
- il modulo umnet ridefinisce le chiamate di rete
- umview −V **test** −p umnet bash precarica il modulo umnet e dà al sistema il nome `test` (utile quando ne abbiamo tanti...)

Virtualizzazione dello stack

Grazie a umnet si possono costruire *stack* di rete con le proprietà volute

- mount −t umnetnull none /dev/net/null rete che fa fallire qualsiasi operazione di rete
- mount −t umnetlwipv6 −o vd0=/tmp/switch none /dev/net/prova dove /tmp/switch è una *named pipe* creata con vde_switch
- mstack /dev/net/prova ip link
- Bisogna avere l'accortezza di usare programmi che usano direttamente le *system call* (p.es. ifconfig non va bene perché legge /proc)
- usando /dev/net/default non c'è bisogno di mstack

Lezione VIII: I confini di una rete

Sicurezza perimetrale

- Poiché in Internet è una rete di reti (locali) si parla di protezione del perimetro di sottorete.
- Abbiamo già visto che l'assunzione è `locale == trusted`.
- I firewall vengono usati per definire località parzialmente diverse da quelle imposte dai mezzi trasmissivi (LAN).

Sicurezza perimetrale

Firewall

(*parete tagliafuoco*) è un dispositivo che:

- è al confine fra due reti A e B
- tutto il traffico tra A e B (e viceversa) deve passare attraverso di esso
- filtra il traffico secondo una precisa politica d'accesso (policy)

Compito dei firewall

Il compito dei firewall è stabilire quale traffico ha accesso alla rete (*policy*) e non controllare che il traffico permesso non faccia danni (*control*, intrusion detection).

Cosa sono i firewall

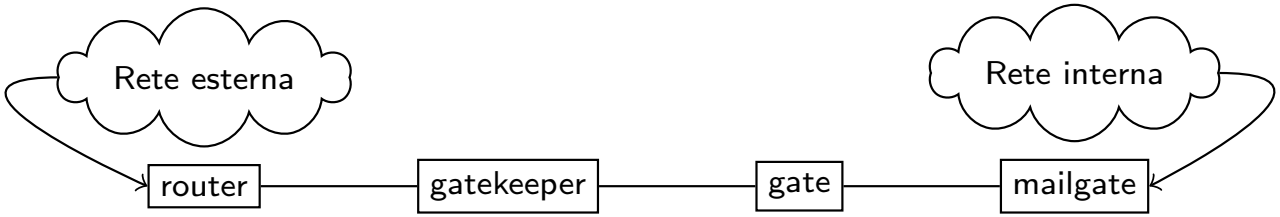
Tipicamente sono realizzati come

- Forwarding gateway
- Filtering router
- Proxy

E stabiliscono politiche (regole) ai vari livelli dello stack TCP/IP

Firewall a vari livelli

I primi firewall (Mogul, 1989 e Ranum, 1992) e



- Gatekeeper proxy applicativo: raccoglie le richieste applicative (Telnet, FTP, SMTP, ...) dall'interno e le manda verso l'esterno
- Gate filtra il traffico

Riassumendo

I firewall

- sono al confine fra due reti
- filtrano il traffico secondo una precisa politica d'accesso (policy)
- servono per definire zone di traffico trusted parzialmente diverse da quelle imposte dalle LAN.

Livelli firewall

In generale si possono avere firewall

- a livello applicativo (application gateway, proxy)
- a livello di trasporto (circuit gateway)
- a livello rete (packet filter)

Livelli firewall

- Esistono anche ibridi: dynamic packet filter agiscono a livello rete e trasporto (e talvolta anche applicativo).
- Possono essere realizzati via software o hardware (piú veloci, ma piú costosi e meno flessibili nelle configurazioni).

Stateless filtering

È il metodo piú semplice e piú comune

Stateless filtering

Ogni pacchetto (o comando protocollare, se a livello applicativo) è valutato in isolamento, senza tenere traccia di quelli precedenti

ACL

In pratica si tratta di avere una Access Control List (ACL) che *filtra* i pacchetti o le richieste, uno alla volta

int addr	int port	ext addr	ext port	action
*	*	`a.b.c.d`	*	block
`192.168.2.3`	110	*	110	allow

Digressione: ACL

Una ACL fissa la politica d'accesso: espressa in maniera compatta (e comprensibile). Come va interpretato *il silenzio* dell'ACL?

default deny Vietato tutto ciò che non è esplicitamente permesso

default permit Permesso tutto ciò che non è esplicitamente vietato

Default deny

Normalmente l'ACL è una serie di regole che vengono esaminate dalla prima all'ultima, quindi se l'ultima regola è equivalente a

int addr	int port	ext addr	ext port	action
*	*	*	*	block

si ha *default deny*

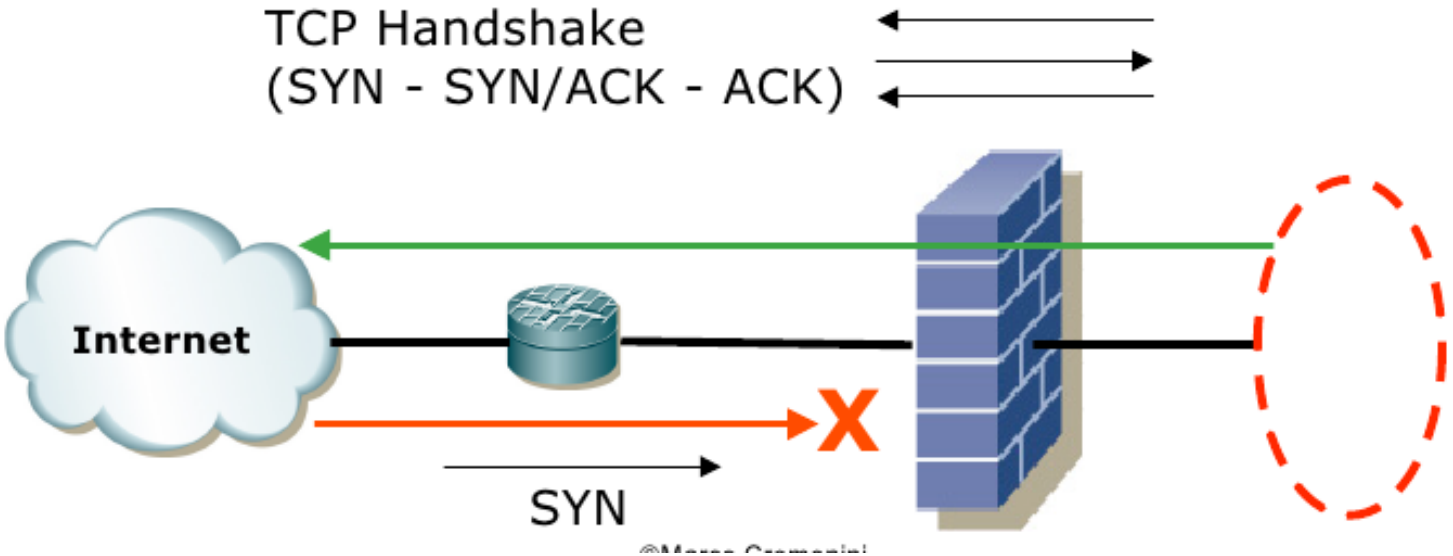
Stateful filtering

Stateful filtering

Si tiene traccia di uno *stato* del sistema e il filtraggio avviene sulla storia dei pacchetti o delle richieste.

Allo scopo occorre mantenere una tabella delle connessioni

Stateful filtering



client addr	client port	ext addr	ext port	state
131.175.12.1	2367	159.132.34.2	22	established

Deep packet inspection

Firewall stateful che operano filtraggio applicativo analizzando il contenuto dei pacchetti vengono talvolta detti deep packet filters.

- Analisi del traffico applicativo, la cui liceità va valutata caso per caso
- Generalmente basati su pattern matching di stringhe

Riassumendo

I firewall si differenziano per

- il livello a cui agiscono
- il tipo di regole di filtraggio
  - stateless
  - stateful
  - "deep packet inspection"

Lezione IX: Pattern ricorrenti

Configurazioni ricorrenti

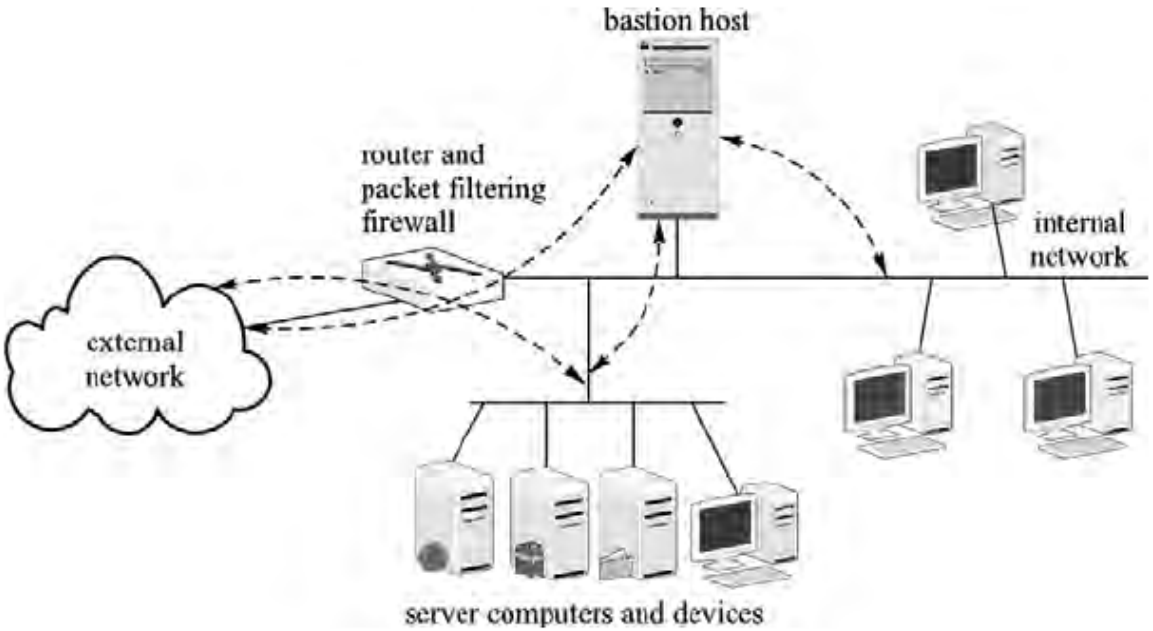
SHBH *Single-homed bastion host*

DHBH *Double-homed bastion host*

DMZ *Demilitarized zone* (o *screened subnet*)

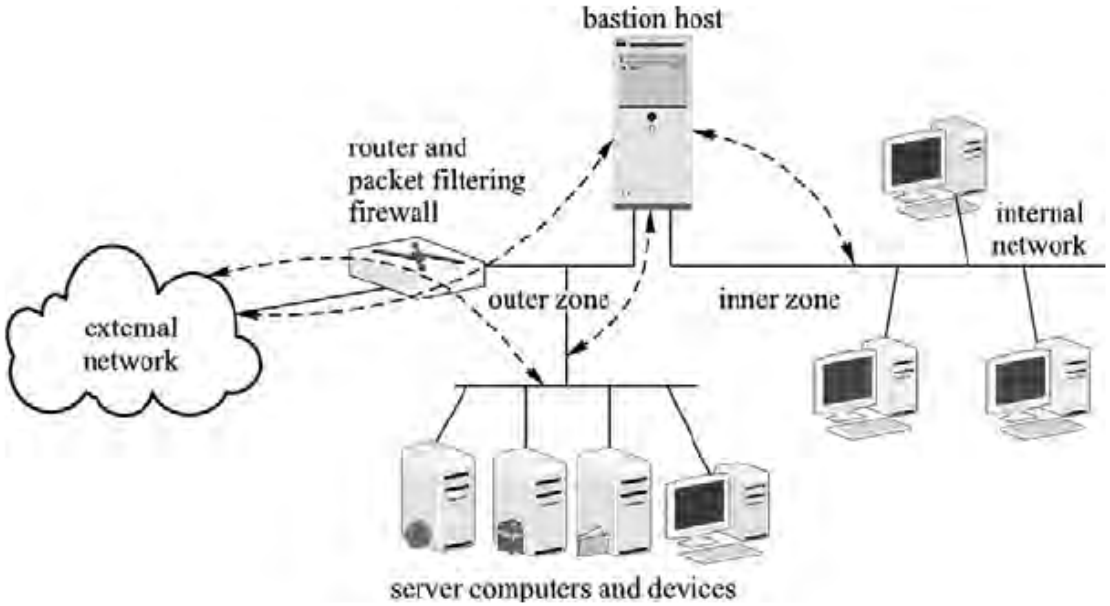
Un *bastion host* è un nodo particolarmente protetto e capace di difesa prolungata che però può essere lasciato al nemico senza danni per la rete interna.

Single-homed bastion host



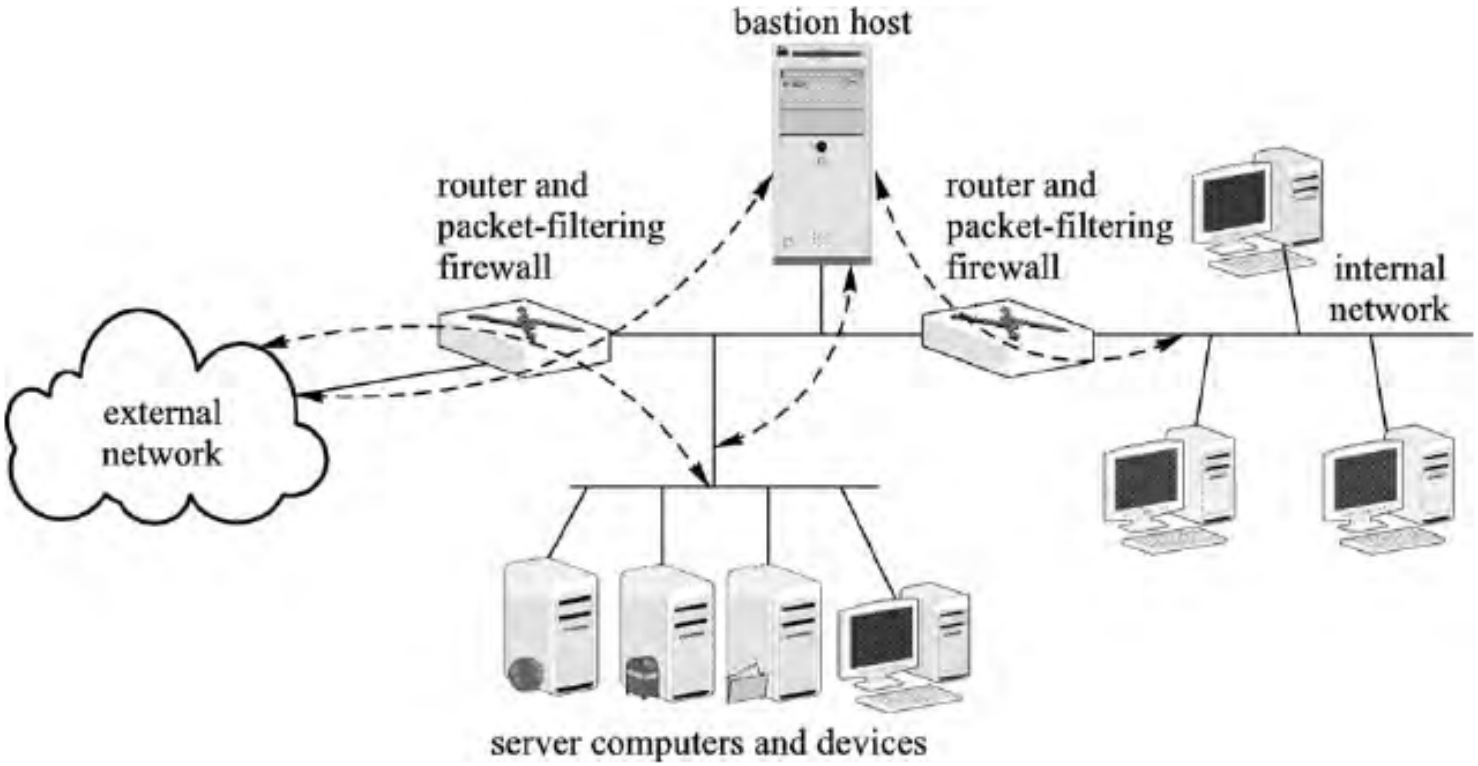
Nel caso il firewall venga compromesso, la rete interna rimane isolata (dal bastion host) dagli attacchi esterni.

Double-homed bastion host



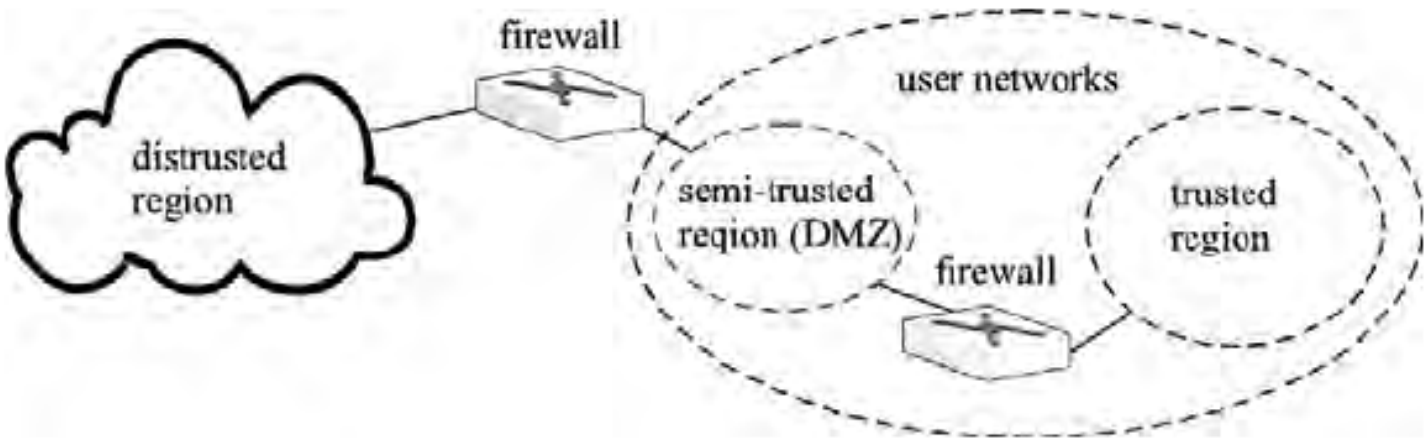
In questo caso si hanno due sottoreti: una “intima” inaccessibile dall’esterno e una piú esterna, ma sempre difesa dal bastion host.

Screened subnet



Si usano due firewall per creare una zona di interdizione

Screened subnet



Si usano due firewall per creare una zona di interdizione

Effetti di un firewall

Grazie al firewall:

- separazione in zone aventi diverso grado di sicurezza
- solo i componenti esterni al firewall sono direttamente accessibili
- è possibile regolare la "direzionalità" delle connessioni (i socket rimangono bidirezionali, naturalmente)

Riassumendo

- un firewall realizza una separazione in zone aventi diverso grado di sicurezza
- Alcune delle configurazioni piú comuni prevedono
  - bastion host
  - zone di interdizione

Stateless filtering TCP

ACL per filtraggio:

verso	IP src	IP dst	prot.	port src	port dst	flag	azione

Stateless filtering TCP

verso	IN/OUT o le zone sorgente e destinazione (es. DMZ→Internet), o delle interfacce (es. eth0→eth1)
IP sorgente/destinatario	indirizzi (es. 159.149.10.1, 159.149.10.0/24) o *variabili*
protocollo	TCP, UDP, ICMP, IP
porta sorgente/destinazione	valore o range (es. > 1023)
flag	se è attivo ACK (solo TCP)
azione	permit, deny

Uso di variabili

- scrivere politiche di tipo generale, che possono essere *istanziate* sulla specifica topologia di rete
- modificare indipendentemente politiche e topologia

Esempio

DMZ := 159.149.70.0/24
Internal := 192.168.20.0/24
Private := 10.0.0.0/8
External := not(Internal or DMZ or Private)
WebServer := 159.149.70.11 and 159.149.70.12

Protezione contro lo spoofing IP

ingress ed egress filtering.

verso	IP src	IP dst	prot.	port src	port dst	flag	azione
OUT	Internal	Any	IP	Any	Any	*	Permit
IN	External	Internal	IP	Any	Any	*	Permit
Any	Any	Any	Any	Any	Any	*	Deny

SSH con stateless filtering

La politica da implementare autorizza solo connessioni SSH dall'interno della rete aziendale verso l'esterno.
semplificazione: identifichiamo SSH con i pacchetti TCP con porta destinazione 22(si noti che talvolta si cambia la porta proprio per ragioni di sicurezza!)

verso	IP src	IP dst	prot.	port src	port dst	flag	azione
OUT	Internal	Any	TCP	> 1023	22	*	Permit
IN	Any	Internal	TCP	22	> 1023	*	Permit
Any	Any	Any	Any	Any	Any	*	Deny

SSH con stateless filtering

In realtà però possiamo notare che i pacchetti provenienti dall'esterno della rete dovrebbero essere solo risposte del server: quindi ACK deve essere settato.
Inoltre solo alcuni server ssh potrebbero essere autorizzati.

SSH

sshSrvs := 159.149.70.13 and 159.149.70.42

verso	IP src	IP dst	prot.	port src	port dst	flag	azione
OUT	Internal	sshSrvs	TCP	> 1023	22	1/0	Permit
IN	sshSrvs	Internal	TCP	22	> 1023	1	Permit
Any	Any	Any	Any	Any	Any	*	Deny

Riassumendo

- ingress ed egress filtering
- La scrittura delle regole di filtering impone di adattare la politica di sicurezza al *modello* imposto dal meccanismo di filtraggio
  - `SSH == tcp port 22`
- Occorre una conoscenza approfondita di protocolli e applicazioni

LPP

Principio del Least Privilege (LPP):

> *"ogni attore dispone del minimo dei privilegi necessari per raggiungere gli obiettivi assegnatigli dalle specifiche del sistema"*

È molto difficile da applicare: c'è una costante tensione fra flessibilità e sicurezza.

Protocolli firewall-friendly

Protocolli come Telnet, SSH, rlogin, etc. sono semplici da gestire:

- per loro natura implicano ruoli ben definiti del client e server
- il pattern di scambio di messaggi è un semplice request/reply

In generale invece esistono protocolli molto piú elaborati che richiedono politiche assai piú sofisticate per applicare il LPP.

SMTP

Politica: Nella rete aziendale un solo server SMTP è autorizzato a gestire la posta elettronica con l'esterno.

- SMTP: protocollo firewall-friendly
- Client interni alla rete non passano per il firewall

SMTP

Primo tentativo: In analogia con quanto fatto per SSH

smtpSrv := 159.149.70.23

External := not(159.149.70.0/24)

verso	**IP src**	**IP dst**	**prot.**	**port src**	**port dst**	**flag**	**azione**
IN	External	smtpSrv	TCP	> 1023	25	1/0	Permit
OUT	smtpSrv	External	TCP	25	> 1023	1	Permit
Any	Any	Any	Any	Any	Any	*	Deny

È corretto?

SMTP

Primo tentativo: In analogia con quanto fatto per SSH

smtpSrv := 159.149.70.23

External := not(159.149.70.0/24)

verso	IP src	IP dst	prot.	port src	port dst	flag	azione
IN	External	smtpSrv	TCP	> 1023	25	1/0	Permit
OUT	smtpSrv	External	TCP	25	> 1023	1	Permit
Any	Any	Any	Any	Any	Any	*	Deny

È corretto? No: le connessioni SYN vengono bloccate

SMTP

Le regole devono essere necessariamente piú sofisticate perché vogliamo:

- Scambiare posta: un Mail Server (MS) riceve e invia posta "da" e "verso" altri MS.
- Ricevere posta: MS si connettono al MS aziendale agendo da client.
- Inviare posta: il MS aziendale si connette ad altri MS agendo da client.

Il tipo di connessioni da gestire non è uno solo!

SMTP

Secondo tentativo:

verso	**IP src**	**IP dst**	**prot.**	**port src**	**port dst**	**flag**	**azione**
IN	External	smtpSrv	TCP	Any	Any	1/0	Permit
OUT	smtpSrv	External	TCP	Any	Any	1/0	Permit
Any	Any	Any	Any	Any	Any	*	Deny

SMTP

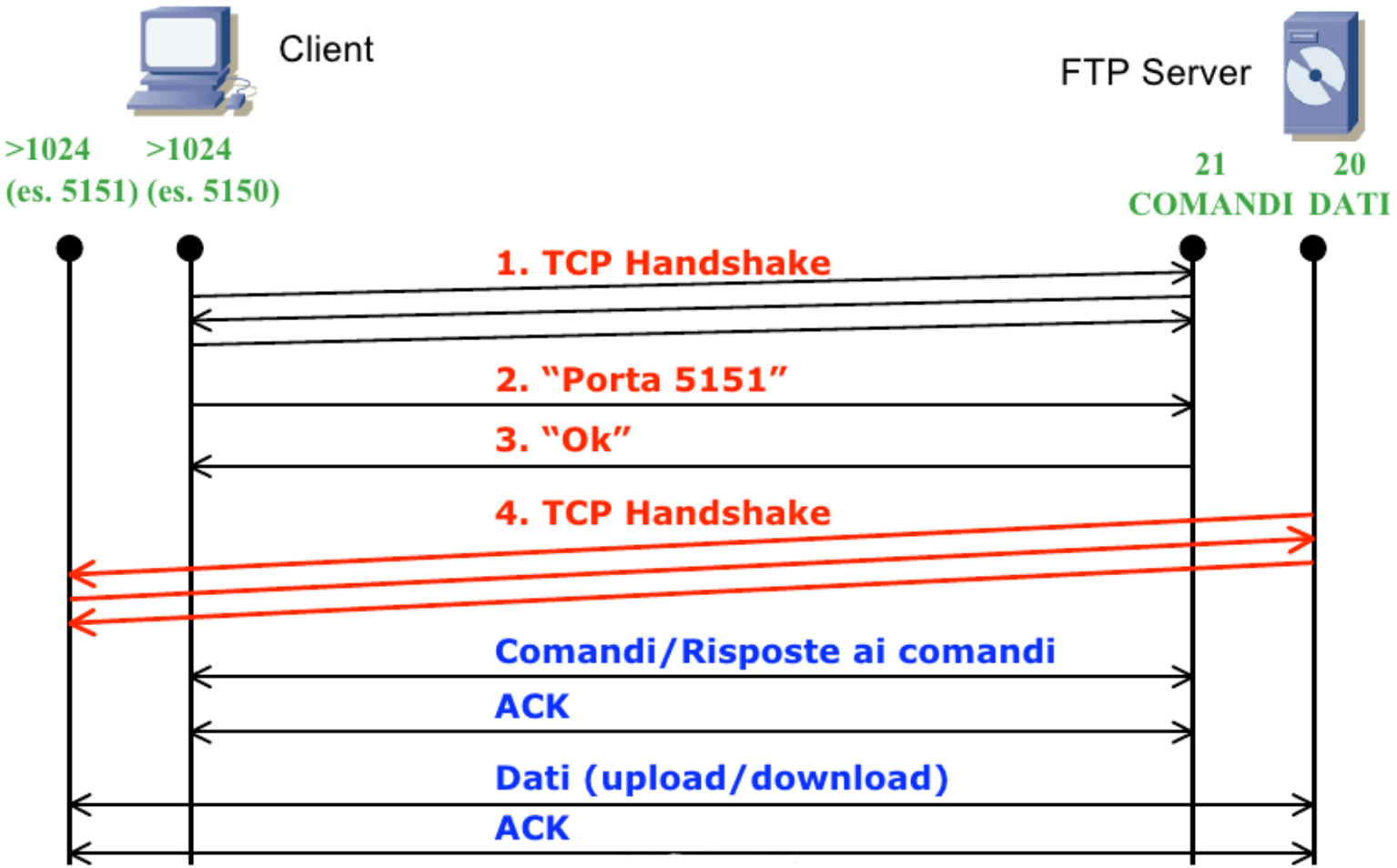
verso	IP src	IP dst	prot.	port src	port dst	flag	azione
IN	External	smtpSrv	TCP	> 1023	25	1/0	Permit
OUT	smtpSrv	External	TCP	25	> 1023	1	Permit
OUT	smtpSrv	External	TCP	> 1023	25	1/0	Permit
IN	External	smtpSrv	TCP	25	> 1023	1	Permit
Any	Any	Any	Any	Any	Any	*	Deny

Riassumendo

- Un principio che dovrebbe ispirare la scrittura delle policy è il Least Privilege
- Non è banale l'applicazione in situazioni realistiche

FTP

FTP non è un protocollo "firewall-friendly"...



FTP

verso	IP src	IP dst	prot.	port src	port dst	flag	azione
OUT	Internal	External	TCP	> 1023	21	1/0	Permit
IN	External	Internal	TCP	21	> 1023	1	Permit
IN	External	Internal	TCP	20	> 1023	1/0	Permit
OUT	Internal	External	TCP	> 1023	20	1	Permit
Any	Any	Any	Any	Any	Any	*	Deny

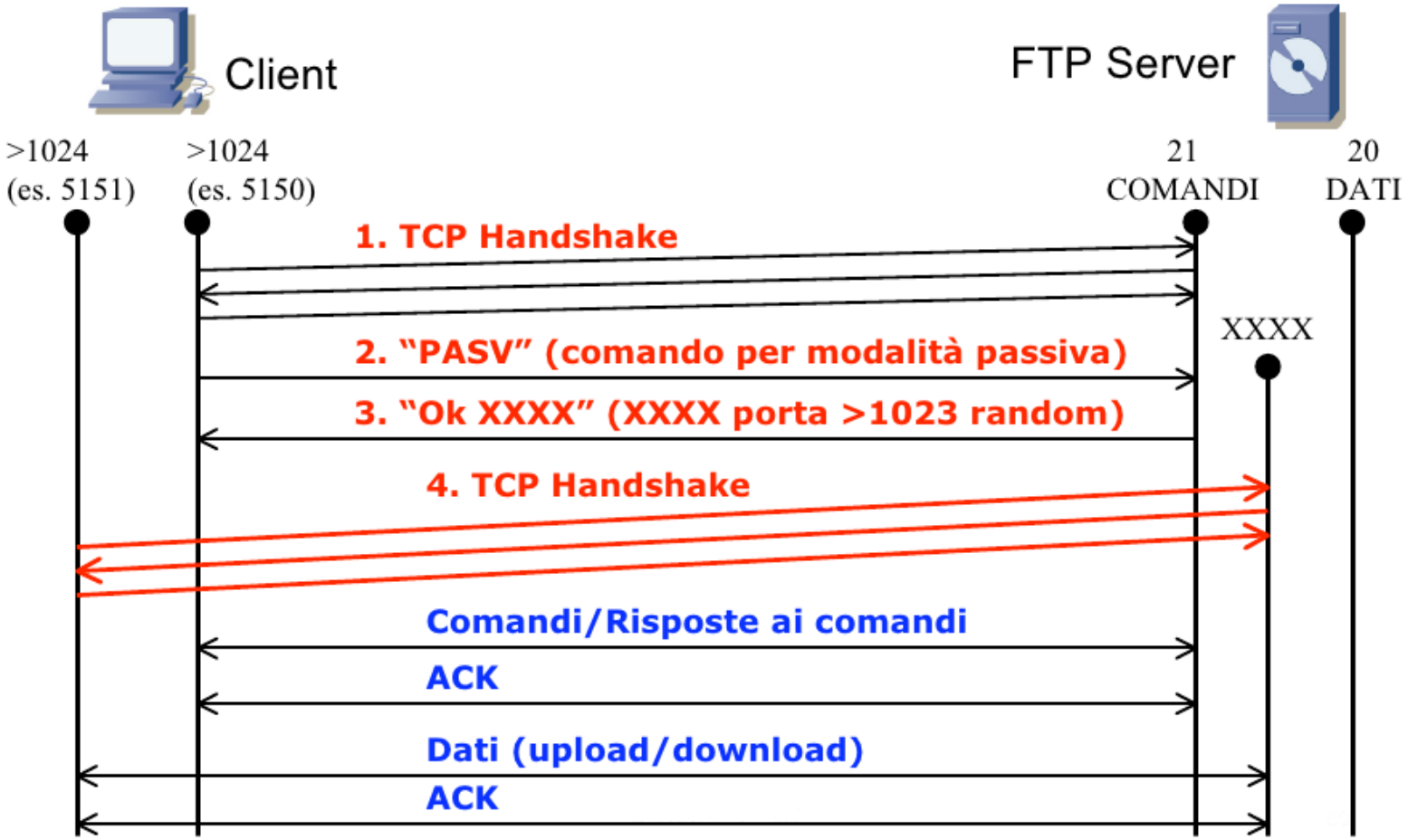
FTP

Il canale dati, dal server verso il client:
ftpserver:20 $\rightarrow$ ftpclient:XXXX
La politica di gestione "solo connessioni da interno a esterno" non è applicabile al caso in oggetto:

- connessione da esterno a interno
- porta di destinazione della connessione non determinata a priori

FTP in “passive mode”

Una nuova versione del protocollo firewall-friendly. . .



FTP in “passive mode”

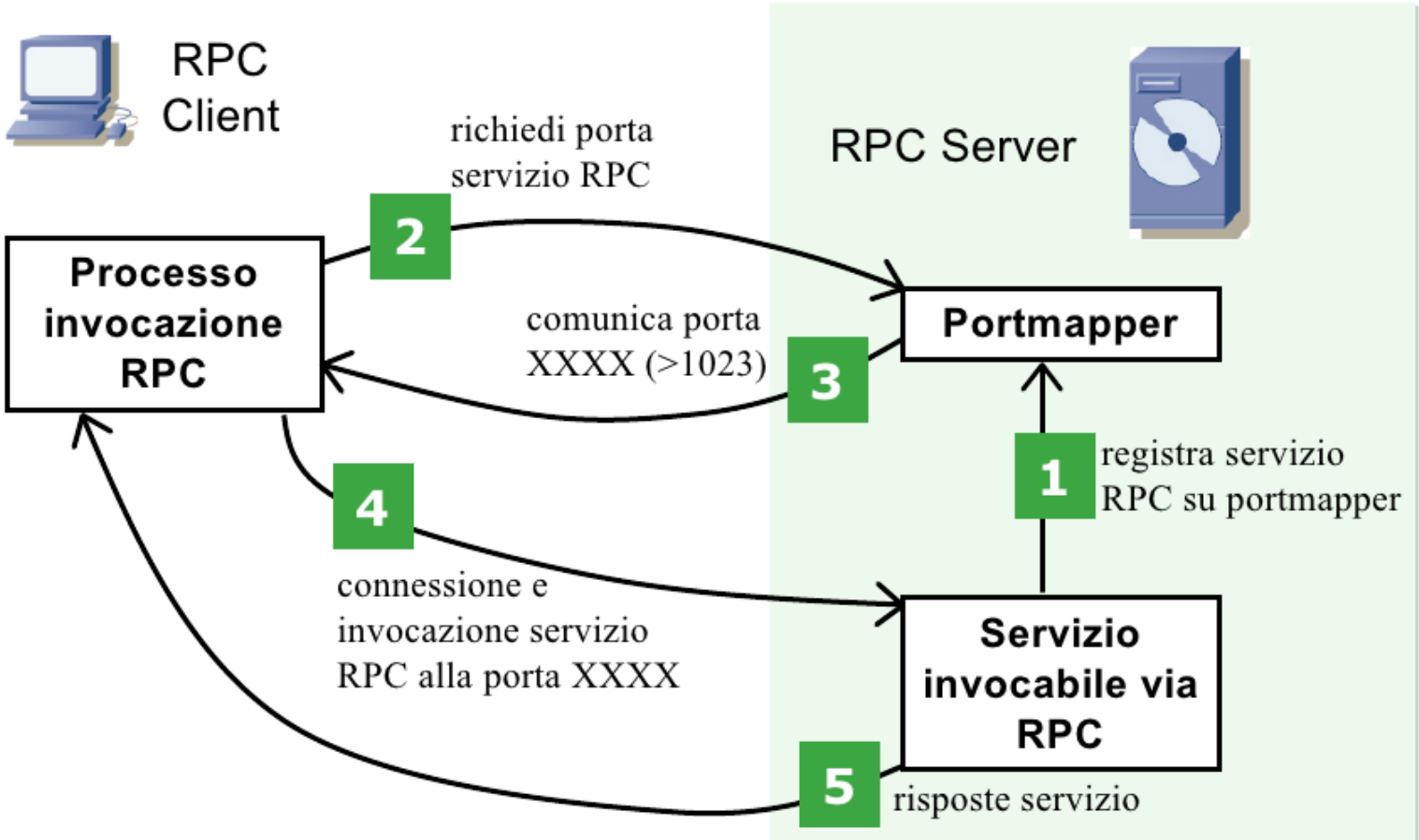
Una nuova versione del protocollo firewall-friendly. . .
La seconda connessione, relativa al canale dati, viene aperta dal client verso il server:
ftpclient:YYYY $\rightarrow$ ftpserver:XXXX
La politica di gestione “solo connessioni solo da interno a esterno” torna ad essere applicabile.

FTP in modo passivo

verso	IP src	IP dst	prot.	port src	port dst	flag	azione
OUT	Internal	External	TCP	> 1023	21	1/0	Permit
IN	External	Internal	TCP	21	> 1023	1	Permit
OUT	Internal	External	TCP	> 1023	> 1023	1/0	Permit
IN	External	Internal	TCP	> 1023	> 1023	1	Permit
Any	Any	Any	Any	Any	Any	*	Deny

RPC

Un protocollo complesso



RPC

Il server RPC (attraverso il servizio Portmapper, nel caso UNIX), determina dinamicamente la porta (> 1023) da assegnare al servizio RPC e quindi non si conosce a priori la porta che il server RPC assegnerà al servizio.
(Versione TCP, Unix)

verso	**IP src**	**IP dst**	**prot.**	**port src**	**port dst**	**flag**	**azione**
IN	External	rpcSrv	TCP	> 1023	111	1/0	Permit
OUT	rpcSrv	External	TCP	111	> 1023	1	Permit
IN	External	rpcSrv	TCP	> 1023	Any	1/0	Permit
OUT	rpcSrv	External	TCP	Any	> 1023	1	Permit
Any	Any	Any	Any	Any	Any	*	Deny

Riassumendo

Alcuni protocolli risultano piú difficili da gestire

- FTP “attivo”
- RPC

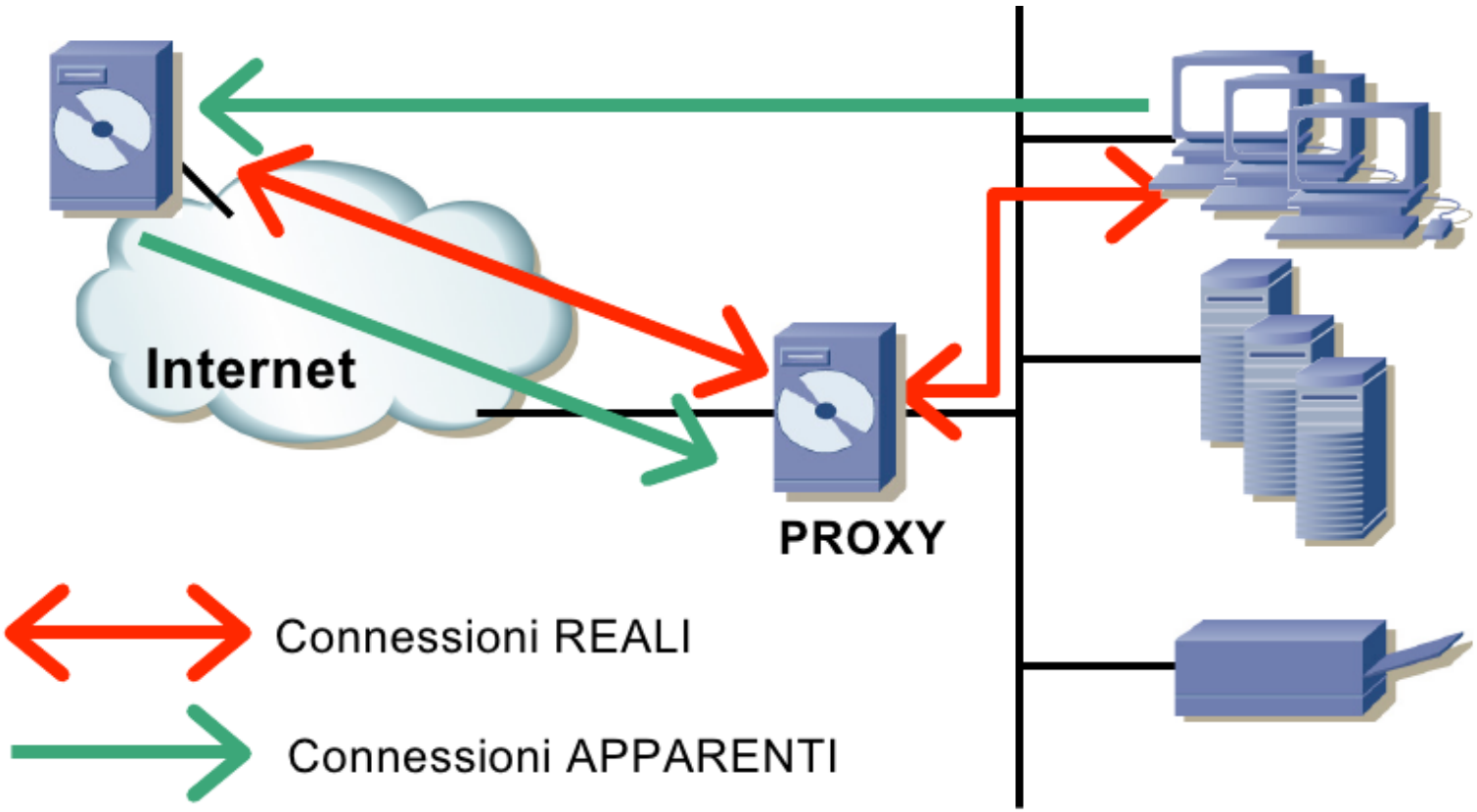
Lezione X: Proxy

Proxy

Un proxy è un componente che media le comunicazioni tra altri due componenti che rimangono inconsapevoli della sua presenza.

- Un proxy disaccoppia la comunicazione tra due componenti rendendola indiretta
- Un proxy agisce sia da client (rispetto al server originale) che da server (rispetto al client originale)

Proxy



Proxy

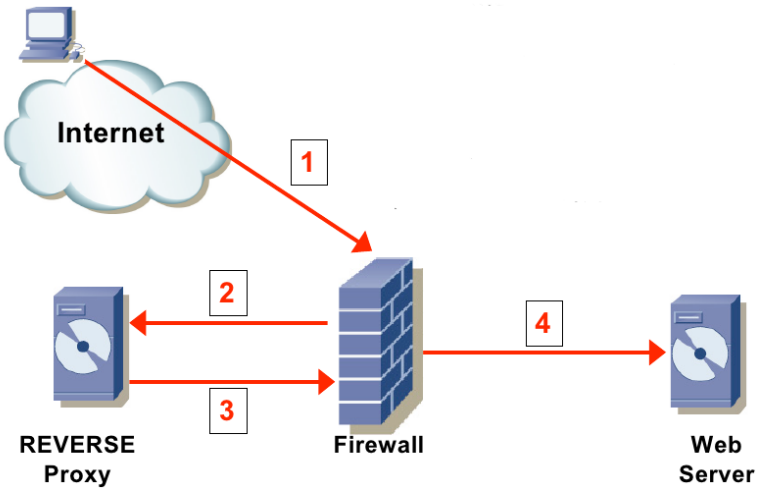
Web Proxy Cache di pagine web.

Anonymizing Proxy Anonimizzazione di connessioni web.

Reverse Proxy Gestiscono l'accesso da utenti esterni a risorse interne.

Proxy Firewall

Reverse proxy

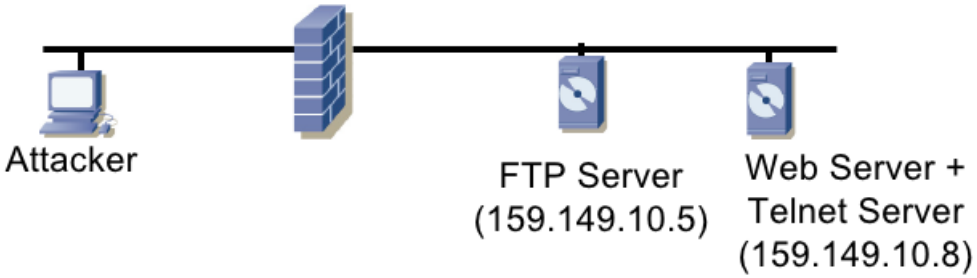


1. Connessione da utente esterno verso il Web Server
2. Redirezione della connessione verso il Reverse Proxy
3. Autenticazione, verifica, filtraggio, ecc...
4. Inoltro verso il Web Server

Proxy firewall

- Può essere usato per analizzare i dati delle applicazioni perché opera a livello applicativo
- Performance potenzialmente molto critiche
- Analogo ad un firewall stateful, ma lavora a livello del protocollo applicativo
- A volte plug-in dei firewall: *protocol decoding*

Firewall proxy per prevenire FTP bounce



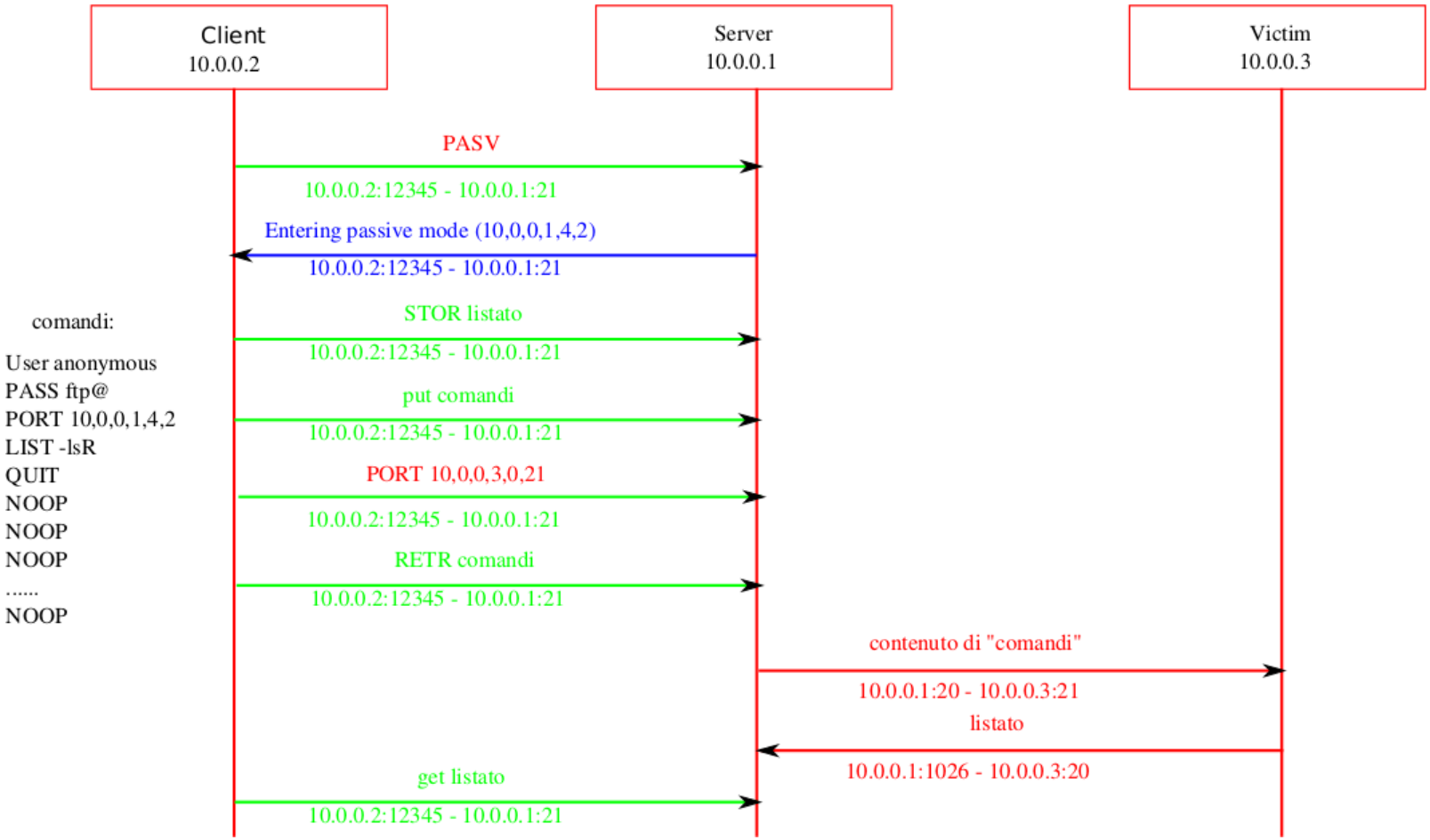
- Il comando PORT di FTP: PORT h1, h2, h3, h4, p1, p2
  - (h1, h2, h3, h4) gli ottetti dell'IP del server
  - $(256 \cdot p1 + p2)$ la porta per la connessione dal server
- PORT 159, 149, 10, 8, 0, 23 (159.149.10.8, porta 23/tcp)
- RETR: apertura di una connessione proveniente dall'FTP server

FTP bounce

Nel caso piú semplice può servire per fare una scansione:

- PORT 159, 149, 10, 8, 0, 23 (159.149.10.8, porta 23/tcp)
- L'attaccante riesce a capire se la porta 23 di 159.149.10.8 accetta connessioni

FTP bounce evoluto



Riassumendo

Un proxy

- firewall stateful che lavorano a livello applicativo
- potenzialmente molto onerosi, ma possono risultare utili quando è possibile prevedere quali applicazioni useranno gli utenti della rete

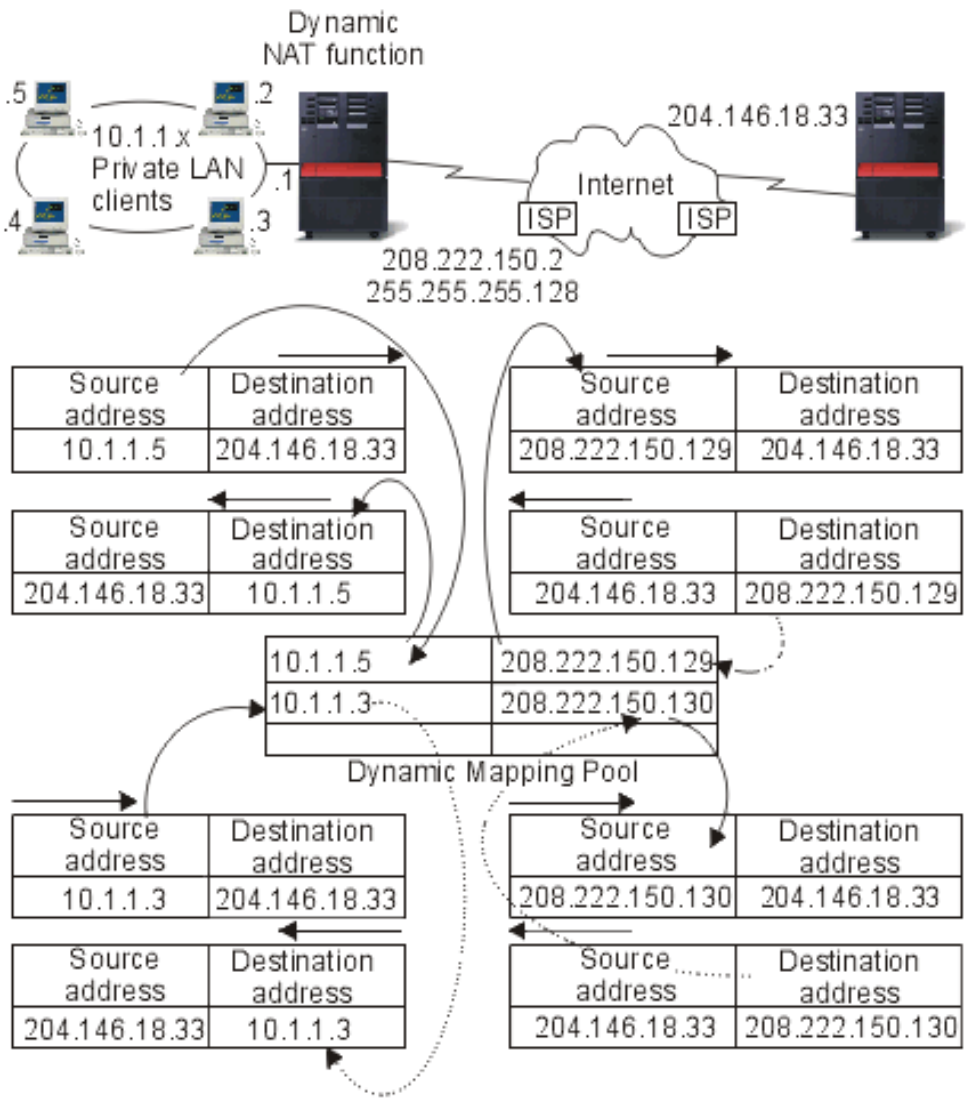
Network Address Translation (NAT) e IP Masquerading

- Consente di manipolare gli indirizzi IP nel passaggio tra le due interfacce di un firewall/router
- Tipicamente viene usato sfruttando le classi di indirizzi IP riservate e non istradabili (10., 172.16-31, e 192.168)
- Maschera gli indirizzi effettivamente utilizzati all'interno della rete

NAT

- Il router modifica gli indirizzi dei pacchetti prima instradarli
- Statico: IP interni mappati staticamente in IP pubblici.
- Dinamico: L'associazione tra IP interno e IP pubblico avviene a run-time

Dynamic NAT

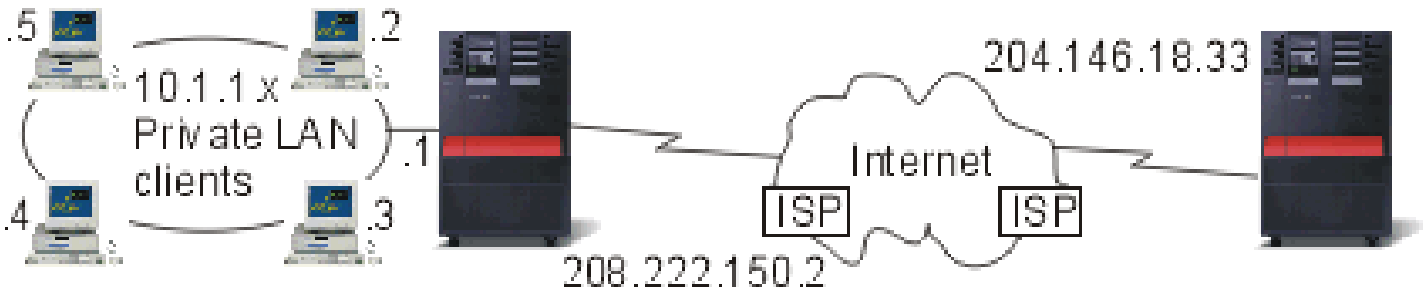


Masquerading

È una forma di NAT capace di lavorare anche ai livelli applicativi.

- Si usa la Port Address Translation: l'associazione avviene modificando la porta sorgente (p.es. ≥ 32536)
- Il masquerading proxy conosce alcuni protocolli e adatta la conversazione alle nuove condizioni (p.es. FTP)

Esempio Masquerading



Masquerade NAT function

Source address	Destination address	SP	DP
10.1.1.5	192.10.1.5	1025	80

Source address	Destination address	SP	DP
208.222.150.2	204.146.18.33	55336	80

Address	10.1.1.5	208.222.150.2
Port	1025	55336

Masquerade mapping table

Source address	Destination address	SP	DP
192.10.1.5	10.1.1.5	80	1025

Source address	Destination address	SP	DP
204.146.18.33	208.222.150.2	80	55336

Riassumendo

NAT e masquerading

- nate piú come tecniche di gestione della rete, che misure di sicurezza
- nascondo la rete interna, garantendo l'irrangiugibilità diretta

Lezione XI: Rilevamento delle intrusioni

Intrusion detection system

IDS

Un sistema di monitoraggio (generalmente del tutto passivo) che genera allarmi

Tre fasi:

1. Raccolta dati
2. Analisi dei dati
3. Generazione degli allarmi

Perché usare un IDS?

In generale i sistemi di monitoraggio sono utili perché:

- le tecnologie di prevenzione degli eventi indesiderati o pericolosi possono fallire
- è utile avere un un meccanismo di segnalazione che permetta di attivare procedure di correzione o di emergenza
- l'uso di strumenti che permettano di monitorare lo stato corrente di un sistema, sia esso un componente che una rete, per accumulare conoscenza statistica sulle modalità d'uso

HIDS e NIDS

In base al punto in cui avviene la raccolta dati

- **HIDS** (Host-based Intrusion Detection System) sistemi che analizzano informazioni relative all'attività locale di un singolo host (log di sistema, accesso a file critici, ...)
- **NIDS** (Network Intrusion Detection System) sistemi che utilizzano le informazioni raccolte da analizzatori di traffico di rete.

Rilevazione di eventi critici

L'analisi dei dati raccolti per rilevare una situazione d'allarme:

Misuse detection Si caratterizza l'abuso: si rilevano le situazioni che ricadono nella descrizione di un attacco (sono detti anche signature based)

Anomaly detection Si caratterizza l'uso normale: si rilevano le situazioni che si scostano dal "normale" funzionamento in modo da poter rilevare anche attacchi ancora sconosciuti

Misuse detection

Elencare le situazioni illecite

Signature Detection

L'amministratore definisce pattern (signature) predefiniti di usi non conformi e il sistema analizza gli eventi monitorati(di rete, di sistema, nei log) rispetto all'elenco di pattern

È la tecnica piú affermata e diffusa

Problemi

I misuse detection rilevano solo attacchi che corrispondono a schemi noti

- Regole rigide non rilevano attacchi non noti e varianti (a volte l'IDS è cosí rigido che basta cambiare un bit per evadere la rilevazione)
- Piú le regole sono flessibili e piú aumenta la complessità di gestione/configurazione
- l'elenco di firme deve essere adattato alle specificità della rete monitorata

Anomaly detection

Anomalie rispetto

- ad eventi singoli: esempio azioni "anomale" di un utente rispetto un profilo d'uso predefinito
  es: `http://example.com/##&<>623??%`
- a dati aggregati: tipicamente, deviazioni rispetto a parametri statistici
  es: il traffico con sorgente 123.45.67.88 è di 5GB al minuto

Anomaly detection

- \+ Non dipende dalla conoscenza puntuale di tutte le modalità di intrusione
- – Molto complesso da realizzare (come fare il modello dell'uso "normale"?) e oneroso da gestire
- Non forniscono informazioni su quale vulnerabilità l'attaccante intende colpire

Modelli di uso normale

Il sistema monitorato inizialmente durante gli usi normali.

- L'ipotesi di assenza di compromissione e normalità non è facile da verificare: in fase di test si rischia l'anomalia
- Impiego di tecniche come data mining, analisi bayesiana, ecc. . .
- È molto complesso tenere il passo con l'evoluzione del sistema
- il monitoraggio introduce inefficienze maggiori rispetto ai misuse

Usi consolidati dell'anomaly detection

Ci sono casi in cui l'anomaly detection funziona bene e il loro uso è ormai standard

- hashing dei file (HIDS): l'integrità dei file del sistema controllata con hash da una distribuzione originale (es. Tripwire)
- Protocol Anomaly Detection (NIDS): analizzato il traffico di rete rispetto alle specifiche del protocollo applicativo

Riassumendo

- HIDS e NIDS
- Misuse e anomaly detection

Falsi allarmi

In generale in tutti gli IDS (misuse e anomaly) occorre bilanciare

falsi negativi attacchi non rilevati

falsi positivi attacchi rilevati corrispondenti a situazioni normali

Falsi allarmi

- Quanto piú la rilevazione è specifica (es. firme molto dettagliate) tanto piú aumenta il carico computazionale e la rilevazione diventa sensibile a variazioni dell'evento analizzato.
- Quanto piú la rilevazione si fa lasca (es. firme generiche) tanto piú il carico computazionale cala, la rilevazione dell'evento analizzato risulta poco influenzata da varianti ma tanto piú vengono rilevati eventi simili ma non pericolosi.

Relazione fra FP e FN

FP e FN risultano correlati inversamente: agendo per diminuire l'una, tipicamente l'altra aumenta.
Il problema si ripropone in moltissime discipline (information retrieval, farmacologia, ...): ogni volta che si ha una decisione binaria (test)

	positivo	negativo
attacco	TP	FN
non attacco	FP	TN

$$TP + TN + FP + FN = totale$$

Relazione fra FP e FN

FP Type I error, falso allarme

FN Type II error, miss

sensibilità del test, recall, hit rate, TPR $\frac{TP}{TP+FN}$

specificità del test $\frac{TN}{TN+FP}$

accuratezza del test $\frac{TP+TN}{totale}$

precisione del test $\frac{TP}{TP+FP}$

FPR $\frac{FP}{TN+FP} = 1 - \text{specificità}$

Quali regole di IDS hanno lavorato meglio?

A	allarme	¬allarme
attacco	TP=63	FN=37
¬attacco	FP=28	TN=72

B	allarme	¬allarme
attacco	TP=77	FN=23
¬attacco	FP=77	TN=23

C	allarme	¬allarme
attacco	TP=24	FN=76
¬attacco	FP=88	TN=12

D	allarme	¬allarme
attacco	TP=76	FN=24
¬attacco	FP=12	TN=88

Quali regole di IDS hanno lavorato meglio?

A	allarme	¬allarme
attacco	TP=63	FN=37
¬attacco	FP=28	TN=72

B piú sensibile	allarme	¬allarme
attacco	TP=77	FN=23
¬attacco	FP=77	TN=23

C	allarme	¬allarme
attacco	TP=24	FN=76
¬attacco	FP=88	TN=12

D piú specifico, accurato, preciso	allarme	¬allarme
attacco	TP=76	FN=24
¬attacco	FP=12	TN=88

Quali regole di IDS hanno lavorato meglio?

A (FPR:0.28,TPR:0.63)	allarme	¬allarme
attacco	TP=63	FN=37
¬attacco	FP=28	TN=72

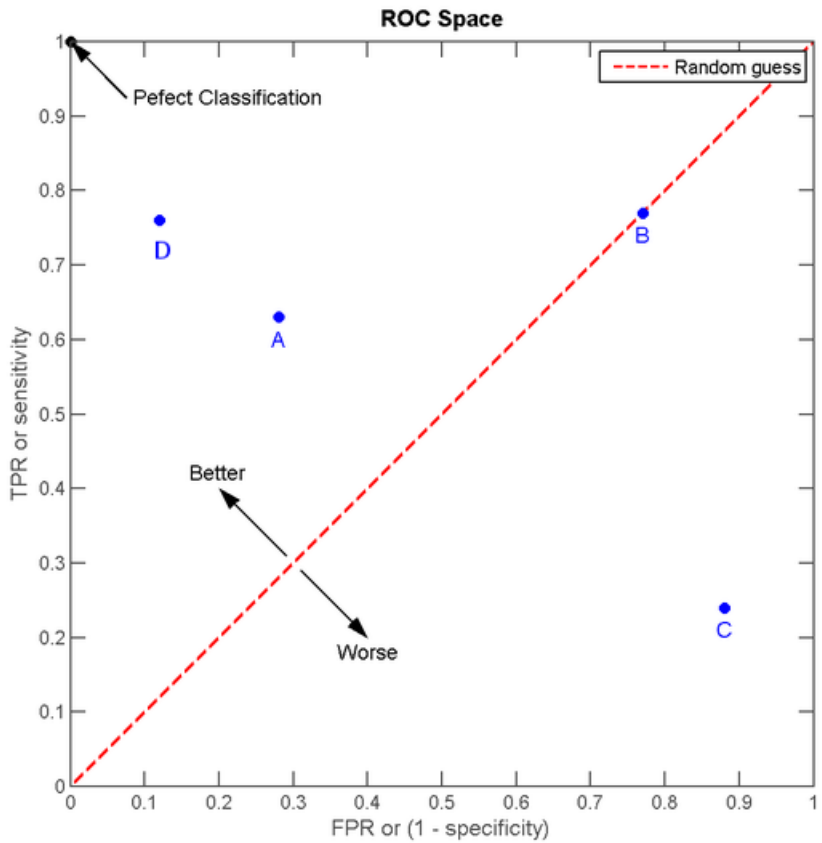
B (0.77,0.77)	allarme	¬allarme
attacco	TP=77	FN=23
¬attacco	FP=77	TN=23

C (0.88,0.24)	allarme	¬allarme
attacco	TP=24	FN=76
¬attacco	FP=88	TN=12

D (0.12,0.76)	allarme	¬allarme
attacco	TP=76	FN=24
¬attacco	FP=12	TN=88

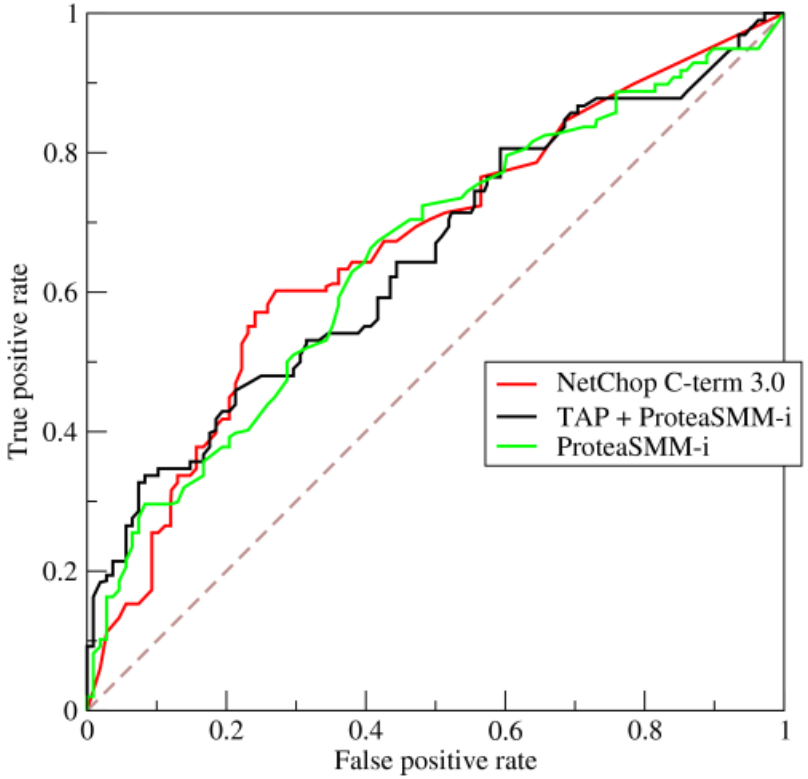
ROC

receiver operating characteristic (ROC): sensibilità vs. tasso dei falsi positivi



E un insieme di regole?

E come valutare l'efficacia di un insieme di regole per tutti i parametri possibili? A volte si usa l'Area under curve.



Riassumendo

Il problema degli IDS sono i falsi allarmi e le mancate segnalazioni

- la qualità di un IDS sta nella relazione fra FP e FN
- per valutarla ci si serve di ROC e AUC

I falsi allarmi

Gli strumenti di analisi come ROC permettono di valutare l'efficacia *ex-post*, valutando il peso di FP e FN in un determinato contesto sperimentale.

Un IDS genera migliaia di allarmi al giorno: qual è la probabilità che un allarme sia davvero relativo a un attacco?

L'intuito inganna perché dipende in maniera complessa dalla probabilità a priori di un attacco.

Esempio dalla letteratura medica

La probabilità che una donna sviluppi un cancro al seno è 0.8%. Se una donna ha il cancro al seno, la probabilità che il suo mammogramma sia positivo è 90%; se non ha il cancro al seno, c'è comunque una probabilità del 7% che il mammogramma sia positivo. Se il mammogramma di una donna è positivo, qual è la probabilità che abbia effettivamente il cancro al seno?

Esempio dalla letteratura medica

La probabilità che una donna sviluppi un cancro al seno è 0.8%. Se una donna ha il cancro al seno, la probabilità che il suo mammogramma sia positivo è 90%; se non ha il cancro al seno, c'è comunque una probabilità del 7% che il mammogramma sia positivo. Se il mammogramma di una donna è positivo, qual è la probabilità che abbia effettivamente il cancro al seno?

Esempio da "Quando i numeri ingannano", di G. Gigerenzer: studi su medici mostrano che la risposta piú frequente al problema cosí posto è 90%.

Esempio: soluzione

- Su 1000 donne, 992 sono sane e 8 malate.
- Delle 8 malate, il 90% ($\simeq 7$) risulteranno positive e il 10% negative ($\simeq 1$).
- Delle 992 sane, il 7% ($\simeq 70$) risulteranno positive e il 93% negative ($\simeq 922$).
- Le positive saranno quindi $7 + 70 = 77$, delle quali sono malate 7: la probabilità che un mammogramma positivo sia indice di malattia è quindi $\frac{7}{77} = 9\%$

Teorema di Bayes

Teorema di Bayes

$$\Pr(\text{attacco}|\text{allarme}) = \frac{\Pr(\text{allarme}|\text{attacco}) \cdot \Pr(\text{attacco})}{\Pr(\text{allarme})}$$

Teorema di Bayes

Teorema di Bayes

$$\Pr(\text{attacco}|\text{allarme}) = \begin{cases} \dfrac{\Pr(\text{allarme}|\text{attacco}) \cdot \Pr(\text{attacco})}{\Pr(\text{allarme})} \\ \dfrac{\Pr(\text{allarme}|\text{attacco}) \cdot \Pr(\text{attacco})}{\Pr(\text{allarme}|\text{attacco}) \cdot \Pr(\text{attacco}) + \Pr(\text{allarme}|\neg\text{attacco}) \cdot \Pr(\neg\text{attacco})} \end{cases}$$

Teorema di Bayes

Teorema di Bayes

$$\Pr(\text{attacco}|\text{allarme}) = \begin{cases} \dfrac{\Pr(\text{allarme}|\text{attacco}) \cdot \Pr(\text{attacco})}{\Pr(\text{allarme})} \\ \dfrac{\Pr(\text{allarme}|\text{attacco}) \cdot \Pr(\text{attacco})}{\Pr(\text{allarme}|\text{attacco}) \cdot \Pr(\text{attacco}) + \Pr(\text{allarme}|\neg\text{attacco}) \cdot \Pr(\neg\text{attacco})} \\ \dfrac{\frac{\text{TP}}{\text{TP+FN}} \cdot \Pr(\text{attacco})}{\frac{\text{TP}}{\text{TP+FN}} \cdot \Pr(\text{attacco}) + \frac{\text{FP}}{\text{FP+TN}} \cdot \Pr(\neg\text{attacco})} \end{cases}$$

Teorema di Bayes

Teorema di Bayes

$$\Pr(\text{attacco}|\text{allarme}) = \begin{cases} \dfrac{\Pr(\text{allarme}|\text{attacco}) \cdot \Pr(\text{attacco})}{\Pr(\text{allarme})} \\ \dfrac{\Pr(\text{allarme}|\text{attacco}) \cdot \Pr(\text{attacco})}{\Pr(\text{allarme}|\text{attacco}) \cdot \Pr(\text{attacco}) + \Pr(\text{allarme}|\neg\text{attacco}) \cdot \Pr(\neg\text{attacco})} \\ \dfrac{\frac{\text{TP}}{\text{TP+FN}} \cdot \Pr(\text{attacco})}{\frac{\text{TP}}{\text{TP+FN}} \cdot \Pr(\text{attacco}) + \frac{\text{FP}}{\text{FP+TN}} \cdot \Pr(\neg\text{attacco})} \end{cases}$$

Per calcolare la probabilità di un allarme veritiero occorre sempre stimare la probabilità a priori di un attacco, che è spesso (fortunatamente!) piuttosto bassa: i falsi allarmi sono inevitabilmente comuni, a meno di avere un IDS straordinariamente preciso o asset particolarmente appetibili.

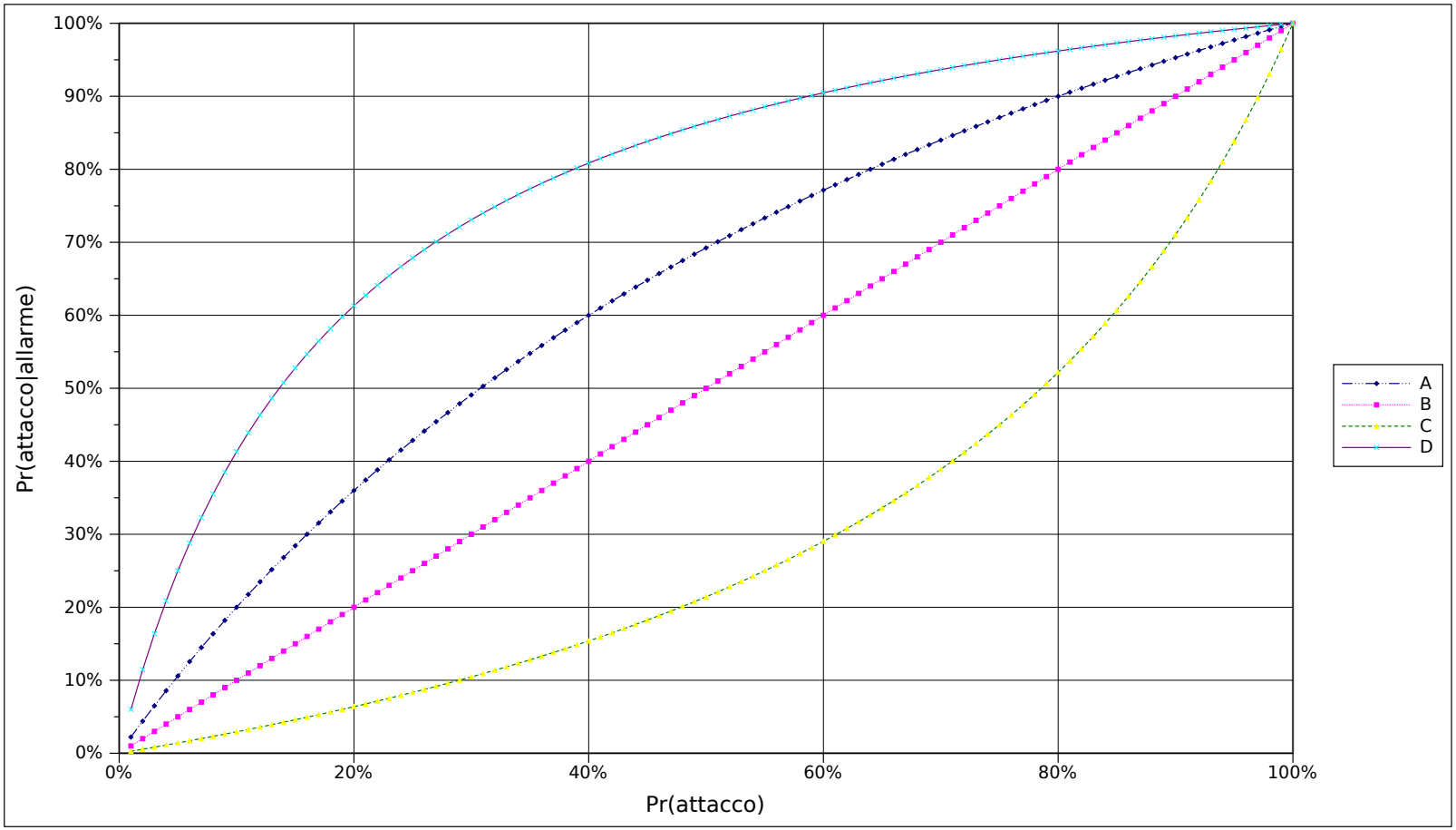
Esempio IDS

Riprendiamo il migliore IDS esaminato con la curva ROC

D (0.12,0.76)	allarme	¬allarme
attacco	TP=76	FN=24
¬attacco	FP=12	TN=88

- Nell'esperimento:
  $\Pr(\text{attacco}) = \frac{76+24}{76+24+12+88} = 50\%$
  $\rightsquigarrow \Pr(\text{attacco}|\text{allarme}) = 86\%$
- Attacchi poco frequenti
  $\Pr(\text{attacco}) = 1\%$
  $\rightsquigarrow \Pr(\text{attacco}|\text{allarme}) = 6\%$

Esempi


100%
90%
80%
70%
60%
50%
40%
30%
20%
10%
0%
Pr(attacco|allarme)
0%
20%
40%
60%
80%
100%
Pr(attacco)
A
B
C
D

Riassumendo

- L'amministratore di un IDS è interessato a stimare la probabilità che un allarme sia davvero relativo a un attacco.
- dipende però dalla probabilità *a priori* di un attacco.

Aspetti architetturali

Un NIDS complementa altre soluzioni, con un'architettura a diversi livelli (defense in-depth).
Importanti aspetti architetturali:

- Quanti sensori installare nella rete
  - costi e complessità di gestione
- Dove installarli
  - quantità vs. ridondanza di informazioni
- Come gestire i dati
  - analisi e logging centralizzato vs. distribuito

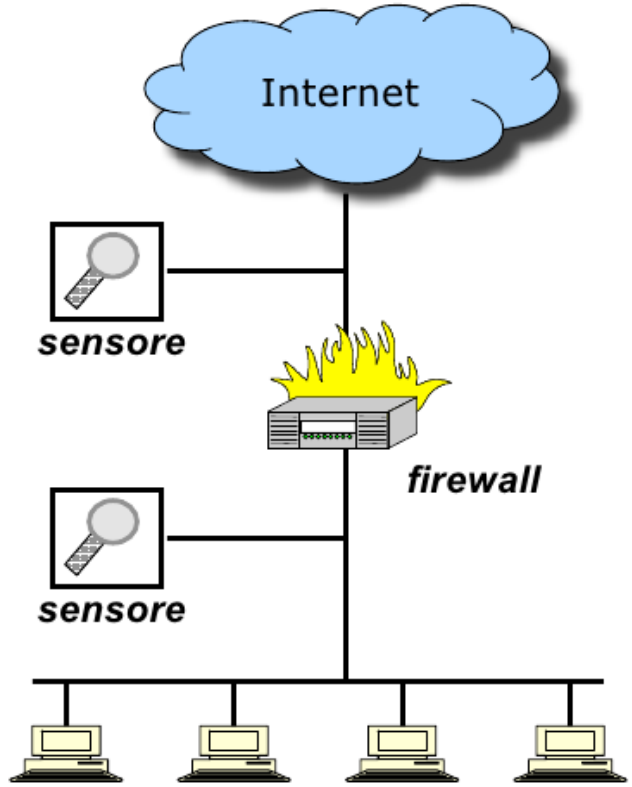
Sensori e firewall

Esterno

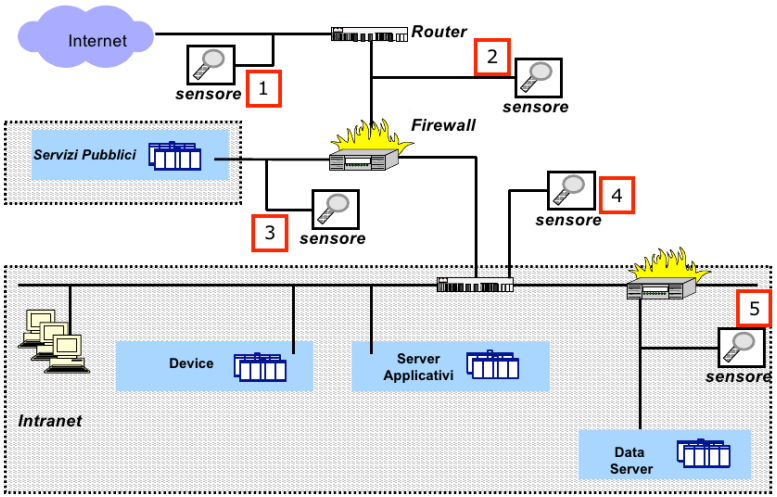
- Rileva l'intero traffico diretto alla rete
- Piú dati
- Piú allarmi

Interno

- Rileva solo il traffico che entra effettivamente
- Verifica l'efficacia del firewall
- Non fornisce info sugli attacchi bloccati dal fw

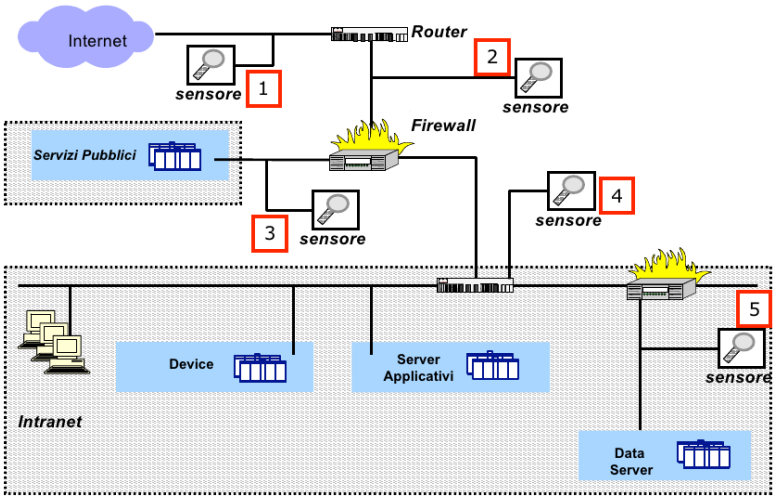


Posizionamento nella rete aziendale



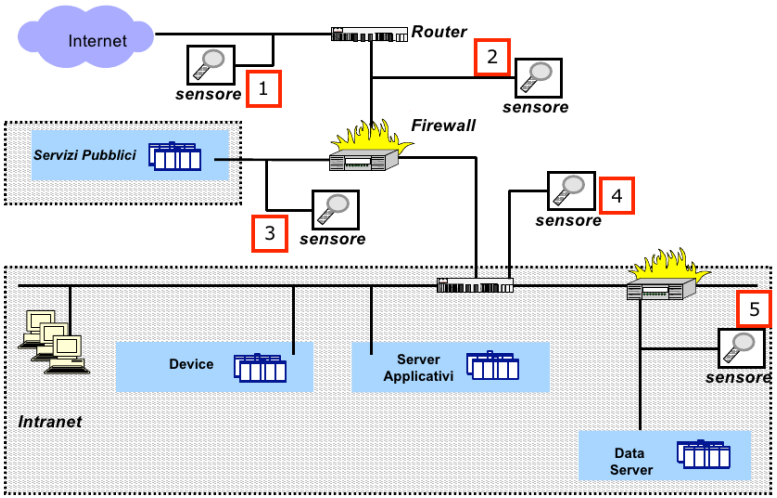
1. Esterno al border router
   - Tutto il traffico diretto alla rete aziendale.
   - Informazione completa e non filtrata.
   - Tanti dati e allarmi

Posizionamento nella rete aziendale



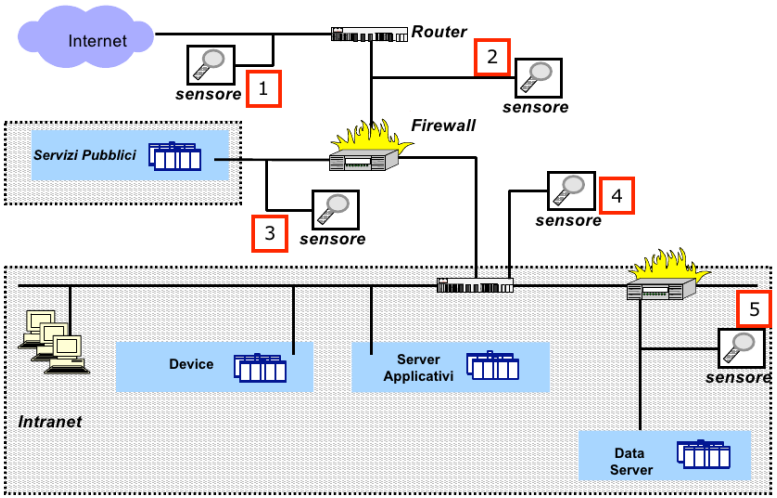
2. Tra border router e firewall
   - Tutto il traffico meno quello filtrato
   - Tanti dati, molti falsi allarmi.

Posizionamento nella rete aziendale



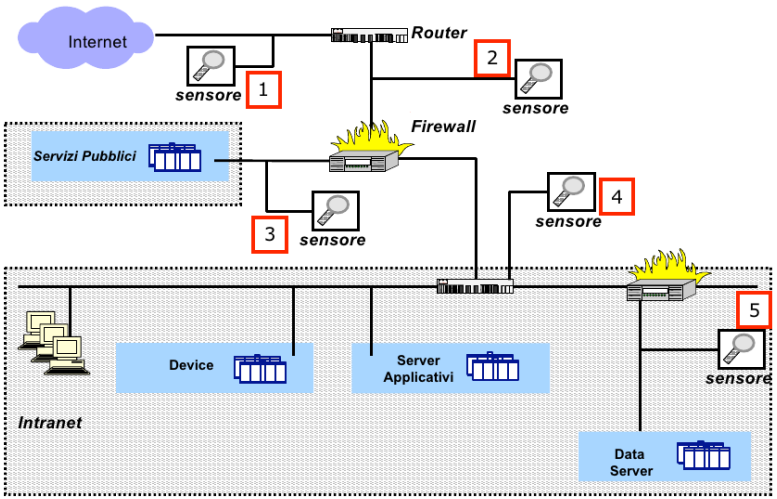
3. Sulla rete dei servizi pubblici, dietro il firewall
   - Tutto il traffico autorizzato dal firewall e diretto ai servizi pubblici.
   - Possibilità filtraggio mirato.
   - Eventuale traffico illecito dai server pubblici.

Posizionamento nella rete aziendale



4. Sulla Intranet
   - Sia il traffico da reti piú esposte (es. DMZ) che interno alla Intranet.
   - Rileva eventuali usi non leciti interni.
   - Difficile dare firme, molti falsi allarmi.

Posizionamento nella rete aziendale



5. Su di un segmento critico della rete aziendale

- Monitora le connessioni dirette ad alcune risorse particolarmente critiche della rete aziendale per le quali si richiede un livello di sicurezza piú elevato (es. i server contenenti dati aziendali sensibili).
- Servizi specifici, quindi possibilità di configurazione mirata delle firme.

Riassumendo

Il posizionamento dell'IDS influisce molto sulla sua efficacia

1. Esterno
2. Tra router e firewall
3. Vicino ai servizi
4. Sulla intranet
5. In punti critici

Lezione XII: Risposta

Risposta di un NIDS

- La tipica risposta di un NIDS al verificarsi di un evento che verifica una firma è la generazione di un allarme
- La forma piú standard di allarme è la scrittura in un corrispondente file di log

Risposta di un NIDS

```
[1:1122:2 ] WEBMISC /etc/passwd [Classification: Attempted
Information Leak ] [Priority:2 ] 09/1610:04:15.826116
192.168.1.1:3143 >192.168.1.2:80 TCP TTL:128 TOS:0x0
ID:12832 IpLen:20 DgmLen:149 DF ***AP***Seq:0xDEFF5454
Ack:0x1A51AF74 Win:0x4470
```

Esistono molte varianti implementate dai diversi NIDS, tra cui salvataggio in formato tcpdump, scrittura su database (es. MySQL), visualizzazione a video ecc.

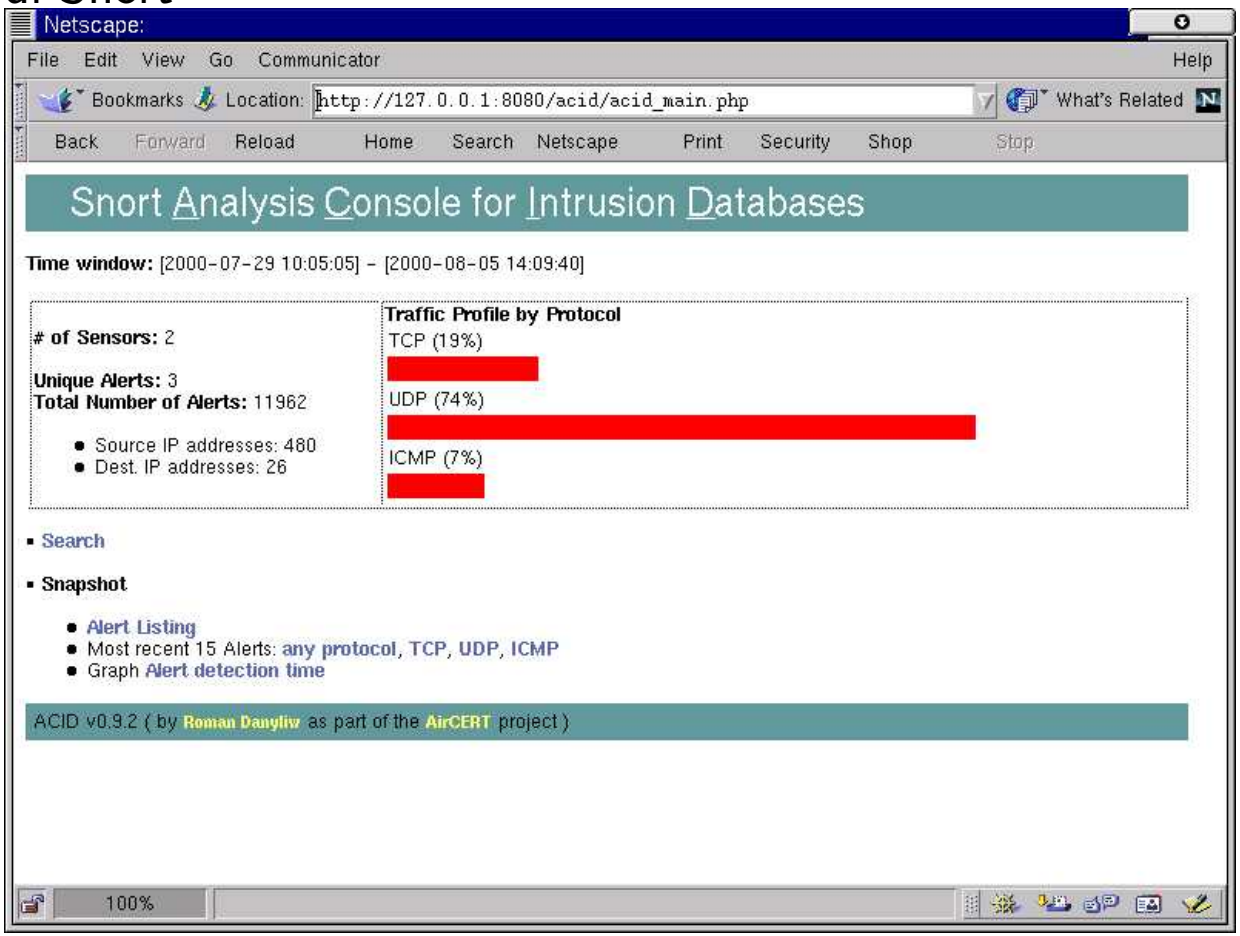
Analisi

La mole di dati è imponente.

- Esistono molti strumenti, sia open-source che integrati nei prodotti commerciali, di analisi dei log prodotti da un NIDS.
- Tipicamente vengono mostrati grafici, statistiche ecc. Sono utili per le analisi *post-mortem* e per il tuning dei sistemi, ma inefficaci per un'azione di contenimento real-time
- L'invio di email a un amministratore è un'altra modalità di risposta diffusa (e onerosa).

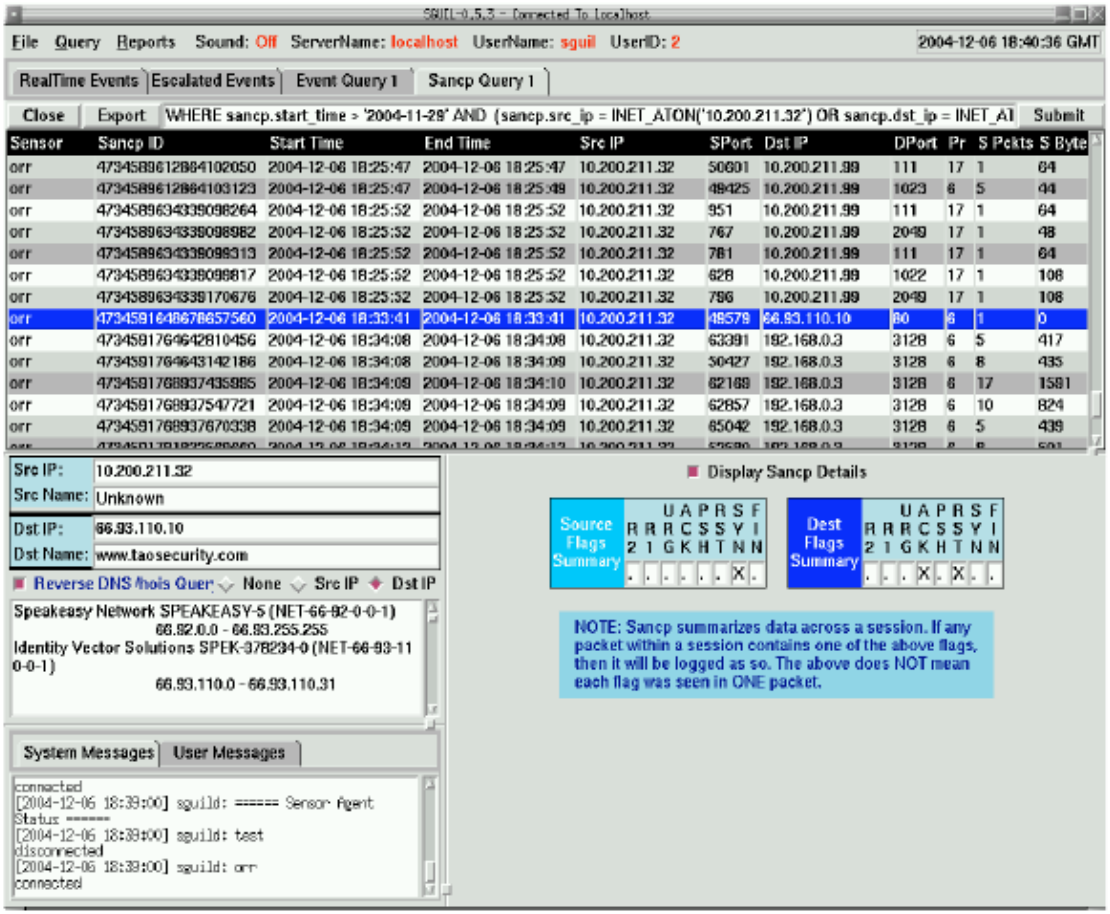
Tool di analisi

ACID (Analysis Console for Intrusion Databases)
http://acidlab.sourceforge.net/ Interfaccia in PHP di analisi dei log di Snort



Tool di analisi

SGUIL (The Analyst Console for Network Security Monitoring) http://sguil.sourceforge.net/index.php Interfaccia per la visualizzazione real-time di alarm generati da Snort



Tool di analisi

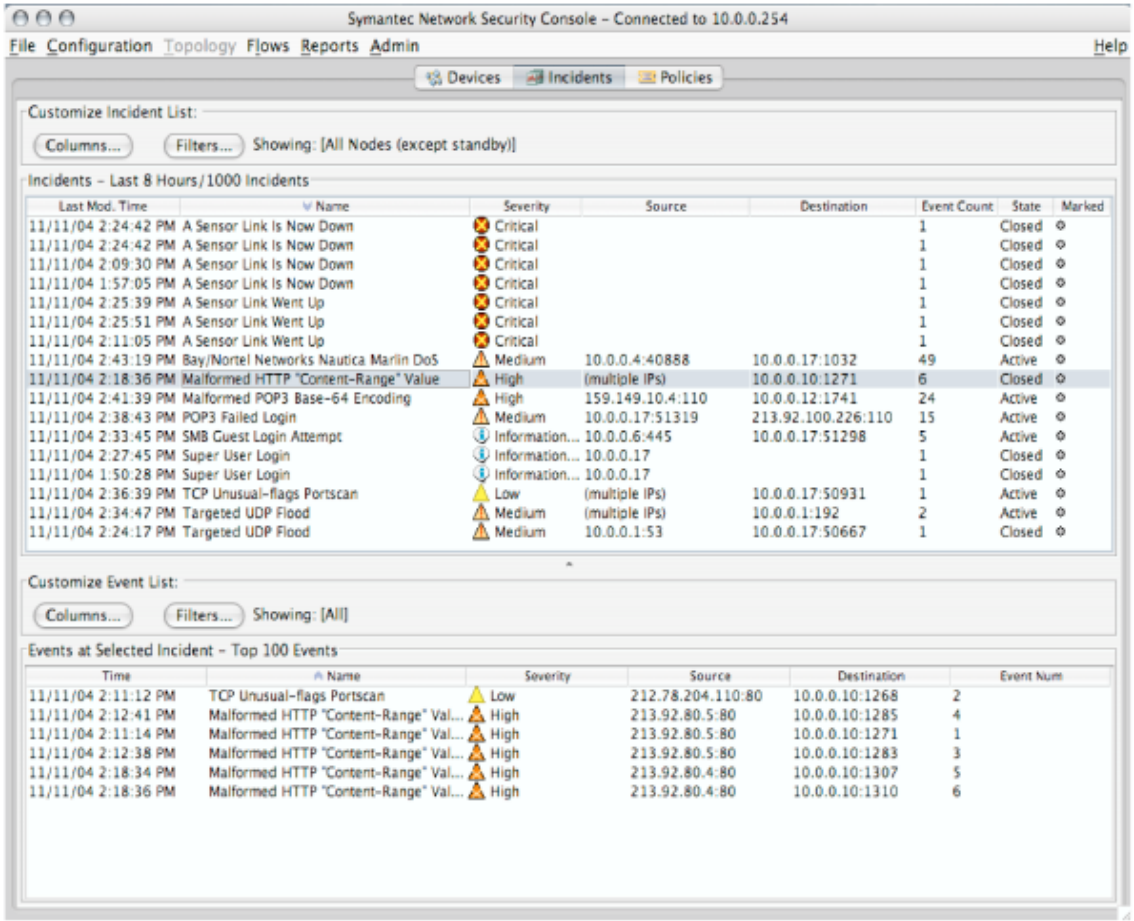
SNORTSNARF

http://www.snort.org/dl/contrib/data_analysis/snortsnarf/

Interfaccia WEB per l'analisi dei log generati da Snort

Tool di analisi

Symantec Network Security 7120 Interfaccia per l'analisi dei log generati dall'appliance



Risposte automatiche

Risposta automatica

Una modalità di allarme che implica la generazione automatica di azioni allo scopo di rispondere attivamente ad una presunta intrusione senza richiedere l'intervento diretto di un operatore.

Risposte automatiche

Esempio Snort:

```
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS
(msg:"WEB-IIS cmd.exe access"; content:"cmd.exe";
react: block; ...)
```

L'opzione `react: block` fa sí che la connessione TCP nella quale si è verificato il tentativo di accesso a `cmd.exe` venga automaticamente terminata

Tipologia di risposte automatiche

Le tecniche piú diffuse sono:

- Reset di sessioni (Session Sniping)
  - L'esempio precedente con Snort è di questo tipo
- Aggiornamento del firewall

Reset di sessioni

Per lo sniping, il NIDS deve essere in grado di forzare la terminazione della connessione

- inviando un pacchetto contenente un RST a entrambi
- devono apparire ai riceventi come inviati dalle controparti

Aggiornamento dei firewall

La rilevazione di un allarme può essere sfruttata per riconfigurare automaticamente le regole di un firewall

- Esempio: la rilevazione di attività di scan viene utilizzata per impedire automaticamente ogni connessione da parte degli indirizzi IP sorgenti coinvolti.

Aggiornamento dei firewall

Meno efficace di quel che potrebbe sembrare:

- Un intrusore può provocare riconfigurazioni che risultano dannose, ad esempio inviando pacchetti con IP spoofed
- Gli effetti possono essere di bloccare le connessioni provenienti da sorgenti legittime (denial-of-service)

Riassumendo

Cosa fare delle segnalazioni dell'IDS

- usare tool di analisi
- interrompere connessioni
- riconfigurare, piú o meno automaticamente, le regole dei firewall

Evasione di IDS

Spesso l'elusione del rilevamento è possibile sfruttando l'uso di alias o altri trucchi che aggirano l'identificazione di una risorsa o di un attacco

Evasione di IDS

Esempio: Una regola che cerchi di verificare la condizione `content:/etc/passwd;` potrebbe essere bypassata da formati equivalenti quali `/etc//\//passwd` oppure `/etc/rc.d/.././\passwd`.
Occorre cercare di riportare la regola all'esame di nomi canonici.

Evasione di IDS

Tecniche di evasione piú sofisticate utilizzano pacchetti frammentati per la loro difficoltà di gestione.
Per esempio, si supponga che il NIDS abbia una finestra per riassemblare i pacchetti frammentati inferiore rispetto al sistema vittima. Il NIDS considererebbe due frammenti come pacchetti indipendenti, il sistema destinatario come pacchetto unico.

Pericoli delle risposte automatiche

Qualunque meccanismo di risposta automatica ha il potenziale difetto di poter essere bypassato e/o sfruttato contro il sistema stesso che viene protetto

Costi delle risposte automatiche

- Le risposte automatiche non sostituiscono l'intervento e l'analisi dell'operatore umano: un apparente risparmio di risorse può risultare in un aggravio di costi
- L'intrusion detection è per sua natura un'attività che necessariamente richiede una forte componente di analisi e di gestione manuale da parte di operatori specializzati (per questo è spesso esternalizzato).

Commenti generali sugli IDS

Un famoso (e controverso) rapporto di Gartner Group del 2003 afferma che gli IDS non valgono gli investimenti richiesti, perché:

- Troppi falsi positivi e negativi
- Richiedono staff dedicato al monitoraggio che dev'essere compiuto 24×7

Commenti generali sugli IDS

- Il processo di risposta agli incidenti è molto oneroso
- Non si riescono a monitorare reti con traffico superiore ai 600MB/s senza inaccettabili decadimenti prestazionali

Commercialmente si è passati al termine IPS (intrusion protection s.), suggerendo cosí di avere a che fare con strumenti piú sofisticati...

Riassumendo

- Le risposte automatiche hanno costi organizzativi e possono risultare strumenti di evasione o attacco
- Il processo di risposta agli incidenti è molto oneroso e richiede staff esperto

Lezione XIII: IDS e attacchi imprevisti

Attacchi imprevisti

I NIDS signature-based si basano sull'assunzione di saper caratterizzare un attacco.

1. Identificare una vulnerabilità: la firma cercherà di rappresentare tutti gli attacchi capaci di sollecitarla;
2. Riconoscere un exploit: la firma cercherà di rappresentare tutte le varianti.

Attacchi imprevisti

Zero day

Un attacco può essere del tutto inatteso: in questo caso si parla di zero-day, ossia il giorno *prima* di quando i NIDS sono in grado di riconoscerlo.

Finestra di vulnerabilità

Il tempo che intercorre fra il momento in cui un attaccante si rende conto di una vulnerabilità e capisce come sfruttarla e il momento in cui l'attacco è identificato dal difensore può essere molto lungo (*vulnerability window*).

Nel 2008 Microsoft ha reso nota una vulnerabilità di IE presente dal 2001, quindi con una finestra potenzialmente di 7 anni!

Ricerca delle vulnerabilità

La ricerca delle vulnerabilità non note è una delle attività dei "laboratori di sicurezza".

- Si cercano vulnerabilità generiche (non di una rete specifica): si analizzano applicazioni e protocolli
- Gli *zero-day* hanno un mercato (non solo underground!)

Tecniche per la ricerca

studio analitico si studiano le specifiche piú o meno formali di applicazioni e protocolli

fuzzing si provano le applicazioni (o i protocolli) con input "strani" casuali

honeypot un sistema che viene realizzato e messo in opera solo come bersaglio

Polimorfismo degli attacchi

La medesima vulnerabilità può essere sfruttata da exploit con forme diverse: gli attacchi hanno quindi natura polimorfica.
In generale è impossibile prevedere tutte le possibili varianti e costruire le firme che permettano di rilevarli.
Una forma completamente nuova è analoga a uno zero-day, anche se la vulnerabilità è già nota.

Cifratura

Una delle tecniche piú diffuse è la cifratura.

- Viene generata per ogni attacco una chiave casuale
- il *payload* dell'attacco viene cifrato, apparendo cosí sempre diverso
- l'unica parte di codice costante è una piccola routine di decifratura (possono bastare 3-4 istruzioni: p.es. cifratura XOR)
- anche la routine di decifratura può essere variata con ulteriori tecniche di polimorfismo

Dead-code insertion

dead-code insertion o trash insertion: aggiungere codice senza modificare il comportamento.

- La tecnica piú semplice è inserire `nop`
- Metodi piú sofisticati fanno uso di sequenze di codice che si annullano vicendevolmente

La ricerca di stringhe costanti fallisce.

```
call 0h
pop ebx
lea ecx, [ebx + 45h]
nop
nop
push ecx
push eax
inc eax
push eax
dec [esp - 0h]
dec eax
sidt [esp - 02h]
pop ebx
add ebx, 1Ch
cli
mov ebp, [ebx]
```

Code transposition

Sposta le istruzioni in modo che l'ordine del codice binario sia differente dall'ordine di esecuzione

- riordinando casualmente blocchi di istruzioni e inserendo salti incondizionati (facile da fare automaticamente)
- mischiando istruzioni indipendenti (richiede analisi sofisticate del codice)

```
call 0h
pop ebx
jmp Step2
Step3: push eax
push eax
sidt [esp - 02h]
jmp Step4
add ebx, 1Ch
jmp Step6
Step2: lea ecx, [ebx + 45h]
push ecx
jmp Step3
Step4: pop ebx
cli
jmp Step5
Step5: mov ebp, [ebx]
```

Instruction substitution e register reassignment

- instruction substitution: dizionari di sequenze di istruzioni equivalenti, che possono essere sostituite tra loro.
- register reassignment sostituisce l'uso di un registro con un altro equivalente.

```
call 0h
pop ebx
lea ecx, [ebx + 42h]
sub esp, 03h
sidt [esp - 02h]
add [esp], 1Ch
mov ebx, [esp]
inc esp
cli
mov ebp, [ebx]
```

Riassumendo

Gli IDS misuse-based necessitano di *firme* degli attacchi:

- A volte non sono ancora note
- È difficile prevedere le varianti introdotte con tecniche di polimorfismo

Generatori di signature

L'idea di base è che grazie alla conoscenza di vulnerabilità e di un certo numero di exploit, si vogliono generare automaticamente signature utili a bloccare exploit non ancora rilevati "in the wild".

Generatori di signature

semantic-based modellano il comportamento di un attacco: se la rilevazione richiede l'interpretazione del modello, può essere molto dispendiosa.

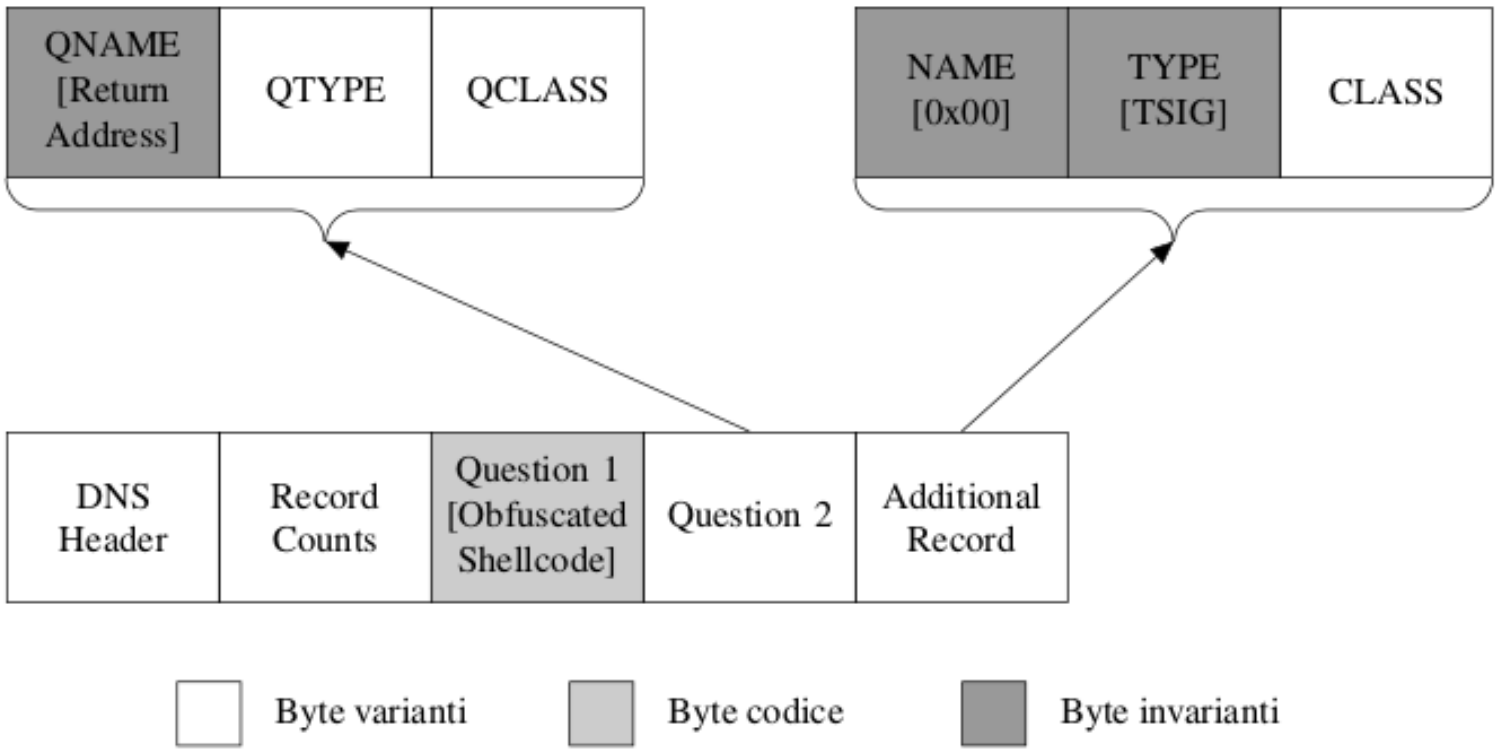
content-based si basano sulla ricerca di invarianti: in realtà è piuttosto raro che la parte invariante di un exploit sia sufficientemente ricca per limitare i falsi positivi.

Hamsa

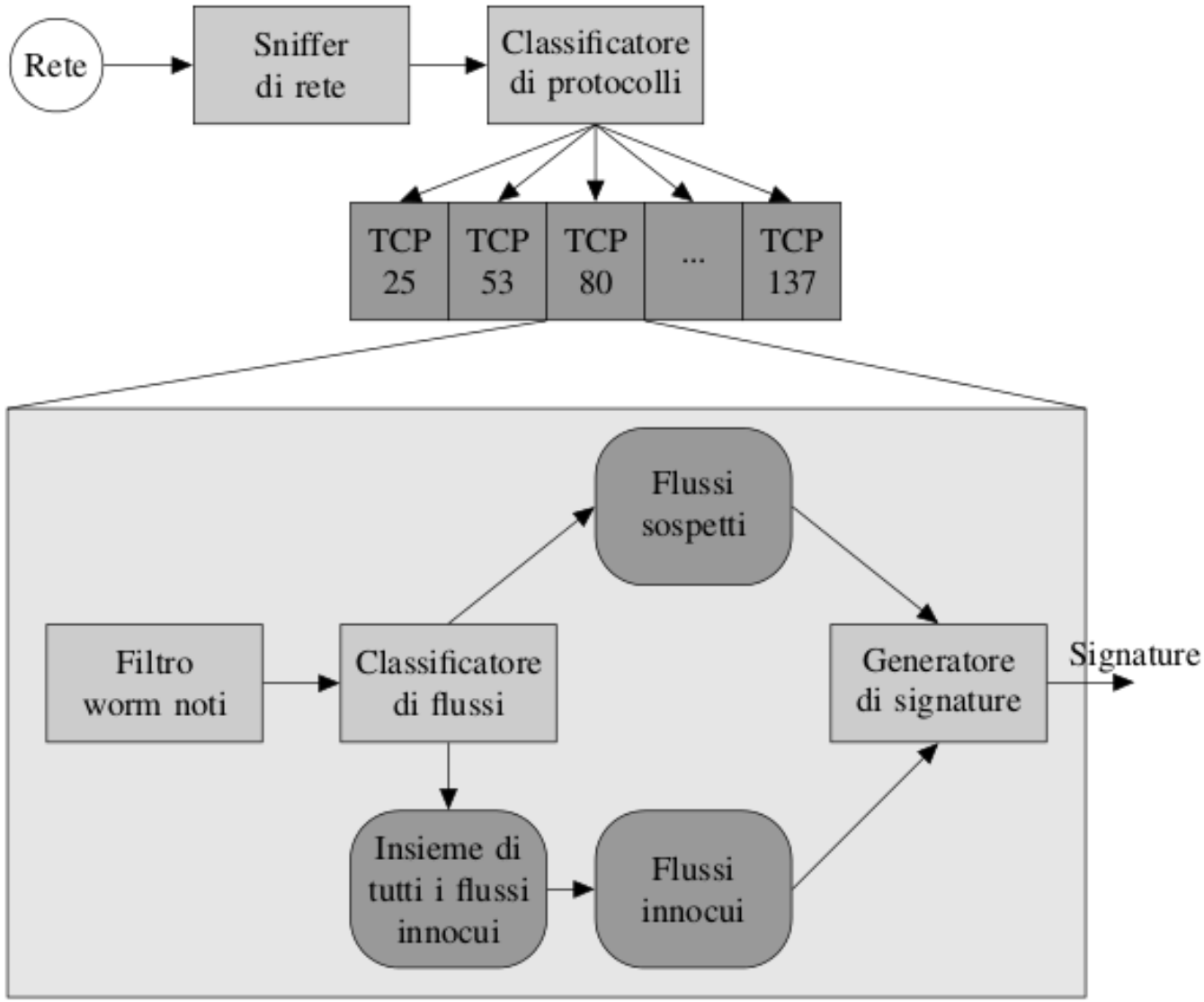
- Zhichun Li, Manan Sanghi, Yan Chen, Ming-Yang Kao and Brian Chavez. Hamsa: Fast Signature Generation for Zero-day Polymorphic Worms with Provable Attack Resilience. IEEE Symposium on Security and Privacy, Oakland, CA, maggio 2006.
- Utilizzabile a livello di rete (gateway e router)
- *content-based*:

invarianti	byte il cui valore e fissato a priori e la cui variazione implica il fallimento dell'attacco
code byte	parte potenzialmente polimorfica, ma con una semantica fissa
wildcard byte	possono assumere qualsiasi valore

Esempio: Lion worm



Architettura di Hamsa



Architettura di Hamsa

classificatore di protocolli considera i flusso TCP (o i pacchetti UDP) e li classifica secondo la porta destinazione

politica di selezione indica quali flussi prelevare e inviare al generatore di signature

generatore di signature genera le signature, considerando i flussi innocui e sospetti

Firme Hamsa

- Sono dette conjuction signature: consiste in un insieme di stringhe e un flusso viene considerato malevolo se contiene tutte le stringhe, indipendentemente dall'ordine.
- Si tratta in realtà di *multi-insiemi* di token, cioè insiemi in cui un elemento può apparire piú volte.

Firme Hamsa

Code-Red II	{'.ida?':1, '%u780':1, ' HTTP/1.0\r\n':1,'GET /':1, '%u':2}
ATPhttpd	{'\x9e\xf8':1, ' HTTP/1.1\r\n':1, 'GET /': 1}

I token indicati devono comparire in un unico flusso e con un numero di occorrenze maggiore o uguale a quello indicato.

Algoritmo di generazione delle firme

Input: Insieme degli invarianti I, insieme dei flussi malevoli M e dei flussi innocui N, vettore u dei falsi positivi massimi
Output: Signature S per un worm presente in M

```
S = creaSignatureVuota()
SignatureCandidata = S
VettoreSignature = []
i = 1
while i < k do
    foreach t ∈ I do
        S = S.aggiungi(t)
        FP = calcolaFalsiPositivi(S, N)
        if FP < u[i] then
            TP = calcolaVeriPositivi(S, M)
            if SignatureCandidata.TP < TP then
                SignatureCandidata = S
            end
        end
        S = S.rimuovi(t)
    end
    if SignatureCandidata == creaSignatureVuota() then
        break
    end
    VettoreSignature.appendi(SignatureCandidata)
    S = SignatureCandidata
    SignatureCandidata = creaSignatureVuota()
    i = i + 1
end
foreach S ∈ VettoreSignature do
    calcolaPunteggio(S)
end
return S con punteggio massimo
```

- k è il numero di token in $\mathcal{I}$
- $\text{calcolaPunteggio}(\mathcal{S}) = -log_{10}(\delta + FP_{\mathcal{S}}) + a \cdot TP_{\mathcal{S}} + b \cdot \text{lunghezza}(\mathcal{S})$
- tutti i parametri sono scelti in modo empirico (anche per la classificazione innocuo/sospetto)
- $u(i) = u(1) \cdot {u_r}^{(i-1)}$ con $u(1) = 0,15$ e $u_r = 0,5$

Attacchi a Hamsa

Target feature manipulation Si cerca di variare le parti considerate invarianti

Innocuous pool poisoning prima di iniziare la diffusione vera e propria e quindi prima di lanciare un attacco verso una nuova macchina, ci si preoccupa di inviare una serie di pacchetti leciti contenenti ciascuno un invariante inserito nelle parti di traffico che possono essere modificate a piacere.

Suspicious pool poisoning l'attaccante incorpora finti invarianti all'interno dei flussi malevoli per portare alla generazione di signature che dipendono da tali finti invarianti al posto o in aggiunta agli invarianti veramente necessari.

Riassumendo

Generare automaticamente le varianti di un attacco:

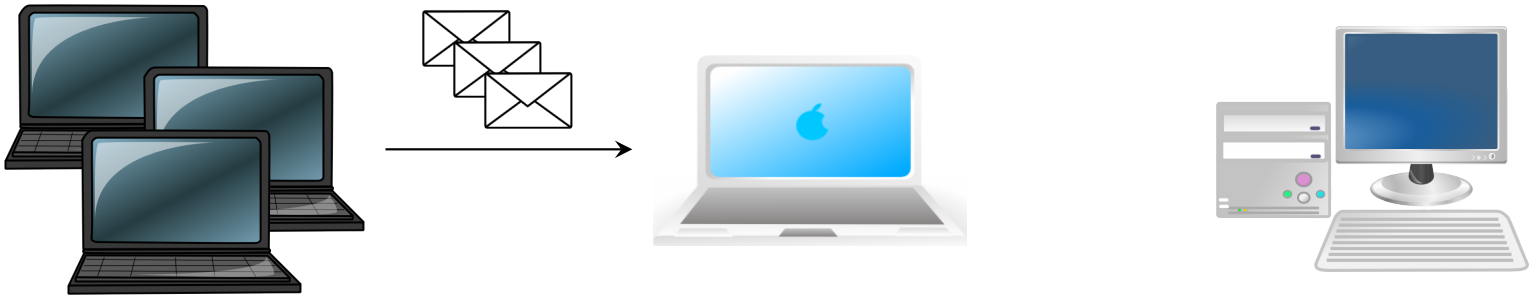
- È un'operazione con fortissime connotazioni empiriche (in generale è un obiettivo irrealizzabile)
- Come sempre, un meccanismo automatico può essere sfruttato anche dall'attaccante (*poisoning*)

Malware undeground economy

Il malware viene diffuso sfruttando vulnerabilità generiche allo scopo di compiere attacchi piú redditizi.

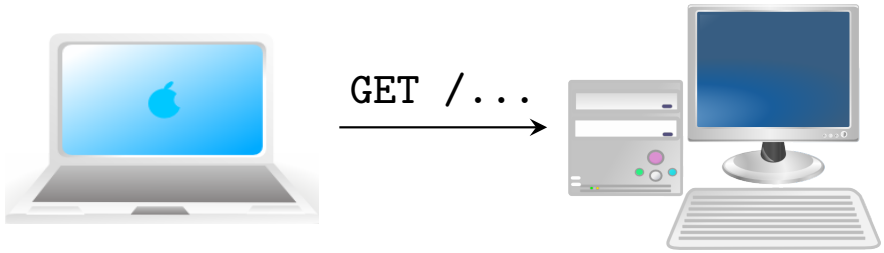
Phishing

1. campagna di spam



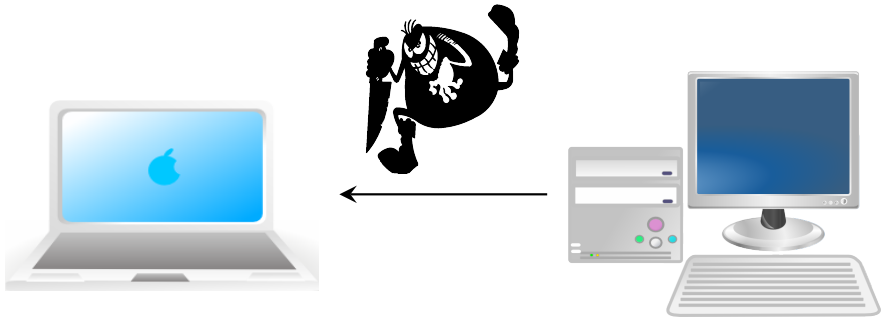
Phishing

1. campagna di spam
2. social engineering



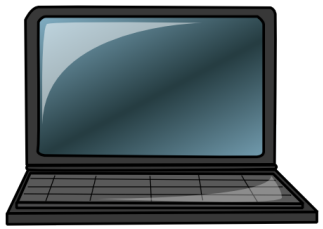
Phishing

1. campagna di spam
2. social engineering
3. furto credenziali & malware



Phishing

1. campagna di spam
2. social engineering
3. furto credenziali & malware
4. infezione macchine



Underground economy
Vendita informazioni rubate

Goods & services	Percentage	Range of prices
Bank accounts	22%	$10-$1000
Credit cards	13%	$0.40-$20
Full identities	9%	$1-$15
Online auction site accounts	7%	$1-$8
Scams	7%	$2.50-$50/week (hosting)
Mailers	6%	$1-$10
Email addresses	5%	$0.83/MB-$10/MB
Email passwords	5%	$4-$30
Drop (request or offer)	5%	10%-20% of drop amount
Proxies	5%	$1.50-$30

Symantec

Underground economy
Furto credenziali — Portata del fenomeno

- Università di Mannheim — Limbo & ZeuS
- $\sim$ 70 dropzone
- **33 GB** di dati
- 11000 account bancari, 150000 account mail

Underground economy

Dropzone	# Machines	Data amount	Country
webpinkXXX.cn	26,150	1.5 GB	China
coXXX-google.cn	12,460	1.2 GB	Malaysia
77.XXX.159.202	10,394	503 MB	Russia
finXXXonline.com	6,932	438 MB	Estonia
Other	108,122	24.4 GB	
Total	164,058	28.0 GB	

Learning More About the Underground Economy — T. Holz, M. Engelberth, F. Freiling, 2008

Underground economy

Malware as a service

- Bot in affitto ($\sim$ \$1000–\$2000/mese)
- MPACK: exploit toolkit a $\sim$ \$1000

CAPTCHA?

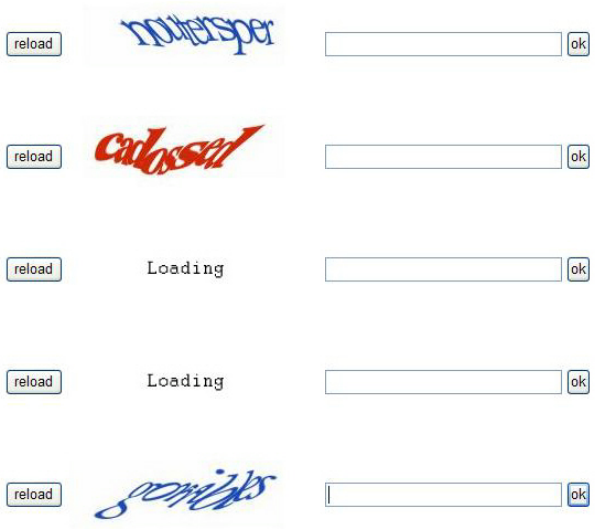
- OCR, Fuzzy OCR, . . .

> 100K captcha al giorno, $1.5–$8 per 1000 captcha

Underground economy
The spam business

CAPTCHA?

- OCR, Fuzzy OCR, ...
- "Human computation"!

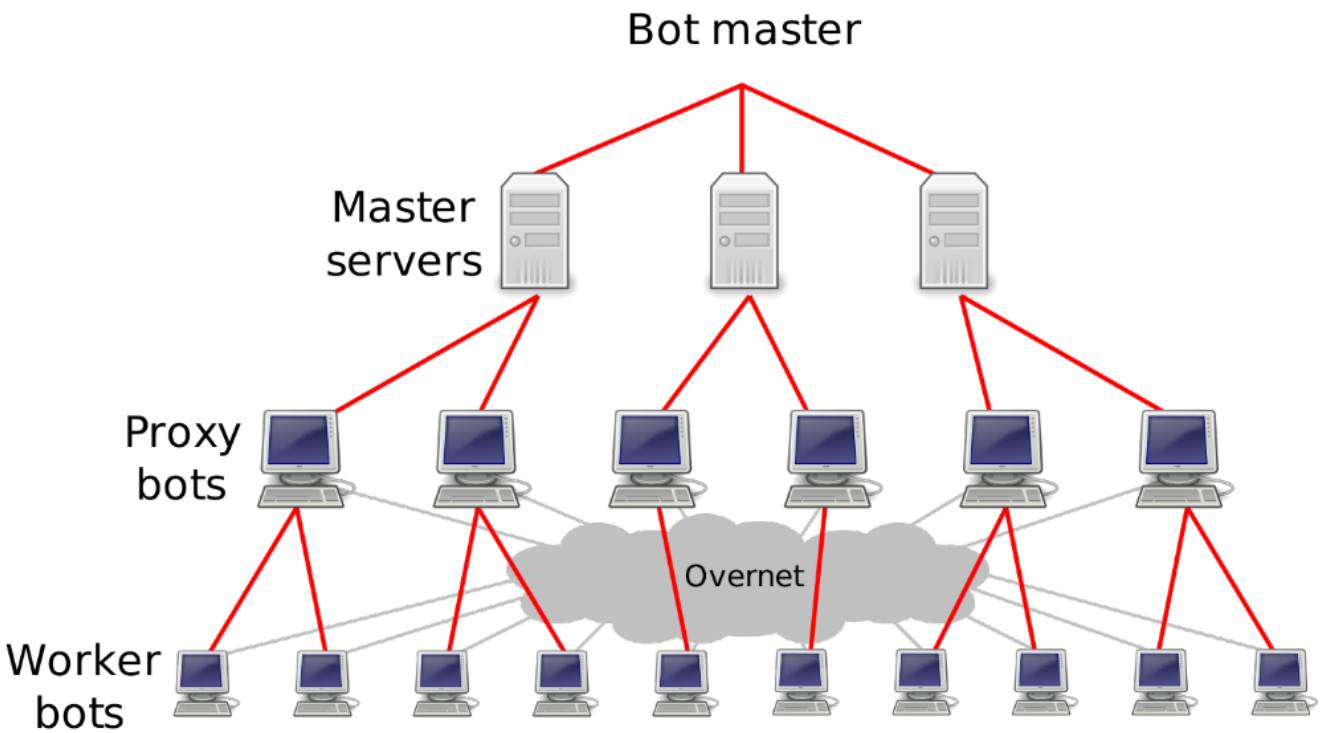


> 100K captcha al giorno, $1.5–$8 per 1000 captcha

Funzionalità del malware
Click fraud

- *Google*: 10% dei *click* sono fraudolenti ($\sim$ \$1B)
- Clickbot.A ($\sim$ 50k host infetti)
- molti "clickbot" commerciali
- ClickJacking

Funzionalità del malware
Botnet



Botnet
Botnet & spam

Nome	Dimensione	Capacità di spam
Conficker	9.000.000	10G/giorno
Kraken	495.000	9G/giorno
Srizbi	450.000	60G/giorno
Rustock	150.000	30G/giorno
Cutwail	125.000	16G/giorno
Storm	$>$ 1.000.000	3G/giorno
Grum	50.000	2G/giorno
Mega-D	35.000	10G/giorno

Riassumendo

- Fermare la diffusione del malware è importante perché è la linfa di un'economia underground piuttosto ampia
- Anche se il danno al singolo target è limitato, può avere effetti molto negativi sull'ecosistema.

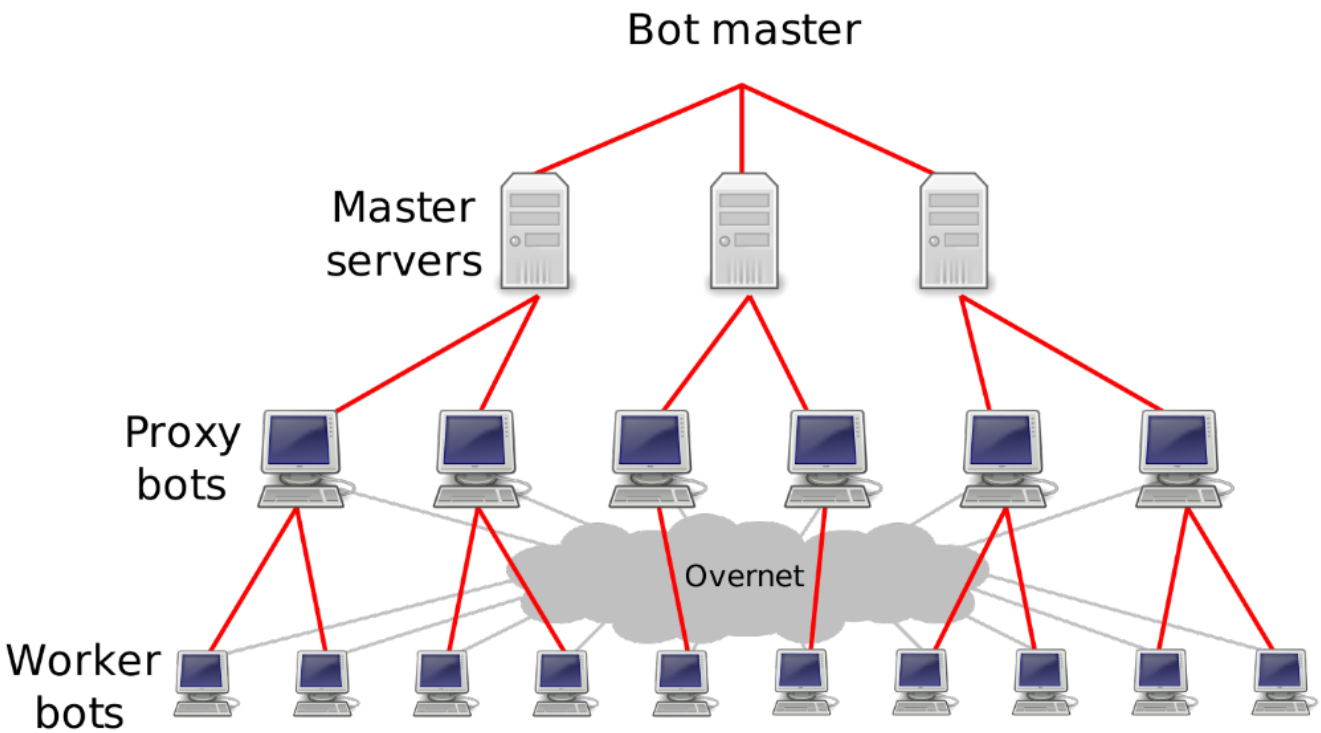
Lezione XIV: Botnet Fast-Flux

Botnet

Botnet

- una rete di macchine infette (bot, zombie) controllate da un unico attaccante (bot-master, mother-ship)
- usate per: spam, DDoS, phishing, scam, SQL injection massivi, ...

Botnet



Botnet
Non solo spam...

Analisi di 10 giorni di traffico di rete generato da Torpig:

Unique IP Count	1.148.264
Unique Torpig keys (machines)	180.835
POP accounts	415.206
Email addresses	1.235.122
Passwords	411.039
Unique credit cards	875
Unique ATM pins	141
Unique social security numbers	21

Tecniche di propagazione

Propagation mechanisms	Percentage
File sharing executables	40%
File transfer/email attachment	32%
File transfer/CIFS	28%
File sharing/P2P	19%
Remotely exploitable vulnerability	17%
SQL	3%
Back door/Kuang2	3%
Back door/SubSeven	3%
File transfer/embedded HTTP URI/Yahoo! Messenger	2%
Web	1%

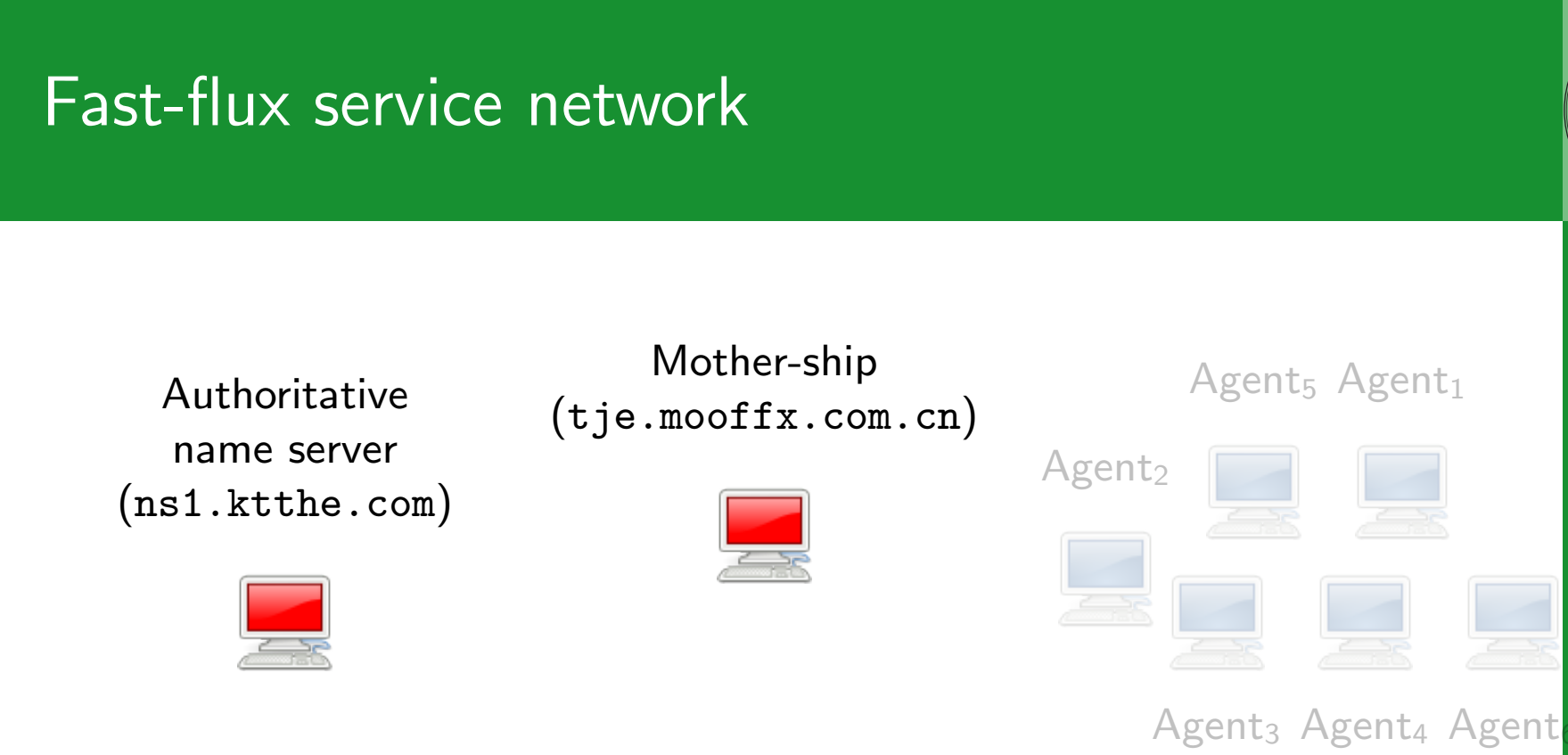
Symantec, 2007

Botnet Fast-flux

Fast-flux service network

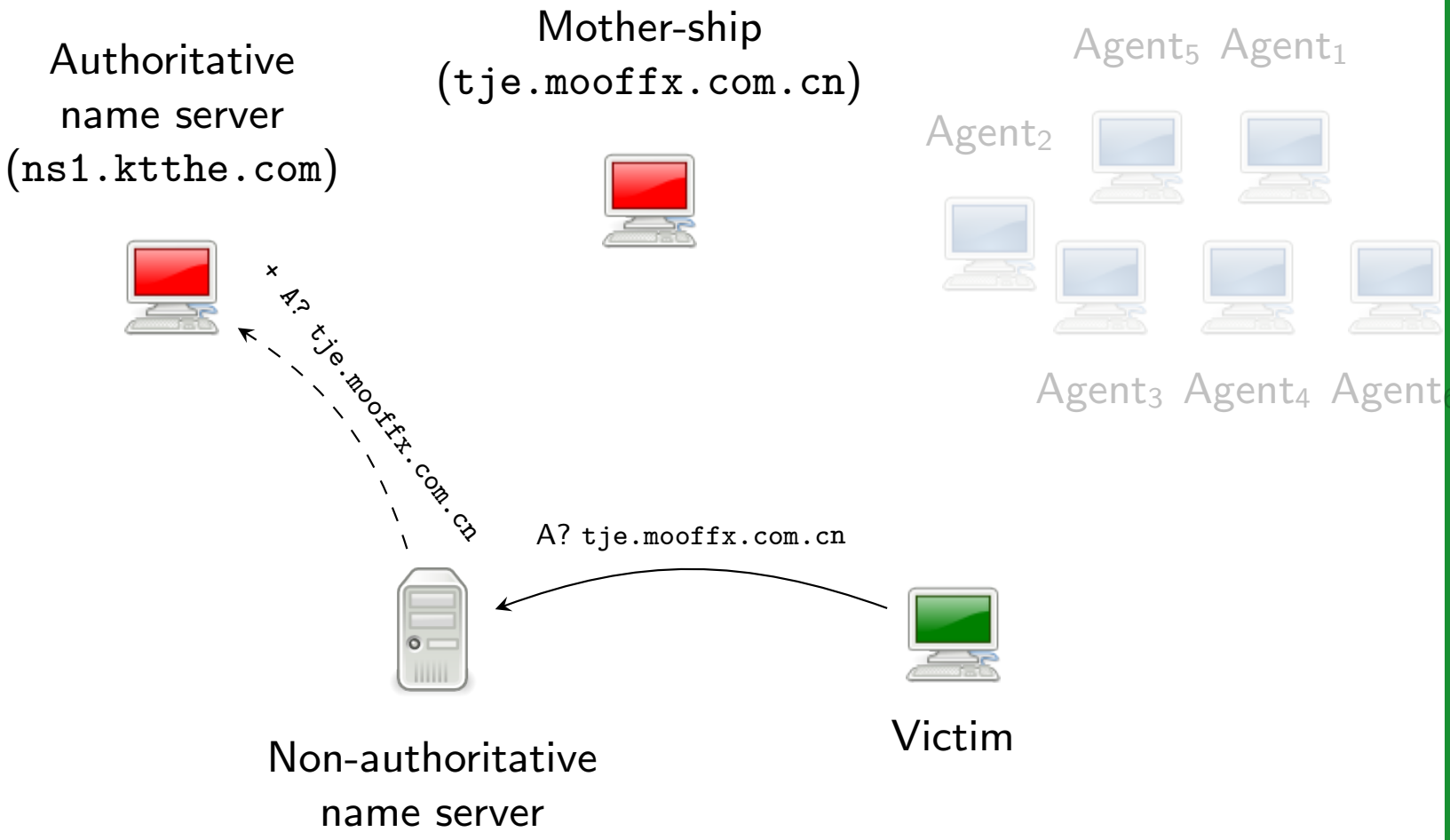
- una tecnica ($\sim$ 2007) utilizzata per aumentare la robustezza della botnet, rendendola piú difficile da identificare.
- l'idea è semplice: si aggiunge un livello di indirettezza fra vittime e attaccante.

Fast-flux service network

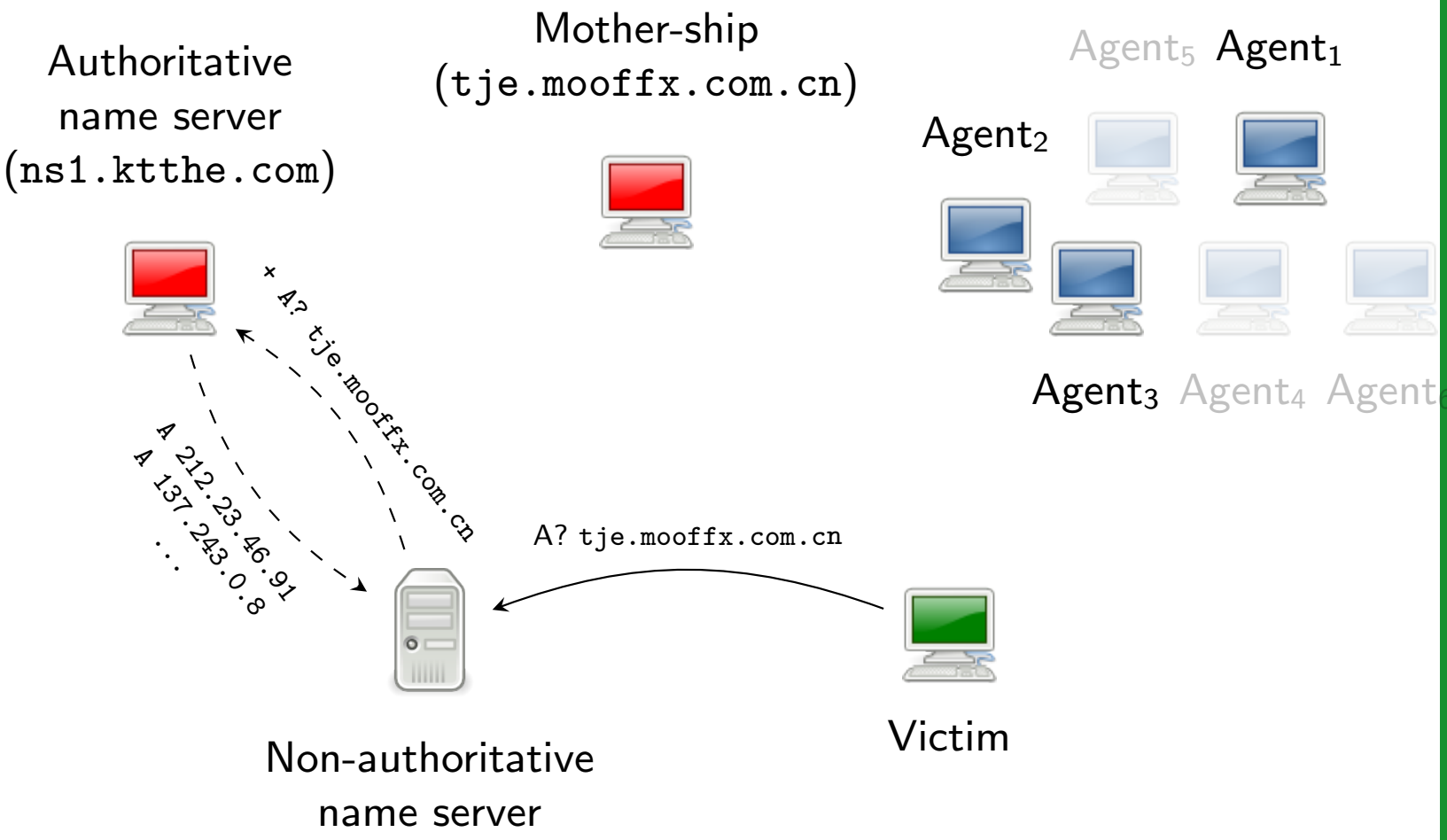


Fast-flux service network

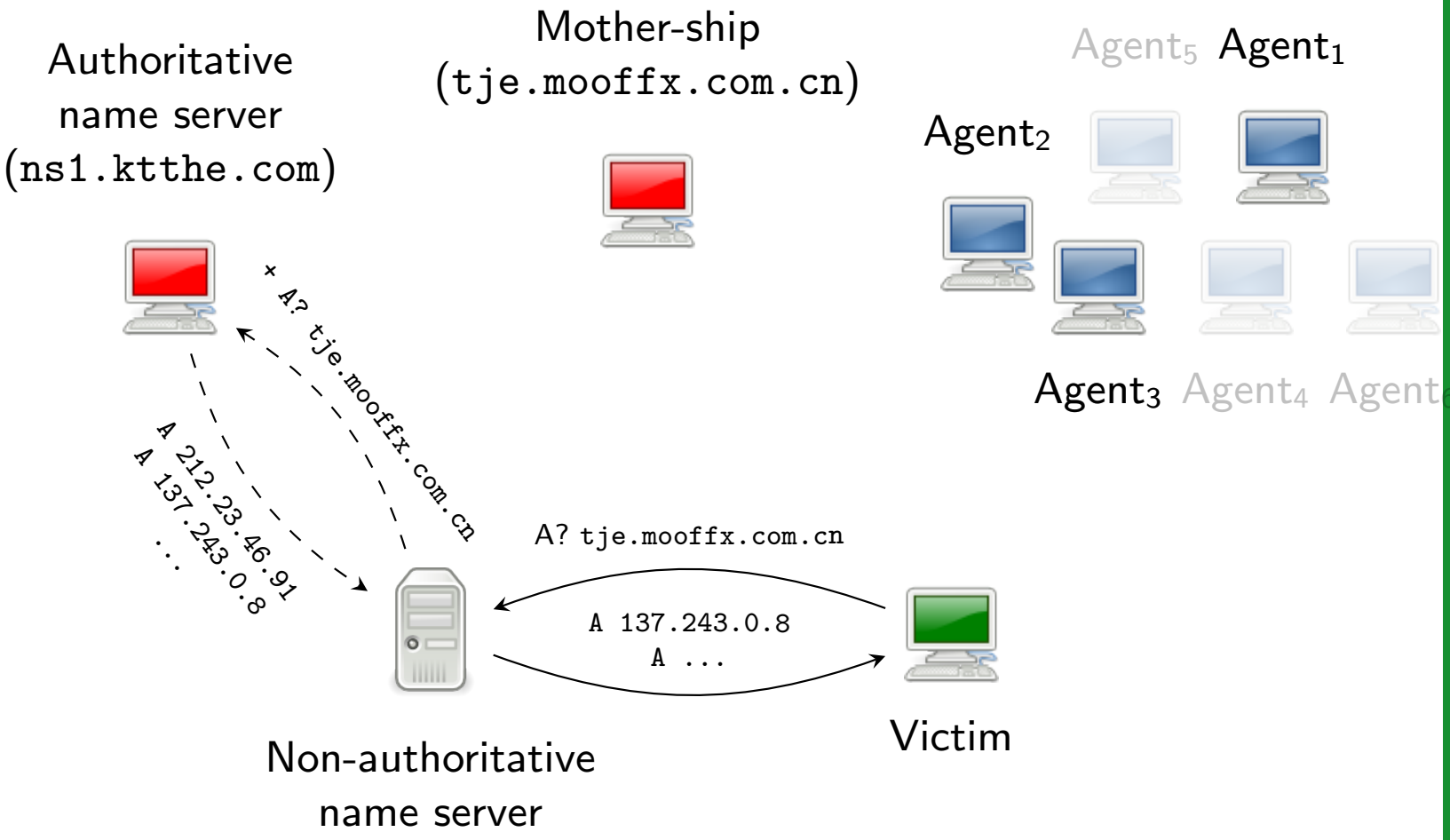
Fast-flux service network



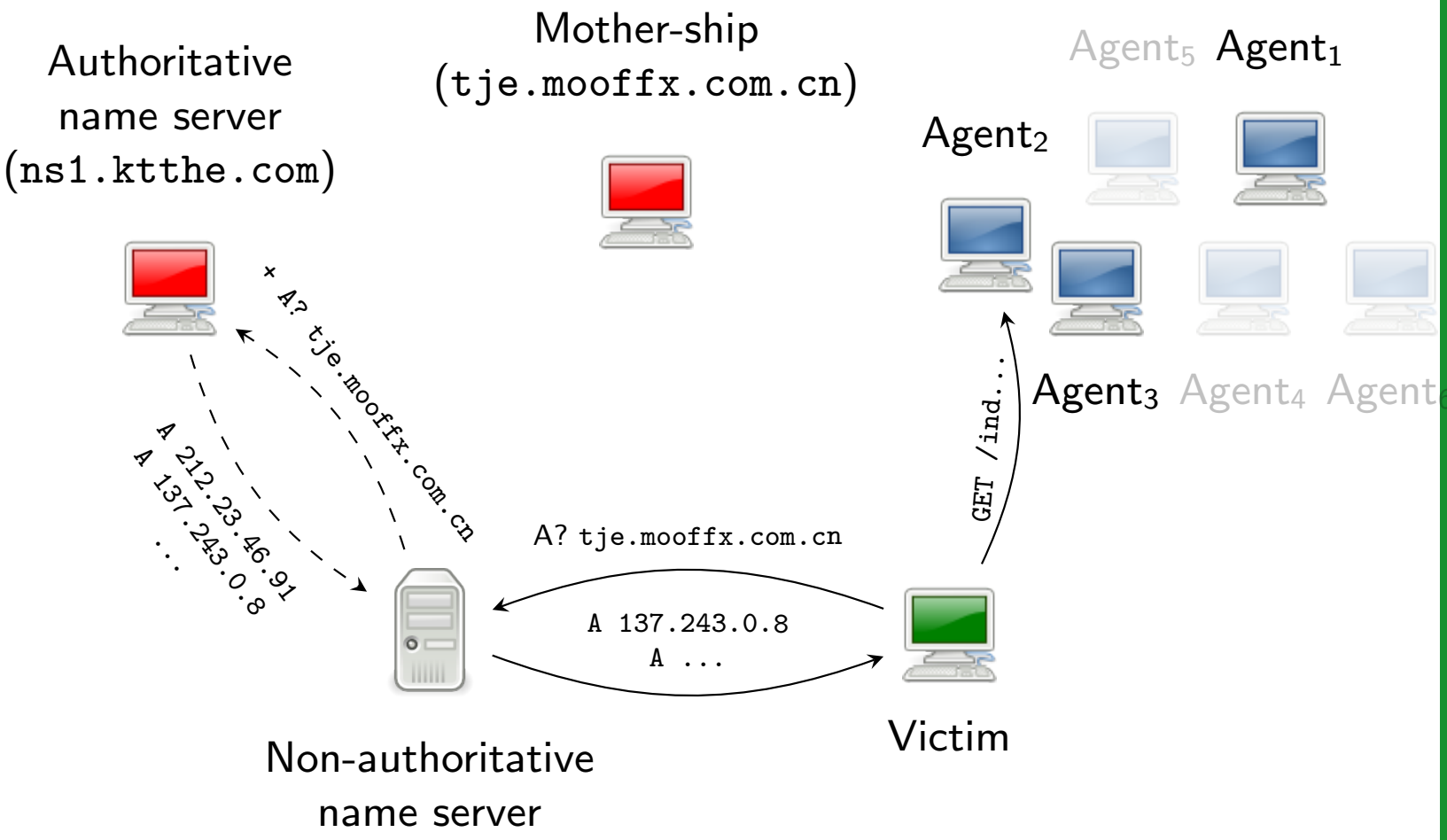
Fast-flux service network



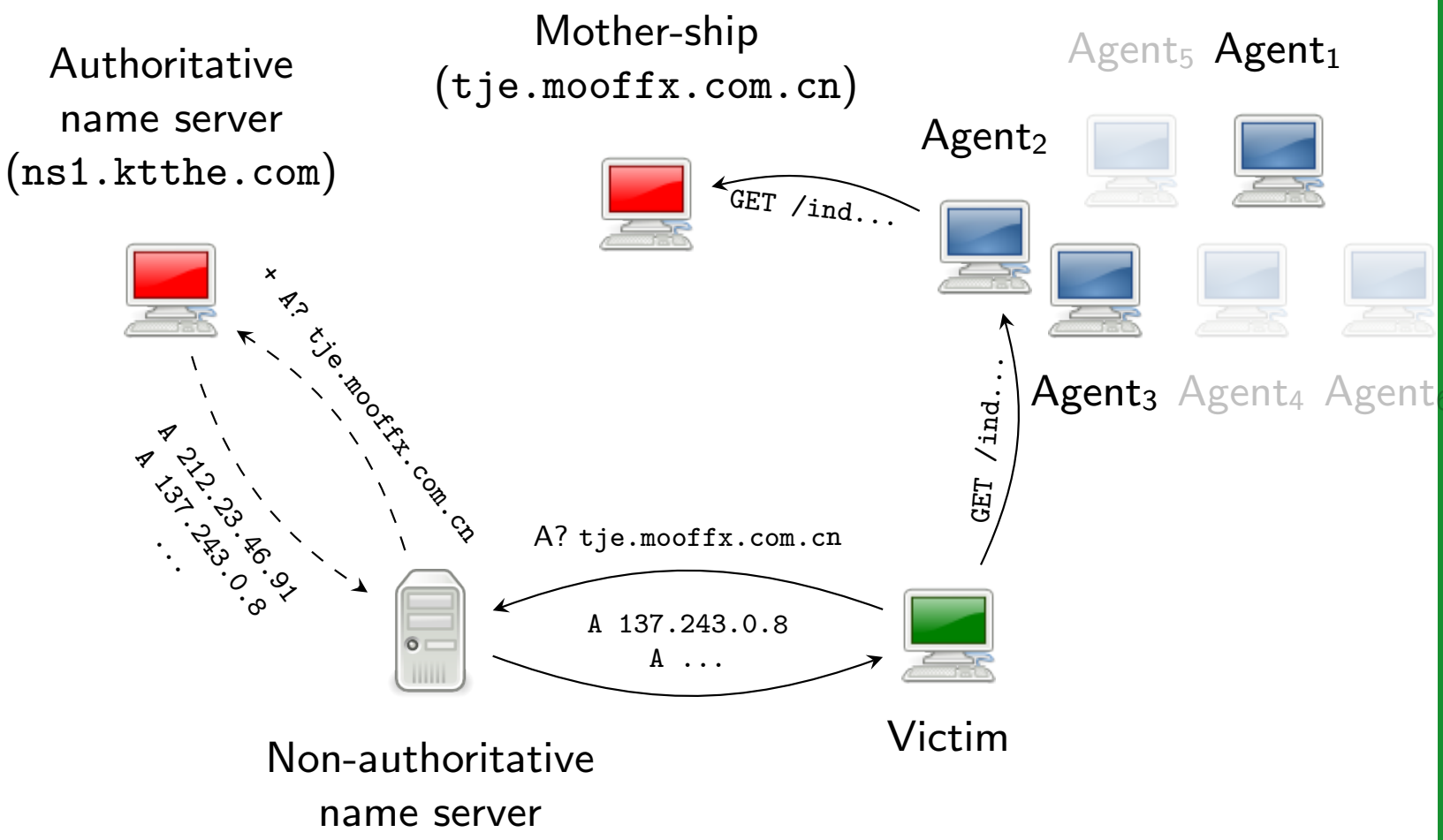
Fast-flux service network



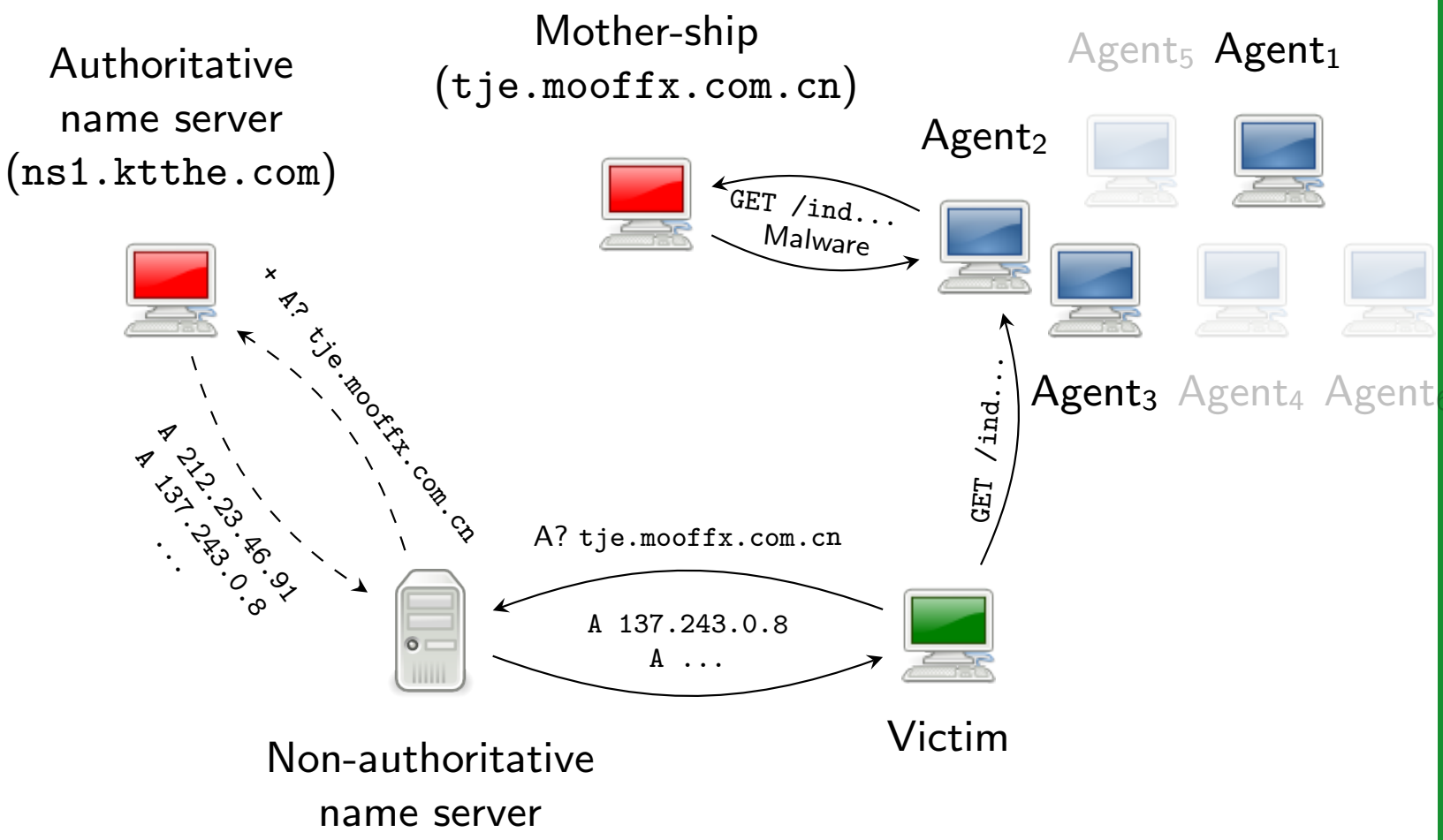
Fast-flux service network



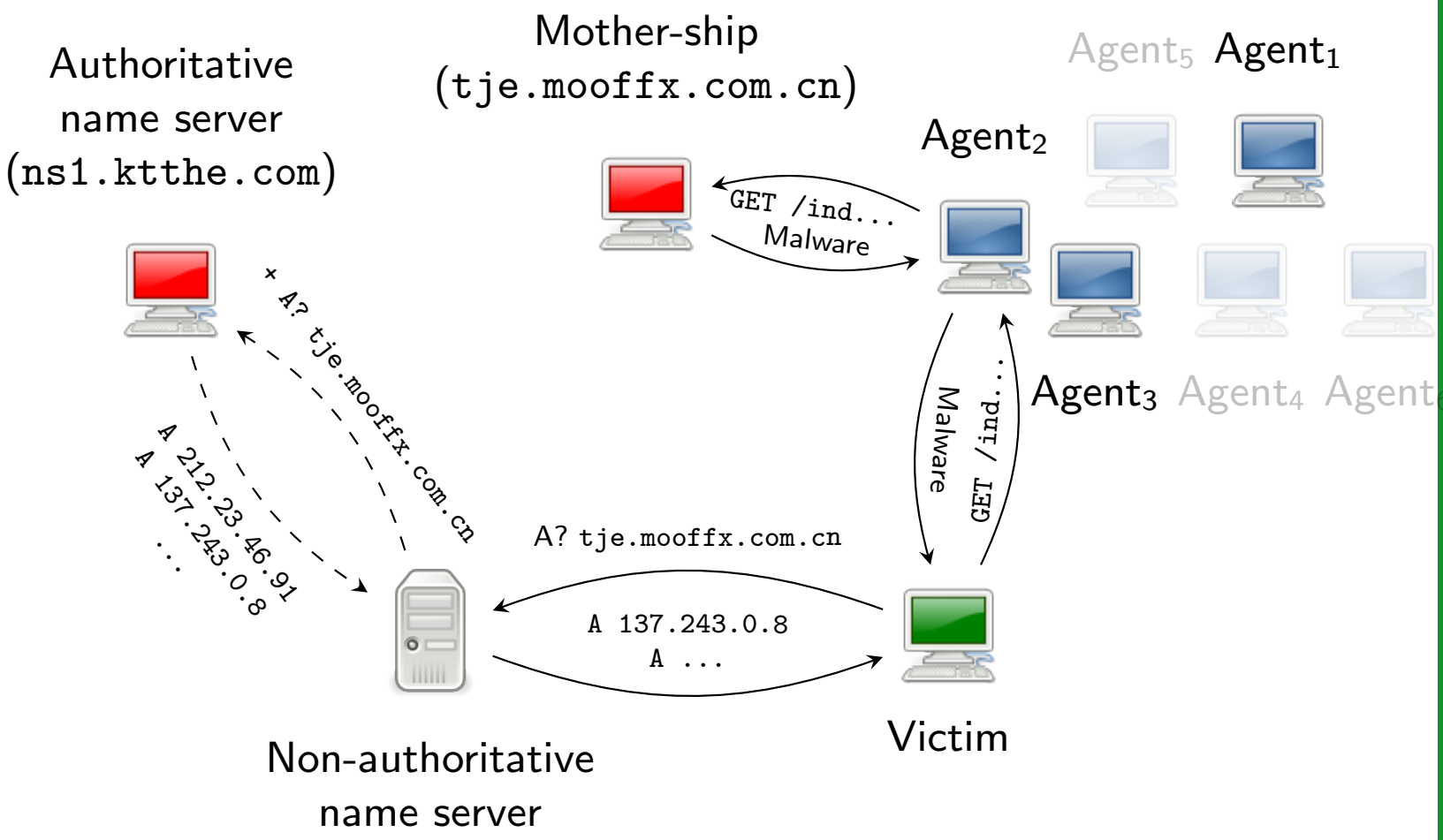
Fast-flux service network



Fast-flux service network



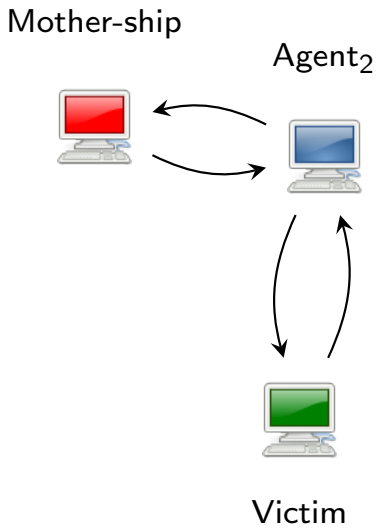
Fast-flux service network



Fast-flux service network

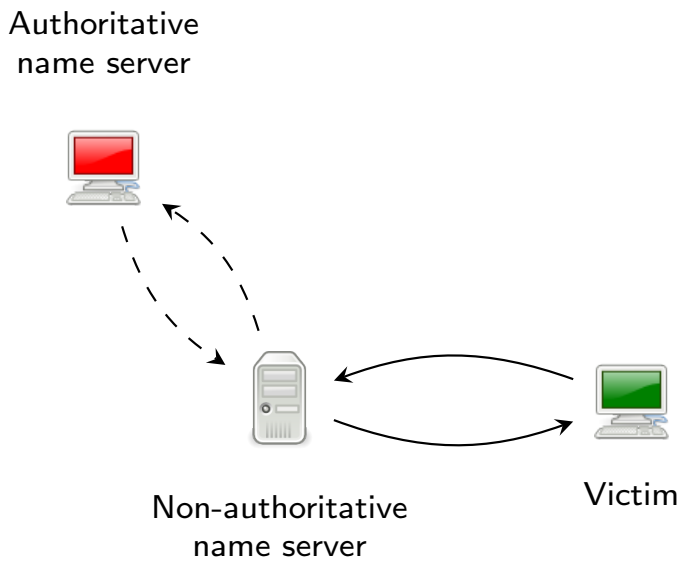
- I bot offline, disinfettati o problematici vengono immediatamente rimpiazzati da altri
- Composta da milioni di agenti (Nostri esperimenti: ∼121.000 fast-flux FQDN ∼360.000 host)
- Piú domini vengono utilizzati dalla stessa botnet (non basta chiudere un dominio)

Fast-flux service network



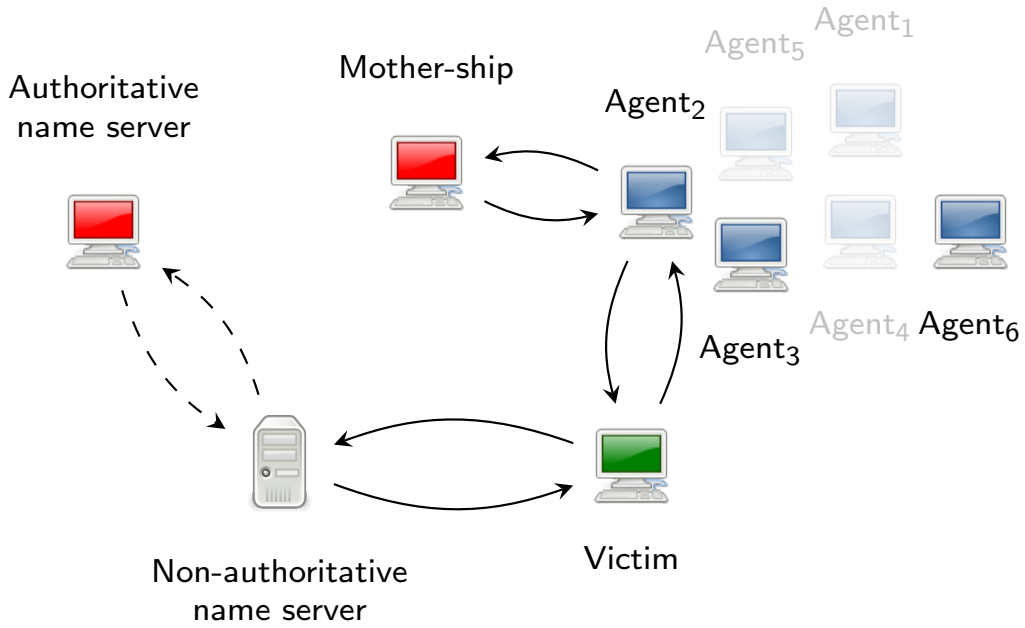
- I bot offline, disinfettati o problematici vengono immediatamente rimpiazzati da altri
- Composta da milioni di agenti (Nostri esperimenti: ∼121.000 fast-flux FQDN ∼360.000 host)
- Piú domini vengono utilizzati dalla stessa botnet (non basta chiudere un dominio)

Fast-flux service network



- I bot offline, disinfettati o problematici vengono immediatamente rimpiazzati da altri
- Composta da milioni di agenti (Nostri esperimenti: ∼121.000 fast-flux FQDN ∼360.000 host)
- Piú domini vengono utilizzati dalla stessa botnet (non basta chiudere un dominio)

Fast-flux service network



- I bot offline, disinfettati o problematici vengono immediatamente rimpiazzati da altri
- Composta da milioni di agenti (Nostri esperimenti: ∼121.000 fast-flux FQDN ∼360.000 host)
- Piú domini vengono utilizzati dalla stessa botnet (non basta chiudere un dominio)

Fast-Flux @ Unimi

Come identificare una FFSN?

- Ci sono moltissime caratteristiche misurabili...
- ...ma nessuna è sufficiente per identificare una FFSN

Fast-Flux @ Unimi

FluXOR

- si monitora un hostname sospetto, fingendosi una vittima
- si raccolgono dati e si identificano le FFSN tramite classificazione complessa
- si tengono sotto controllo le FFSN per elencare il maggior numero di agenti infetti

Features of fast-flux service network

- Domain
  - Domain age
- Availability
  - # of DNS records of type `A`
  - TTL of DNS records
- Heterogeneity
  - # of networks
  - # of autonomous systems
  - # of resolved QDNs
  - # of assigned network names
  - # of organisations

Benign			
`avast.com`	539	12	3600
`adriaticobishkek.com`	65	21	1200
`google.com`	542	3	300
mean	**493.27**	**2.86**	**4592.53**
std. dev.	**289.27**	**3.89**	**7668.74**
Malicious			
`eveningher.com`	18	127	300
`factvillage.com`	2	117	300
`doacasino.com`	2	33	180
mean	**4.85**	**98.13**	**261.49**
std. dev.	**4.9**	**37.27**	**59.64**

Architettura del sistema

Collector

Raccoglie nomi di dominio sospetti da sonde informative (e.g, spam . . .)

Architettura del sistema

Monitor

- per ogni DN sospetto raccoglie info sulle caratteristiche
- per ogni DN malevolo (classificato dal Detector) raccoglie gli IP degli agenti infetti

Architettura del sistema

Detector

- classifica i sospetti in malevoli e benevoli tramite un classificatore bayesiano
- Training set di partenza: 50 benevoli + 58 malevoli classificati manualmente

Fast-flux service network e truffe via web

Descrizione	*#*
Email processate	144952
URL estratti	34466
FQDN attivi	29368
Fast-flux service network	*9988*
Agenti Fast-flux	*162855*
Botnet Fast-flux	*25*

Fast-flux service network e truffe via web

Botnet	*# agenti*	*# FFSN*
European Pharmacy	65043	3950
Halifax Online Banking	46772	1
Digital Shop	20069	17
Royal Casino	15078	34
Royal VIP Casino	8665	16
Euro Dice Casino	7667	28

Fast-flux service network e truffe via web

Botnet	*# spam email*	*% spam (rispetto al totale)*
European Pharmacy	12056	8.32%
SwissWatchesDirect	3330	2.30%
RXNET	2558	1.76%
MaxHerbal	1897	1.31%
Altre FFSN	*6395*	*4.41%*
Totale	*144952*	*18.10%*

Riassumendo

Le botnet Fast-flux

- nascondono la propria topologia grazie a registrar "compiacenti"
- sono rilevabili con tecniche di data mining

Protezione dei servizi critici

L'accesso ai servizi critici è controllato

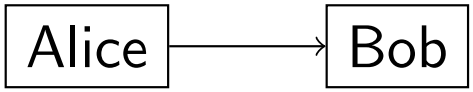
- **Autenticazione:** chi è l'agente (che opera in nome di un *principal*
- **Autorizzazione:** l'agente autenticato ha il permesso?

Autenticazione

Autenticazione

Autenticare significa verificare l'identità di un soggetto (non necessariamente umano)

Modalità di autenticazione



Modalità di base per l'autenticazione (di Alice) tramite rete:

1. password (ossia la conoscenza di un segreto)
2. locazione (logica o fisica) da cui proviene la richiesta di autenticazione
3. per mezzo di operazioni crittografiche su dati forniti dall'autenticatore (Bob).

Pericoli

Alcune vulnerabilità sono intrinseche:

- Le password possono essere indovinate
- Le locazioni possono essere millantate
- I dati crittografici possono essere intercettati e riutilizzati (replay attack)

Difese

Queste minacce possono essere mitigate

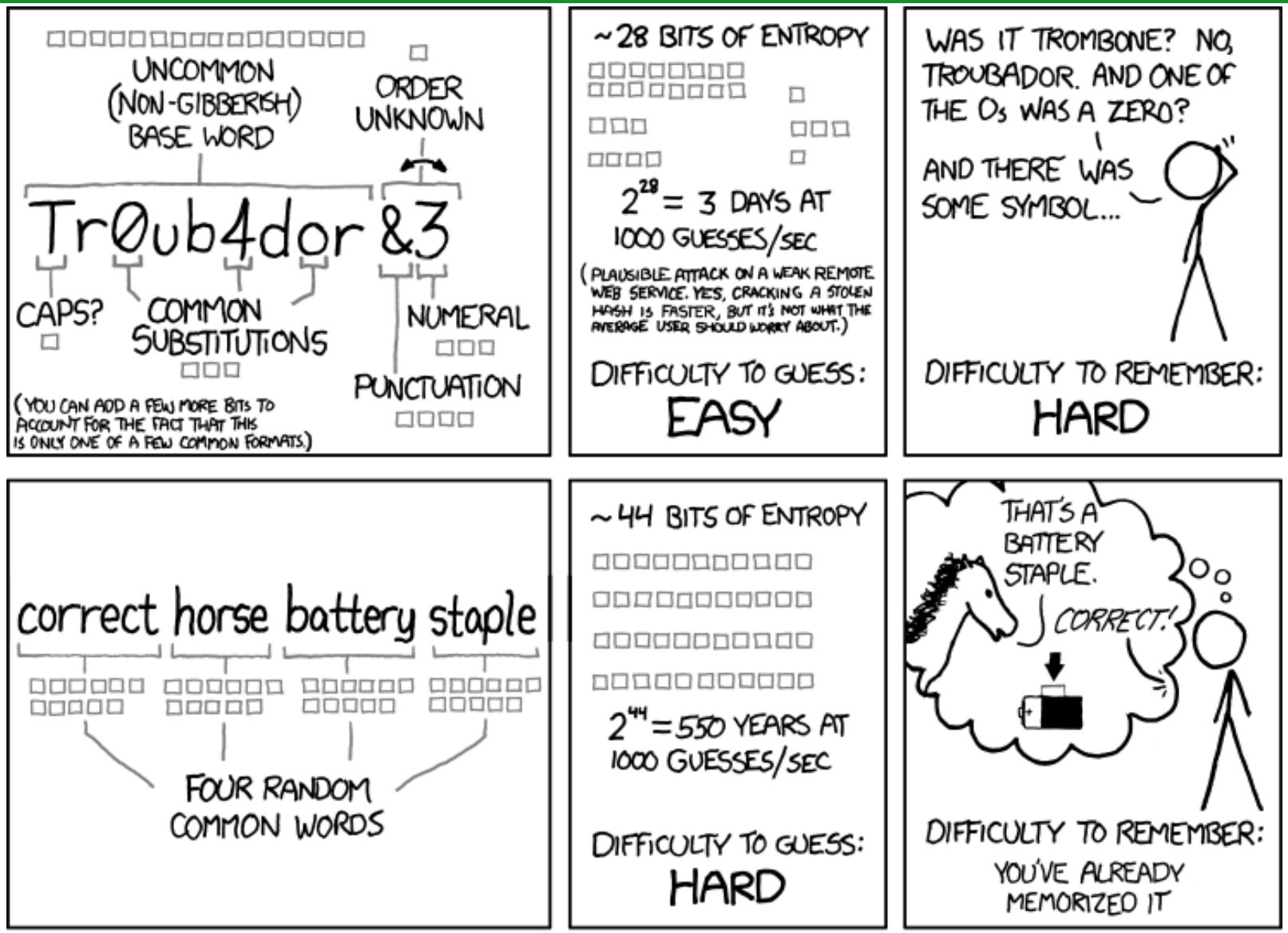
- Aumentando la cardinalità delle password possibili
- Controlli di coerenza
- Crittografia a chiave pubblica e protocolli articolati

L'autorizzazione conseguita con l'autenticazione dura un intervallo temporale detto sessione.

Lezione XV: Autenticazione

Password guessing

- Una password può essere scelta in maniera prevedibile (anziché del tutto casuale) nell'insieme possibile.
- *Online guessing*: l'attaccante prova tutte le password possibili (brute force); si limitano i tentativi e/o si rallenta il feedback



	45	3
	182	4
	1002	5
	3106	6
	5694	7
	10748	8
	15374	9
	19126	10
	20532	11
	15996	12
	11225	13
Da `/usr/share/dict/italian`	6931	14
	3535	15
	2020	16
	733	17
	339	18
	160	19
	72	20
	40	21
	10	22
	2	23
	5	24
	1	25

$$26^8 = 2,088 \cdot 10^{11}$$

$$4000^4 = 2,560 \cdot 10^{14}$$

Offline guessing

Offline guessing: l'attaccante accede all'elenco dei segreti (generalmente crittati con hash) e prova elenchi di parole (dictionary attack); si salano gli hash per rendere impraticabile la realizzazione di *rainbow table*.

Utente	salt	stored password
Alice	42	$\mathrm{hash}(42 \vert \mathrm{password}_{\mathrm{Alice}})$

- Possibilità di intercettazione
- Utilizzo in occasioni differenti
- distribuzione iniziale delle credenziali (si fanno scadere al primo accesso)

Riassumendo

Per i servizi critici è fondamentale curare il controllo degli accessi

- Autenticazione e autorizzazione sono logicamente distinte
- Le credenziali sono un elemento critico per la sicurezza di tutto il sistema di controllo.

Credenziali generalizzate

Alice può provare la sua identità mostrando

- qualcosa che sa (password tradizionale)
- qualcosa che ha (authentication token)
- qualcosa che è (biometria)

È naturalmente possibile (e spesso desiderabile) avere autenticazioni a piú fattori.

Client inaffidabili

La diffusione del malware ha reso spesso inaffidabili i client

Chi garantisce che la schermata di login non sia un *cavallo di Troia* capace di memorizzare/rubare le credenziali?

I tastierini

La protezione del "tastierino" che **cambia ad ogni login** è puramente apparente. Un esempio di falsa sicurezza, che tra l'altro impedisce all'utente di utilizzare meccanismi automatici di memorizzazione delle password

Anti-malware

Contro client alterati è difficile proteggersi, ma protezioni piú efficaci sono:

- In ogni sessione viene comunicata solo parte della password
- Two-factor authentication (2FA)
- One-time password (OTP)

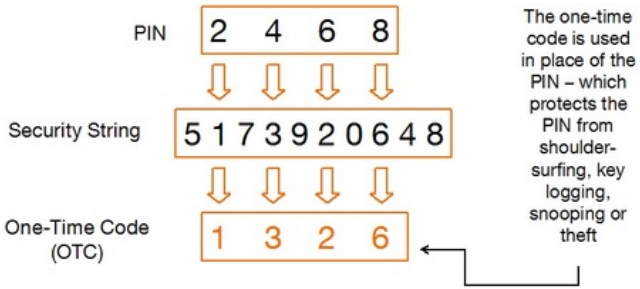
Altri attacchi

Nessuna di queste misure protegge da:

- Man in the Browser il client alterato non si limita ad intercettare le credenziali, ma è in grado di manipolare i dati della transazione
- Session hijacking l'attaccante è in grado di manipolare una sessione già in corso

2FA

Due (o piú) meccanismi di autenticazione: una password e il possesso di un oggetto fisico: p.es. un security token o un telefono cellulare.



È importante che i due fattori siano effettivamente indipendenti: web browsing con uno smart-phone e sms?

Security token

I piú diffusi si basano su synchronous dynamic password: la password cambia ad intervalli regolari, per esempio ogni minuto. La sincronia è un fattore critico

One Time Password

In generale si tratta di password utilizzate una volta soltanto e pertanto non riutilizzabili da un eventuale intercettore.
L'effettivo possesso di un security token è generalmente verificato tramite una one time password generata dal token stesso o richiesta *out of band*.

Riassumendo

Le autenticazioni possono combinare piú fattori

- Segreti
- Oggetti fisici
- Parametri biometrici

Lo schema di Lamport

Leslie Lamport nel 1981 ha proposto uno schema per autenticazione tramite OTP che non prevede la necessità di sincronizzazione temporale.
Si basa sull'esistenza di una funzione di hash H sicura (non invertibile).

Lo schema di Lamport

1. Alice e Bob concordano un segreto W
2. Bob conserva $H(\dots H(H(W))\dots) = H^n(W)$ e n
3. Autenticazione
   1. Alice comunica la propria *username*
   2. Bob risponde con il numero n
   3. Alice comunica $x = H^{n-1}(W)$
   4. Bob verifica che $H(x) = H^n(W)$ e decrementa n

Lo schema funziona n volte, poi bisogna cambiare W.

S/KEY

Lo schema di Lamport è stato implementato da Neil Haller, Phil Karn e John Walden in S/KEY.
Usa numeri a 64 bit + 2 bit di parità.
Stringhe casuali di 8 caratteri sono difficili da utilizzare per un utente umano, è prevista una mappatura su 2048 parole da 1 a 4 caratteri: i 66 bit diventano una sequenza di 6 parole
(`TAG SLOW NOV MIN WOOL KENO` $\leftrightarrow$ `0x3F3BF4B4145FD74B`).

```
{       "A",    "ABE",  "ACE",  "ACT",  "AD",   "ADA",  "ADD",
"AGO",  "AID",  "AIM",  "AIR",  "ALL",  "ALP",  "AM",   "AMY",
"AN",   "ANA",  "AND",  "ANN",  "ANT",  "ANY",  "APE",  "APS",
"APT",  "ARC",  "ARE",  "ARK",  "ARM",  "ART",  "AS",   "ASH",
"ASK",  "AT",   "ATE",  "AUG",  "AUK",  "AVE",  "AWE",  "AWK",
"AWL",  "AWN",  "AX",   "AYE",  "BAD",  "BAG",  "BAH",  "BAM",
"BAN",  "BAR",  "BAT",  "BAY",  "BE",   "BED",  "BEE",  "BEG",
"BEN",  "BET",  "BEY",  "BIB",  "BID",  "BIG",  "BIN",  "BIT",
"BOB",  "BOG",  "BON",  "BOO",  "BOP",  "BOW",  "BOY",  "BUB",
"BUD",  "BUG",  "BUM",  "BUN",  "BUS",  "BUT",  "BUY",  "BY",
"BYE",  "CAB",  "CAL",  "CAM",  "CAN",  "CAP",  "CAR",  "CAT",
"CAW",  "COD",  "COG",  "COL",  "CON",  "COO",  "COP",  "COT",
"COW",  "COY",  "CRY",  "CUB",  "CUE",  "CUP",  "CUR",  "CUT",
"DAB",  "DAD",  "DAM",  "DAN",  "DAR",  "DAY",  "DEE",  "DEL",
"DEN",  "DES",  "DEW",  "DID",  "DIE",  "DIG",  "DIN",  "DIP",
"DO",   "DOE",  "DOG",  "DON",  "DOT",  "DOW",  "DRY",  "DUB",
"DUD",  "DUE",  "DUG",  "DUN",  "EAR",  "EAT",  "ED",   "EEL",
"EGG",  "EGO",  "ELI",  "ELK",  "ELM",  "ELY",  "EM",   "END",
"EST",  "ETC",  "EVA",  "EVE",  "EWE",  "EYE",  "FAD",  "FAN",
"FAR",  "FAT",  "FAY",  "FED",  "FEE",  "FEW",  "FIB",  "FIG",
"FIN",  "FIR",  "FIT",  "FLO",  "FLY",  "FOE",  "FOG",  "FOR",
"FRY",  "FUM",  "FUN",  "FUR",  "GAB",  "GAD",  "GAG",  "GAL",
"GAM",  "GAP",  "GAS",  "GAY",  "GEE",  "GEL",  "GEM",  "GET",
"GIG",  "GIL",  "GIN",  "GO",   "GOT",  "GUM",  "GUN",  "GUS",
"GUT",  "GUY",  "GYM",  "GYP",  "HA",   "HAD",  "HAL",  "HAM",
"HAN",  "HAP",  "HAS",  "HAT",  "HAW",  "HAY",  "HE",   "HEM",
"HEN",  "HER",  "HEW",  "HEY",  "HI",   "HID",  "HIM",  "HIP",
"HIS",  "HIT",  "HO",   "HOB",  "HOC",  "HOE",  "HOG",  "HOP",
"HOT",  "HOW",  "HUB",  "HUE",  "HUG",  "HUH",  "HUM",  "HUT",
"I",    "ICY",  "IDA",  "IF",   "IKE",  "ILL",  "INK",  "INN",
"IO",   "ION",  "IQ",   "IRA",  "IRE",  "IRK",  "IS",   "IT",
"ITS",  "IVY",  "JAB",  "JAG",  "JAM",  "JAN",  "JAR",  "JAW",
"JAY",  "JET",  "JIG",  "JIM",  "JO",   "JOB",  "JOE",  "JOG",
"JOT",  "JOY",  "JUG",  "JUT",  "KAY",  "KEG",  "KEN",  "KEY",
"KID",  "KIM",  "KIN",  "KIT",  "LA",   "LAB",  "LAC",  "LAD",
"LAG",  "LAM",  "LAP",  "LAW",  "LAY",  "LEA",  "LED",  "LEE",
"LEG",  "LEN",  "LEO",  "LET",  "LEW",  "LID",  "LIE",  "LIN",
"LIP",  "LIT",  "LO",   "LOB",  "LOG",  "LOP",  "LOS",  "LOT",
"LOU",  "LOW",  "LOY",  "LUG",  "LYE",  "MA",   "MAC",  "MAD",
"MAE",  "MAN",  "MAO",  "MAP",  "MAT",  "MAW",  "MAY",  "ME",
```

Vulnerabilità

Non protegge da Man in the Browser o Session Hijacking. Piú grave è l'attacco con n piccolo

1. Bob conserva il numero n
2. Mallory impersona Bob e manda ad Alice un $n' < n$
3. Alice risponde con $H^{n'-1}(W)$
4. Mallory potrà usare $H^{n'-1}(W)$ per sostituirsi ad Alice quando Bob arriverà a conservare n'

Mitigato rendendo Alice edotta su n corrente, in modo che possa insospettirsi per eventuali $n' \ll n$

Paper Lamport

Funziona bene con modalità “carta e penna”.

- Alice conserva una lista cartacea di password $(H^n(W), H^{n-1}(W), \dots)$
- Una volta usata la prima della lista la cancella

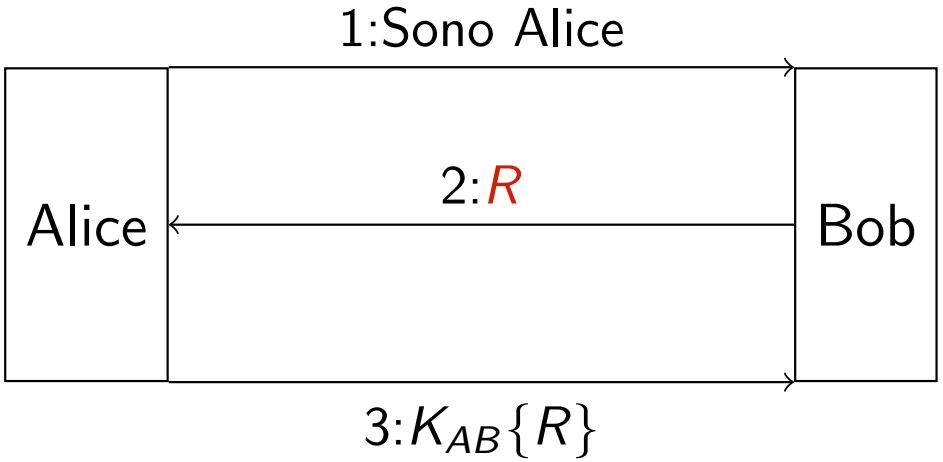
Questo schema non è suscettibile dell'attacco di n piccolo, ma la lista è naturalmente un punto critico.

Riassumendo

Lo schema di Lamport

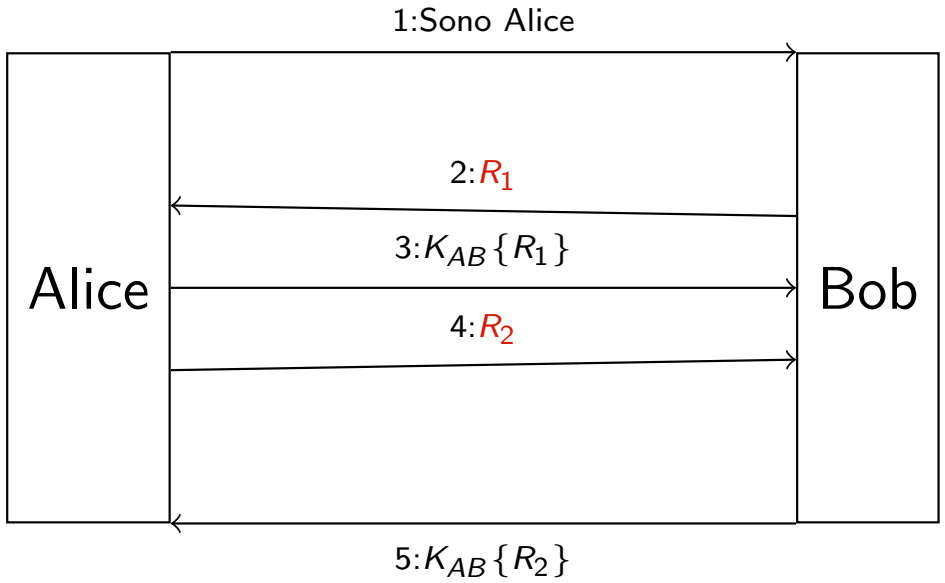
- Un meccanismo algoritmico per OTP
- Si basa sull'esistenza di funzioni di hash non invertibili
- Vulnerabile all'attacco '*n* piccolo'

Challenge/Response



- L'autenticazione non è reciproca: qualcuno potrebbe sostituirsi a Bob.
- Offline-guessing di K_{AB} è possibile intercettando R e $K_{AB}\{R\}$

Challenge/Response con mutua autenticazione

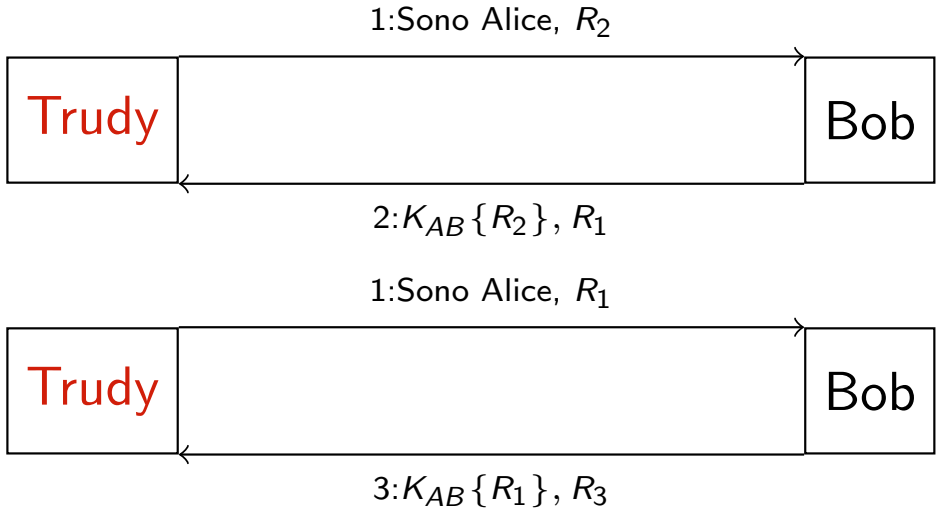


Sono necessari ben 5 scambi: si può rendere piú efficiente?

Challenge/Response con mutua autenticazione

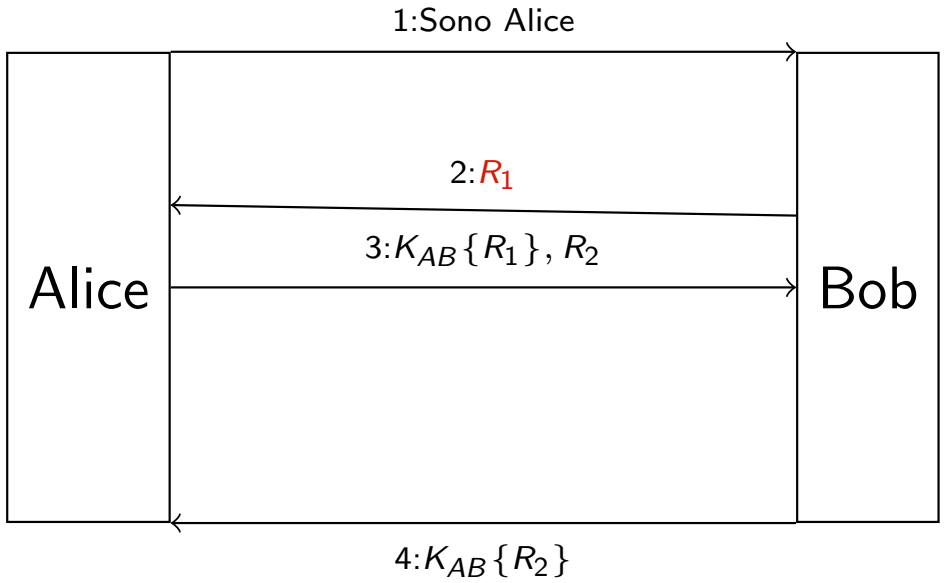
Challenge/Response con mutua autenticazione

Purtroppo si presta ad un reflection attack



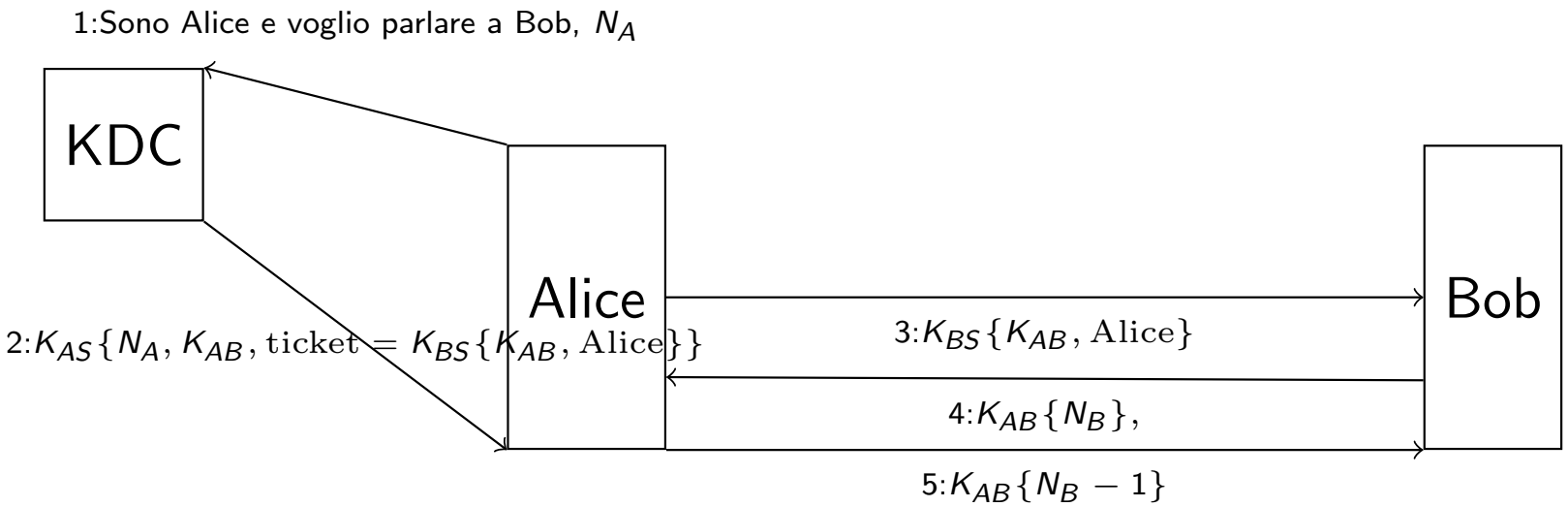
Inoltre si presta all'offline guessing (anche senza intercettazione!).

Challenge/Response con mutua autenticazione



Needham-Schroeder

Per semplificare la gestione dei segreti condivisi, possono essere introdotti dei Key Distribution Center (KDC).
Needham-Schroeder [1978]



Needham-Schroeder è vulnerabile

Qualora K_{AB} sia compromesso (p.es. accedendo alla macchina di Alice) è possibile un replay attack del *ticket*, quindi occorre complicarlo ulteriormente introducendo dei timestamp, in modo che i ticket non possano essere riutilizzati. Un'evoluzione (molto diffusa) che include i timestamp è Kerberos.

Riassumendo

I protocolli crittografici possono essere molto utili

- Permettono mutua autenticazione
- Occorre fare molta attenzione alle possibilità di replay
- La gestione delle chiavi è architetturalmente la faccenda piú complicata

Il single sign-on (SSO)

L'idea di avere credenziali che permettono l'accesso a sistemi diversi è appetibile per una serie di ragioni

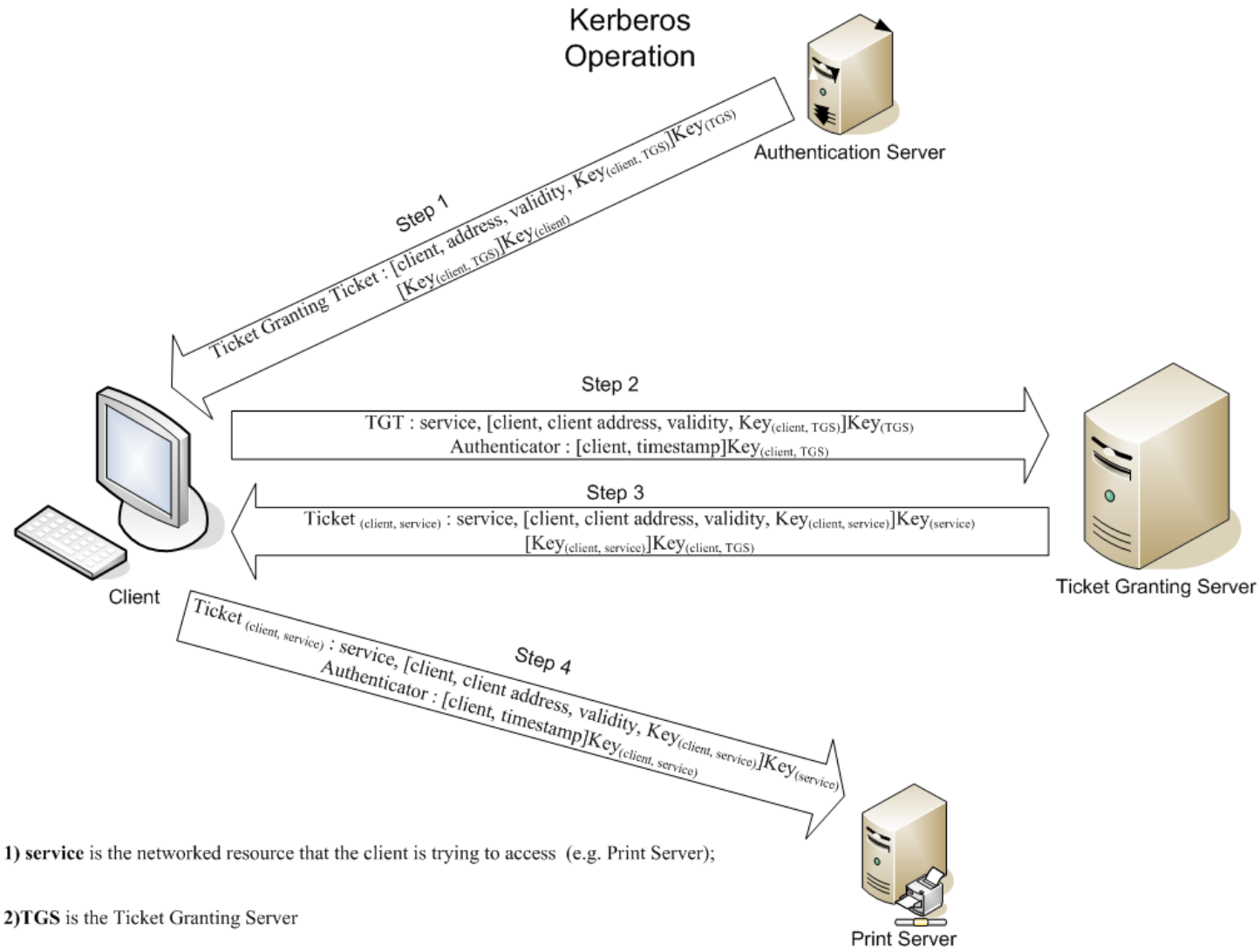
- Riduce il problema di trovare un buon segreto (sufficientemente casuale, ecc.)
- Riduce l'overhead totale di gestione degli accessi
- Permette la gestione centralizzata degli accessi, piú semplice da manutenere

(Aumenta la criticità delle credenziali, però)

Architettura di base

La maggior parte dei protocolli sono disegnati sull'impronta di Kerberos, che è una variazione con timestamp del protocollo Needham-Schroeder.
Un Ticket-Granting Server (TGS) fornisce ticket d'accesso a scadenza.

Kerberos



[Figura: wikipedia.org]

Il SSO sul web

Il SSO sembra particolarmente attraente in situazioni come i servizi web a bassa criticità:

- Decine di password da ricordare
- Utenti poco sensibili al problema

Ma soluzioni tipo Kerberos sono difficili da adottare, perché occorre che servizi indipendenti concordino sulla KDC.

OpenID

L'idea è che esistano siti che facciano da OpenID provider (OP) e gli OpenID Consumer (OC) accettano come credenziali quelle che gli utenti ottengono dagli OP (cui sostanzialmente gli OC delegano l'autenticazione)

Lo schema

1. Alice si connette al sito `meraviglie.org` (OC)
2. Per accedere comunica che ha un account su `faccialibro.com/openid/alice`
3. Alice accede (con credenziali tradizionali) a `faccialibro.com` (OP) e produce un certificato d'accesso
4. `meraviglie.org` accetta il certificato d'accesso come credenziale d'autenticazione

Problemi di OpenID

Lettura obbligatoria:
http://www.untrusted.ca/cache/openid.html
I principali rischi

- Facilità di allestire inganni di tipo **phishing**
- Privacy

Alternative migliori

In realtà sembra meglio avere credenziali multiple e gestirle con modalità come:

- PasswordSafe (`http://passwordsafe.sf.net`): un db criptato
- SuperGenPass (`http://supergenpass.com/`): un'unica password viene giustapposta ad un identificatore del sito e la vera password è ottenuta con una funzione hash. L'idea è ottima, la realizzazione ha diverse vulnerabilità: meglio usare alternative piú *crypto-savvy*, p. e.s. (`http://hpass.chmd.fr/`).

Riassumendo

Il SSO

- tipo kerberos è inadatto alla gestione di servizi web indipendenti
- le soluzioni del tipo OpenID presentano diversi problemi

Lezione XVI: L'assegnazione automatica di IP

Il DHCP

Il Dynamic Host Configuration Protocol è un elemento critico nelle reti in cui i numeri IP sono assegnati dinamicamente.

- Permette configurazioni dinamiche (quindi aggiornate)
- Ma non prevede forme di autenticazione (ipotesi trusted LAN)

Cosa fa DHCP

- Assegna i numeri IP
- Default gateway
- DNS server

Particolarmente adatto nelle reti la cui topologia cambia continuamente (es. ISP)

Funzionamento del protocollo

1. L'amministratore della rete mantiene un pool di configurazioni sul DHCP server
2. Quando un client si connette alla rete (spesso al boot) fa broadcast di una richiesta di configurazione
3. Il server assegna una configurazione del pool, comunicandola al client

Il protocollo

1. Il client fa broadcast `DHCPDISCOVER`
2. Il server risponde con `DHCPOFFER`
3. Se accetta, il client fa broadcast di `DHCPREQUEST` (il broadcast serve a rispondere anche ad eventuali altri server)
4. Il server manda un `DHCPPACK`

Vulnerabilità

Il protocollo lavora a livello di rete locale in cui i nodi

- condividono il mezzo trasmissivo
- sono "identificati" dal MAC

Address starvation

Non essendo prevista nessuna forma di autenticazione, un attaccante:

- manda molte richieste, con MAC differenti
- il pool si esaurisce
- client legittimi non riescono a ottenere una configurazione

Contromisure

Una parziale contromisura già presente nel protocollo è il concetto di leasing: una configurazione viene "noleggiata" solo per un certo tempo, poi ritorna disponibile nel pool.

Contromisure

Una parziale contromisura già presente nel protocollo è il concetto di leasing: una configurazione viene "noleggiata" solo per un certo tempo, poi ritorna disponibile nel pool.

Contromisure

Molti switch permettono di limitare il numero di MAC utilizzabili da una determinata borchia: se questo limite è minore della disponibilità del pool, l'attaccante deve controllare piú borchie per essere efficace.

Rogue Server

I client che entrano nella rete non conoscono l'indirizzo del DHCP server (infatti fanno broadcast)

- Un attaccante può allestire un rogue server
- Deve raggiungere il client prima del server legittimo

Rogue Server: sostituire il gw

Un rogue server può comunicare un configurazione scorretta.

Sostituirsi al gateway In questo modo intercetta tutto il traffico (senza agire in modalità promiscua)

Rogue Server: sostituire il DNS

Un rogue server può comunicare un configurazione scorretta.

Sostituirsi al DNS In questo modo può manipolare tutte le destinazioni espresse con nome simbolico

Difese

- L'amministratore di rete può monitorare i nodi che fanno da DHCP server o anche forzare che ciò avvenga solo da un borchia determinata
- Esistono estensioni di DHCP con varie forme di autenticazione

Riassumendo

Il protocollo DHCP

- Lavora a livello LAN, con mezzo trasmissivo condiviso e identificazione affidata ai MAC
- Generalmente non sono previste forme di autenticazione sicura
  - Address starvation
  - Rogue server

La protezione del DNS

Il DNS è un servizio fondamentale per il buon funzionamento delle reti.

- È un elemento molto importante nella catena di trust delle transazioni iniziate da un utente umano, che raramente usa direttamente i numeri IP
- È un servizio generalmente pubblico e ottenuto in maniera decentralizzata, quindi nessuno ne ha il completo controllo

La protezione del DNS

- Può essere utilizzato anche come strumento di "intelligence" prima di ulteriori attacchi
- Esistono moltissime implementazioni, non tutte curate dal punto di vista della sicurezza
- Per questi motivi è un bersaglio particolarmente attraente

Separazione di servizi DNS

Spesso si allestiscono due server DNS

DNS Esterno Riceve query da utenti esterni per informazioni riguardo host pubblicamente accessibili della rete aziendale, incluso l'MX server (il Mail Relay)

Separazione di servizi DNS

Spesso si allestiscono due server DNS

DNS Interno Riceve query da utenti interni per informazioni su host sia della intranet aziendale che di Internet. Per le query che il DNS Interno non è in grado di risolvere contatta altri DNS (query ricorsive).

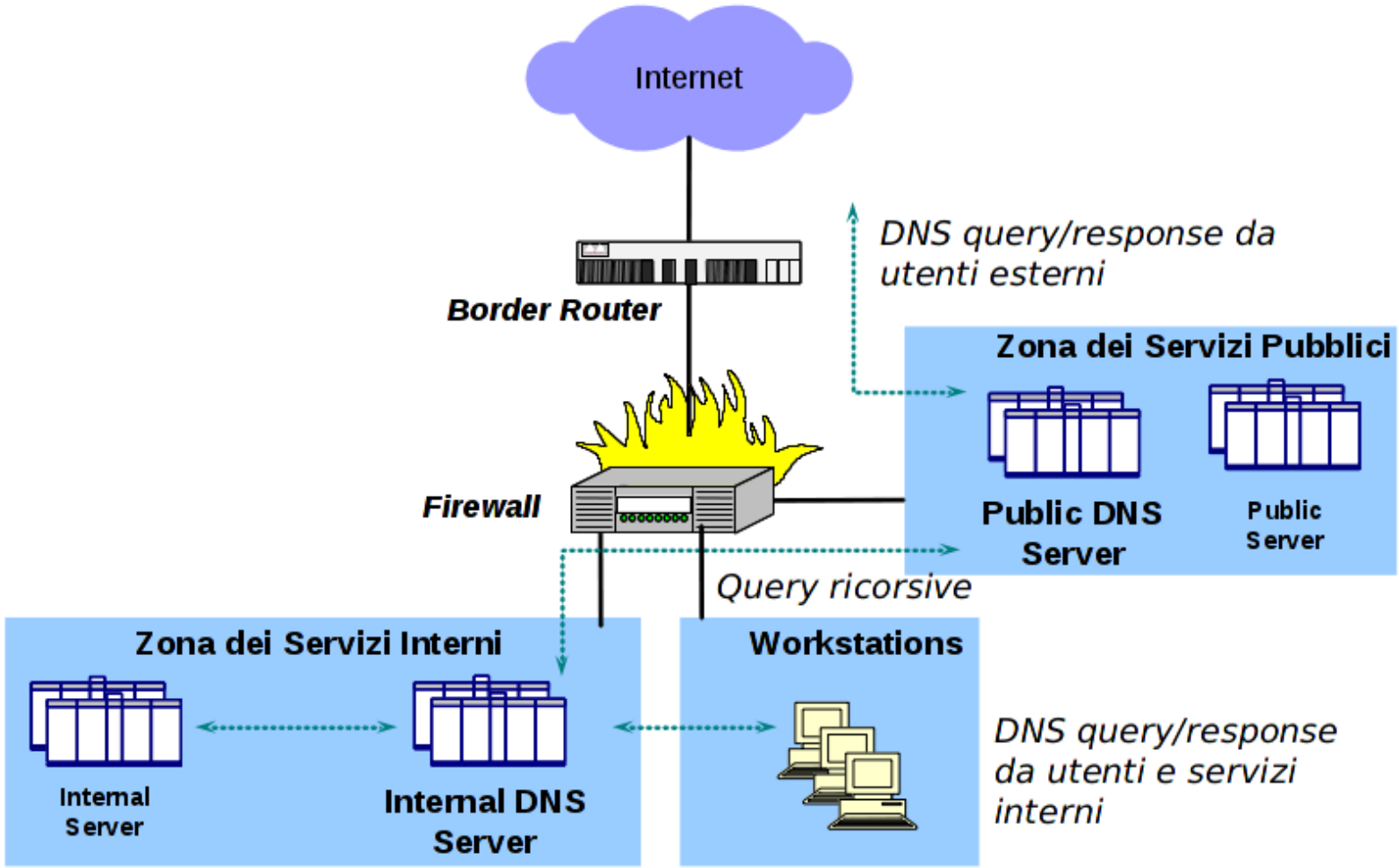
Vantaggi della separazione

I due DNS mantengono informazioni differenti (solo quelle pubbliche l'esterno, tutte quelle della intranet l'interno) e hanno connessioni con zone a diverso grado di sicurezza.

Vantaggi della separazione

- Separazione fisica delle informazioni riguardante servizi pubblici da quelle riguardanti servizi della intranet
- Assegnazione a diverse zone di sicurezza per la protezione delle informazioni
- Isolamento del DNS pubblico dalla rete interna nel caso di compromissione

Separazione DNS



Riassumendo

Il DNS è un servizio altamente critico in una rete

- Il protocollo è privo di feature di sicurezza
- È opportuno strutturarne la difesa in piú livelli

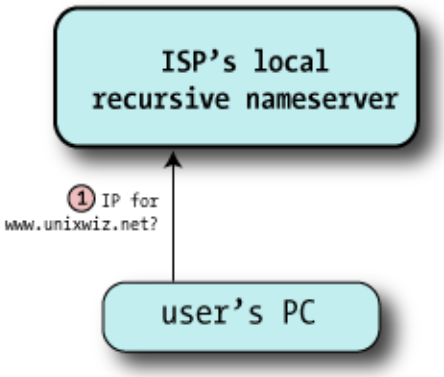
Lezione XVII: Attacchi al DNS

Risoluzione di una query

1. Si vuole risolvere `www.example.com`
2. Richiesta al server DNS configurato
3. Il DNS esamina se è risolvibile localmente o se la query è *ricorsiva:* in questo caso consulta l'elenco dei root server che conosce
4. Il root DNS non conosce l'indirizzo, ma risponde con un record di tipo `NS` corrispondente ai Global Top Level Domain (gTLD) server di `.com`
5. si ripete fin quando si ottiene il ns autorevole (authoritative) per *example.com*

Risoluzione

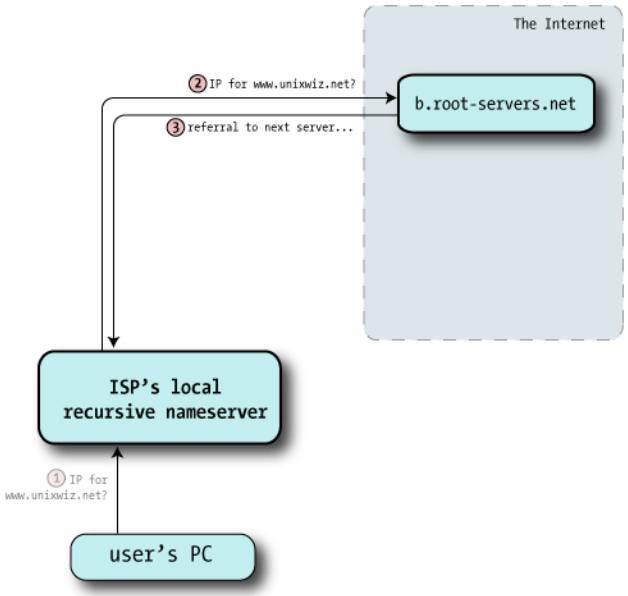
da http://www.unixwiz.net/techtips/iguide-kaminsky-dns-vuln.html



L'utente chiede la risoluzione di `www.unixwiz.net` al DNS del proprio ISP (`dnsr1.sbcglobal.net`)

Risoluzione

da http://www.unixwiz.net/techtips/iguide-kaminsky-dns-vuln.html



13 root server

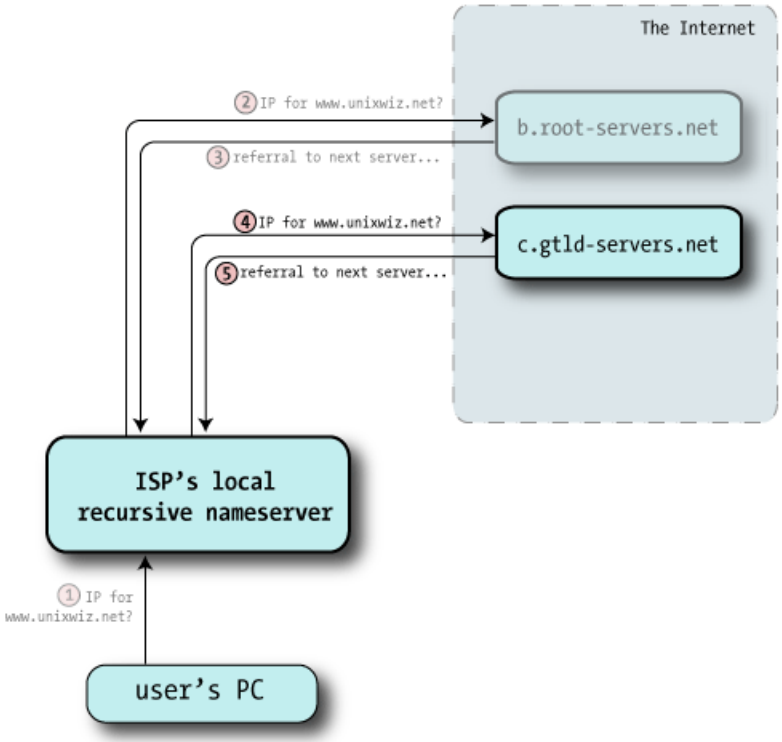
```
A.ROOT-SERVERS.NET.   IN   A   198.41.0.4
B.ROOT-SERVERS.NET.   IN   A   192.228.79.201
C.ROOT-SERVERS.NET.   IN   A   192.33.4.12
...
M.ROOT-SERVERS.NET.   IN   A   202.12.27.33
```

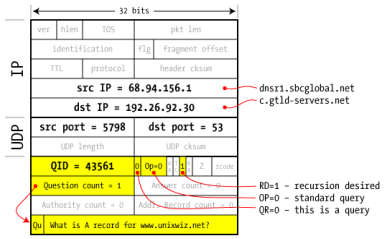
e i name server di .net

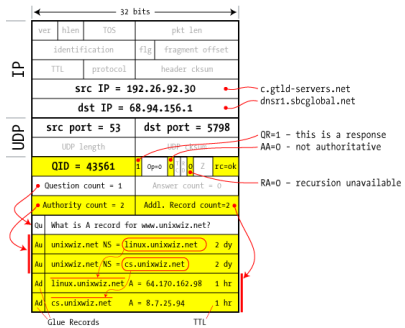
```
/* Authority section */
NET.                  IN   NS A.GTLD-SERVERS.NET.
                      IN   NS B.GTLD-SERVERS.NET.
                      IN   NS C.GTLD-SERVERS.NET.
                      ...
                      IN   NS M.GTLD-SERVERS.NET.

/* Additional section - "glue" records */
A.GTLD-SERVERS.net.   IN   A   192.5.6.30
B.GTLD-SERVERS.net.   IN   A   192.33.14.30
C.GTLD-SERVERS.net.   IN   A   192.26.92.30
...
M.GTLD-SERVERS.net.   IN   A   192.55.83.30
```

Risoluzione

da http://www.unixwiz.net/techtips/iguide-kaminsky-dns-vuln.html

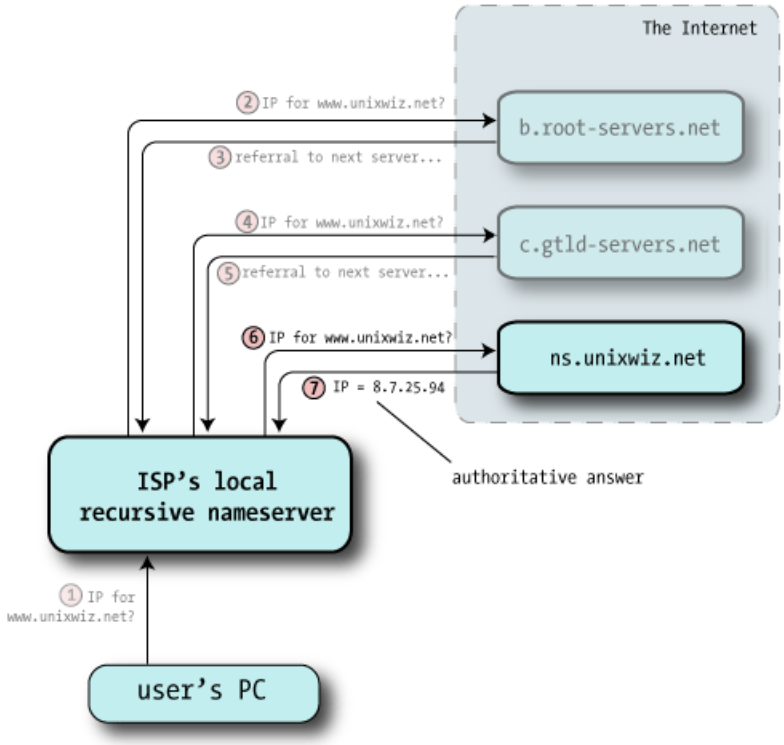


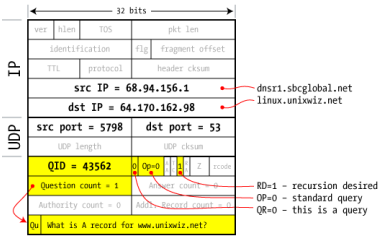


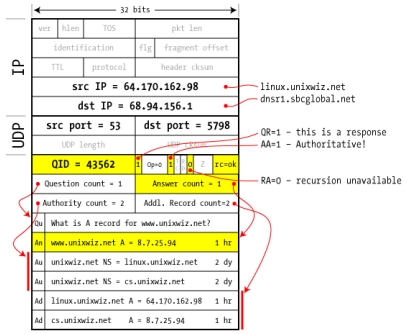


Risoluzione

da http://www.unixwiz.net/techtips/iguide-kaminsky-dns-vuln.html

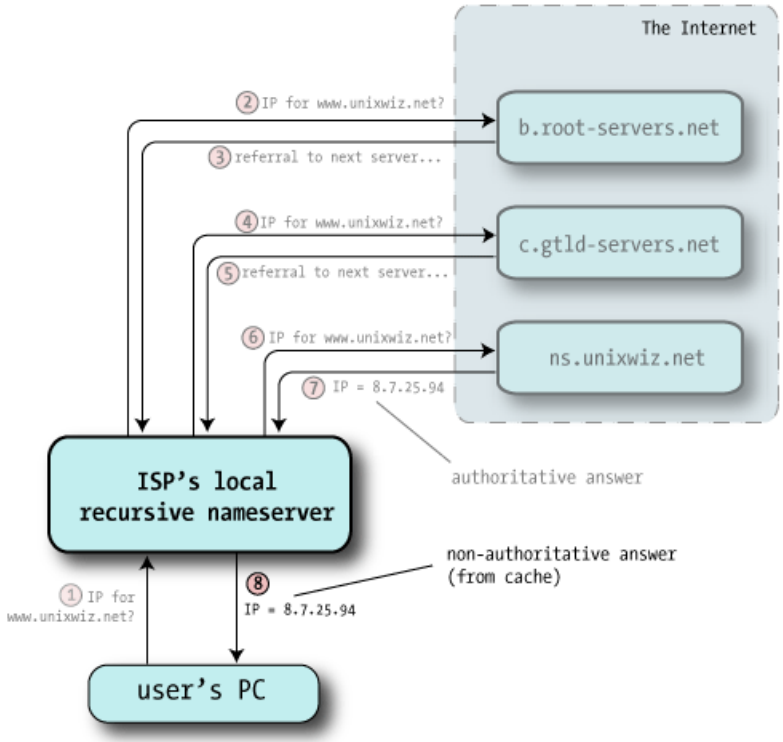






Risoluzione

da http://www.unixwiz.net/techtips/iguide-kaminsky-dns-vuln.html



Il numero IP cercato è `8.7.25.94`. Questo dato può essere conservato (per un tempo pari al TTL) in una cache locale del name server ricorsivo per rendere piú efficiente il processo.

Fragilità del sistema

Il sistema è estremamente efficiente e piuttosto resistente ai guasti, ma nella versione originaria non prevede nessuna tecnica di autenticazione e integrità delle informazioni.

Fragilità del sistema

- Il name server x di `unixwiz.net` potrebbe ospitare le associazioni per i nomi della *zona* `bancaditalia.com`, anche se nessun gTLD ne delegherebbe la risoluzione a x
- Un attacco possibile è l'avvelenamento della cache (cache poisoning)

Cache poisoning

Un attaccante riesce ad alterare la cache di un DNS ricorsivo, che pertanto restituisce un'associazione scorretta.
Come fa il DNS ricorsivo ad "autenticare" la risposta che riceve da `ns.unixwiz.net`?

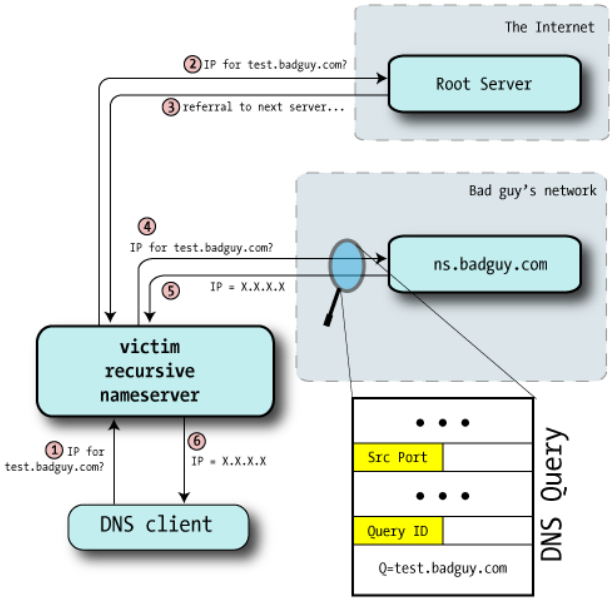
Cache poisoning

1. La risposta deve arrivare con la stessa porta UDP sorgente della richiesta. altrimenti viene scartata
2. La sezione Question coincide con quella della richiesta
3. Il query ID corrisponde a quello della richiesta
4. La risposta contiene dati riguardanti nodi nella zona (non `bancaditalia.com` per esempio)

Se l'attaccante riesce a prevedere questi dati, può alterare la cache

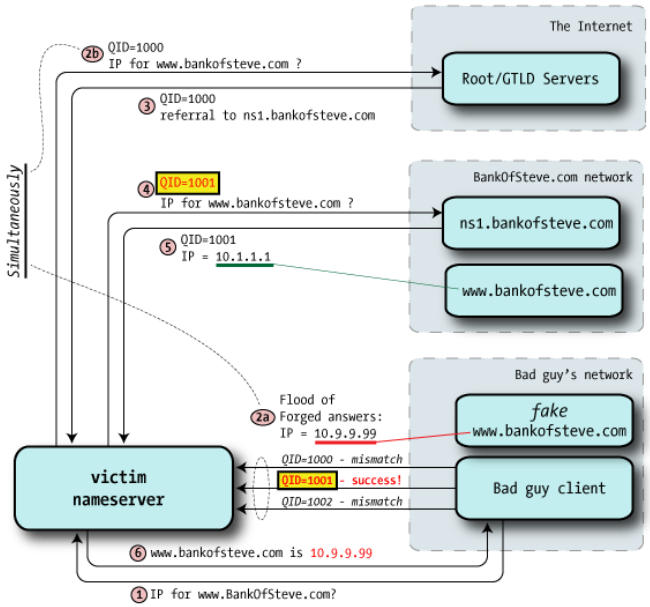
Come indovinare il Query ID?

Spesso è semplicemente un contatore, quindi basta intercettare il traffico di richieste lecite



Caso semplice

Se la porta UDP utilizzata è sempre la stessa (cosí in molte implementazioni) l'attacco è semplice



Caso semplice

Ovviamente non funziona se il nome è già nella cache. Le chance dell'attaccante sono minori se il dns authoritative è piú *vicino* al dns vittima.

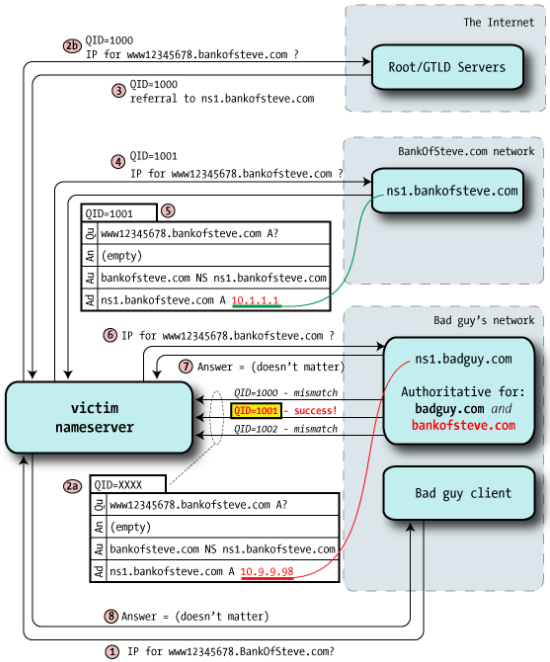
Difesa

La difesa principale è la randomizzazione del query ID

- Con ID sequenziali l'attaccante prova una ventina di ID
- Con ID random (su 16 bit) occorre provare 64K (prima che il dns authoritative risponda)

L'attacco di Dan Kaminsky

L'idea di base è la stessa, ma amplia l'impatto falsificando il dns authoritative stesso. L'attaccante ne allestisce uno proprio, che però normalmente non riceverebbe richieste.



Chance dell'attacco

Con la randomizzazione dei query ID sembra difficile fare le 64K prove necessarie in tempo utile (prima che arrivi la vera risposta).
Ma nella versione Kaminsky l'attaccante genera tanti nomi casuali (p.es. `www12345678.bankofsteve.com`).

Chance dell'attacco

Anche se riesce a fare solo poche (50?) risposte finte prima di essere superato dal vero authoritative, può comunque ripetere questo tentativo tante volte con nomi diversi: ogni tentativo ha probabilità di riuscita $\frac{50}{65536}$
Con 100 tentativi: $7,3\%$, con 1000: $53,4\%$, con 10000: $99,9\%$
Un possibile miglioramento si ha randomizzando anche la porta UDP.
Se la porta è random su 65535 ci vogliono almeno $60 \cdot 10^6$ tentativi.
(MS DNS server sceglie fra 2500 porte, quindi in realtà "bastano" $2,3 \cdot 10^6$)

Riassumendo

Il protocollo DNS

- non prevede autenticazioni nelle query
- l'associazione query/risposta si basa sul numero di porta
- se le porte sono prevedibili, si possono facilmente avvelenare le cache dei DNS

DNSSEC

DNSSEC è uno standard retro-compatibile che aggiunge autenticazione e controllo d'integrità alle query DNS. Prima versione 1997, rivisto nel 2005 e nel 2008.

DNSSEC

- È considerato un elemento fondamentale nelle strategie globali della cosiddetta *trusted* Internet
- In realtà la sua adozione langue:
  - Complessità delle configurazioni
  - Aumento del traffico
  - Perplessità di una parte della comunità sull'efficacia

Nel 2009 risultavano 274 domini firmati su circa 80'000'000 di `.com`

DNSSEC, punti fondamentali

Il concetto fondamentale è che le risposte dei DNS authoritative sono firmate digitalmente

- La chiave pubblica di una zona viene distribuita dalla zona gerarchicamente superiore (la chiave pubblica di `.net` è distribuita da un root server, ecc.)
- C'è la possibilità di avere risposte di non esistenza ("Authenticated denial of existence")

Complicazioni

Il deployment è reso complicato soprattutto dall'esigenza di ruotare le firme che scadono (di solito ogni 30 giorni), per evitare *replay attack*.

Limiti di DNSSEC

Secondo D. J. Bernstein, autore di `djbdns` e di una proposta alternativa (DNSCurve), è mal progettato:

- L'assunzione di base è che non è pensabile usare la crittografia in ogni pacchetto, per ragioni di efficienza.
- Non c'è crittografia dei dati (solo integrità)
- Le firme sono precalcolate (e quindi occorre ruotare le chiavi per limitare i replay)
- Tutti i tool per la modifica dei dati DNS devono essere 'signature aware'
- Non c'è protezione contro DoS

Ulteriori pericoli

Bernstein identifica anche vulnerabilità di DNSSEC

1. Ogni pacchetto di risposta DNS è firmato: se i dati sono alterati andrebbe scartato
   - So che i dati potrebbero essere falsi, ma non ho comunque i dati veri (denial of service)
   - Di fatto, al momento la maggior parte dei server non lo fa

Ulteriori pericoli

2. Vengono firmate solo le associazioni di cui un ns è authoritative: i glue record rimangono falsificabili
3. Il protocollo permette l'amplificazione di DDos (una query di 78 byte si può trasformare in una risposta da 3113 byte)

Riassumendo

DNSSEC è un protocollo che cerca di rendere sicura la risoluzione

- forte pressione per l'adozione
- richiede una complessa gestione delle chiavi
- non risolve tutti i problemi

L'infrastruttura di Internet

Internet è una reti di reti locali.
Il routing a livello globale, però è gestito fra Autonomous System (AS), insiemi di reti locali con un'autonomia amministrativa.

In un AS valgono routing policy specifiche, non necessariamente concordate con gli altri
Il routing fra AS è affidato a protocolli particolari.

BGP

Il Border Gateway Protocol è un protocollo usato per il routing fra AS

- path vector: l'instradamento è fatto conoscendo una serie di path
- le decisioni non sono prese con riferimento alle "distanze", ma a politiche di routing

Testo di riferimento: A. Wong, A. Yeung, *Network Infrastructure Security*, Springer

Concetti di BGP

- I nodi indirizzabili da un AS sono quelli con un determinato *prefisso*
- Un *AS path* è la lista degli AS da attraversare per raggiungere un nodo con un dato prefisso
  1. Un AS A annuncia (`UPDATE`) ai vicini quali prefissi x sa indirizzare ($A\,x$)
  2. Il vicino B annuncia ($B\,A\,x$)
  3. Chi riceve un path che contiene sè stesso non lo riannuncia
  4. I path contengono anche attributi utilizzabili nelle policy

BGP

Le comunicazioni BGP fra AS avvengono tramite una connessione TCP (porta 179). I principali pericoli sono:

- Alterazione dei dati di routing (subverted link)
- Router maligni (subverted router)

Attacchi esterni

Dai subverted link ci si può difendere con un'infrastruttura a chiavi asimmetriche (non presente nel protocollo di base). Come al solito, la gestione delle PKI è complessa, ma molto efficace. (Non difende dall'*interruzione* del collegamento, naturalmente)

Attacchi interni

Un router maligno, per:

- compromissione
- spoofing (se non c'è PKI)
- mal configurato

Vulnerabilità di BGP

Senza opportune precauzioni (estensioni PKI), BGP:

- non prevede autenticazione della sorgente, né integrità dei messaggi
- non c'è controllo sull'ownership dei prefissi
- non c'è controllo sulle informazioni di path

Controlli di coerenza

A volte è possibile rilevare un'incoerenza nelle informazioni di routing

- non sono necessariamente dovute a compromissioni
- quasi mai si riesce a determinare l'informazione corretta
- se l'attaccante conosce la topologia della rete, generalmente può produrre informazioni false, ma coerenti

Riassumendo

Il protocollo BGP per il routing fra AS

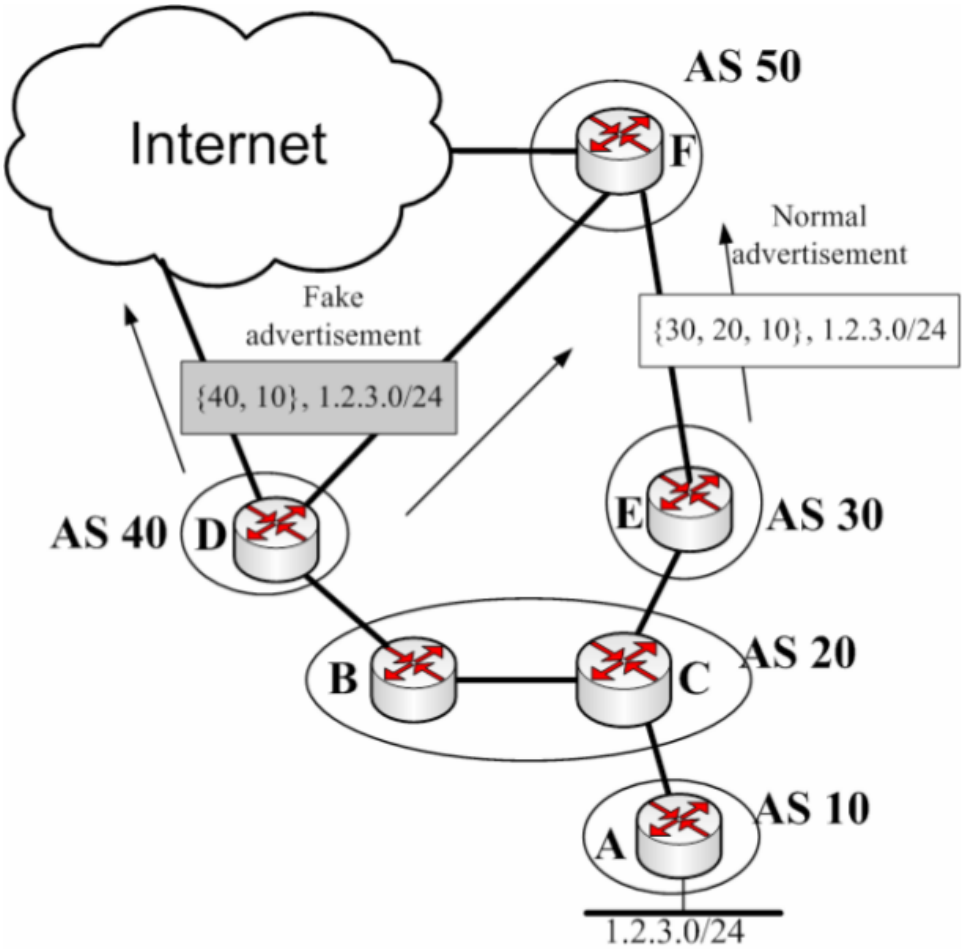
- È un protocollo path vector che permette di fare routing in base a policy complesse (non solo secondo la "distanza")
- Nella versione base non prevede garanzie di sicurezza

Obiettivi degli attacchi

La falsificazione delle informazioni di routing può servire per

- Redirezione del traffico
- Instabilità del routing
- Black hole

Prefix Hijacking



- attaccante D
- Fa finta di controllare il prefisso di A
- Se $AS\,50$ preferisce i path corti, D ha successo nella redirezione

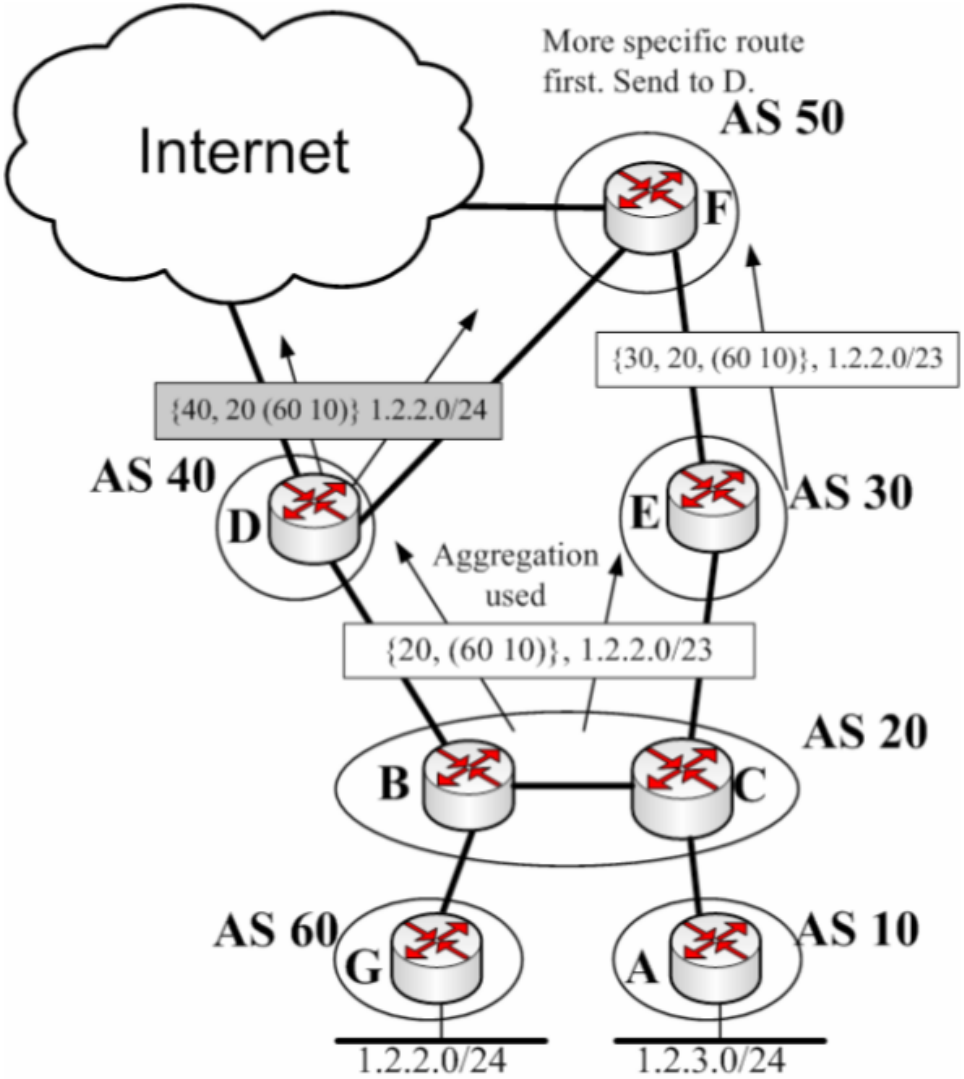
D potrebbe anche attribuirsi i prefissi di $AS\,20$

Prefix aggregation

Quando piú prefissi condividono un certo numero di bit è conveniente aggregarli

- `10.42.2.0/24` e `10.42.3.0/24` condividono i primi 23 bit
- aggregati in `10.42.2.0/23` (o `10.42.3.0/23`) permettono di accorciare i path
- allo scopo si usa un *AS set*

Prefix De-aggregation



- attaccante D
- $AS\,50$ riceve da $AS\,40$ una rotta piú specifica
- Il traffico passa per D

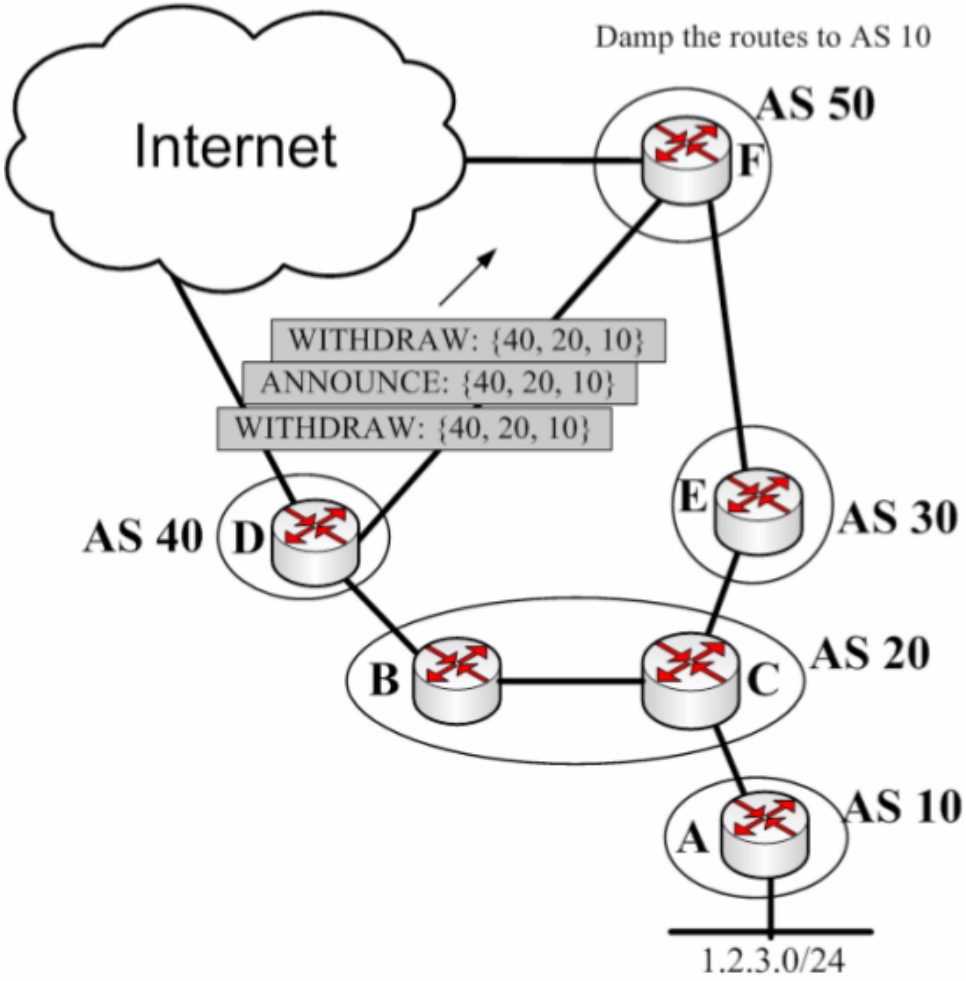
D potrebbe anche attribuirsi il prefisso `1.2.3.0/24`

Route flapping

A livello Internet è perfettamente normale avere una topologia estremamente dinamica: BGP permette di scartare e annunciare nuove rotte con facilità.

- link flapping un link viene disattivato e poi riattivato (normale)
- Se succede spesso però, crea instabilità nella rete perché gli instradamenti sono in continua variazione
- route damping la riattivazione di una rotta viene accettata con tempi crescentemente piú lunghi

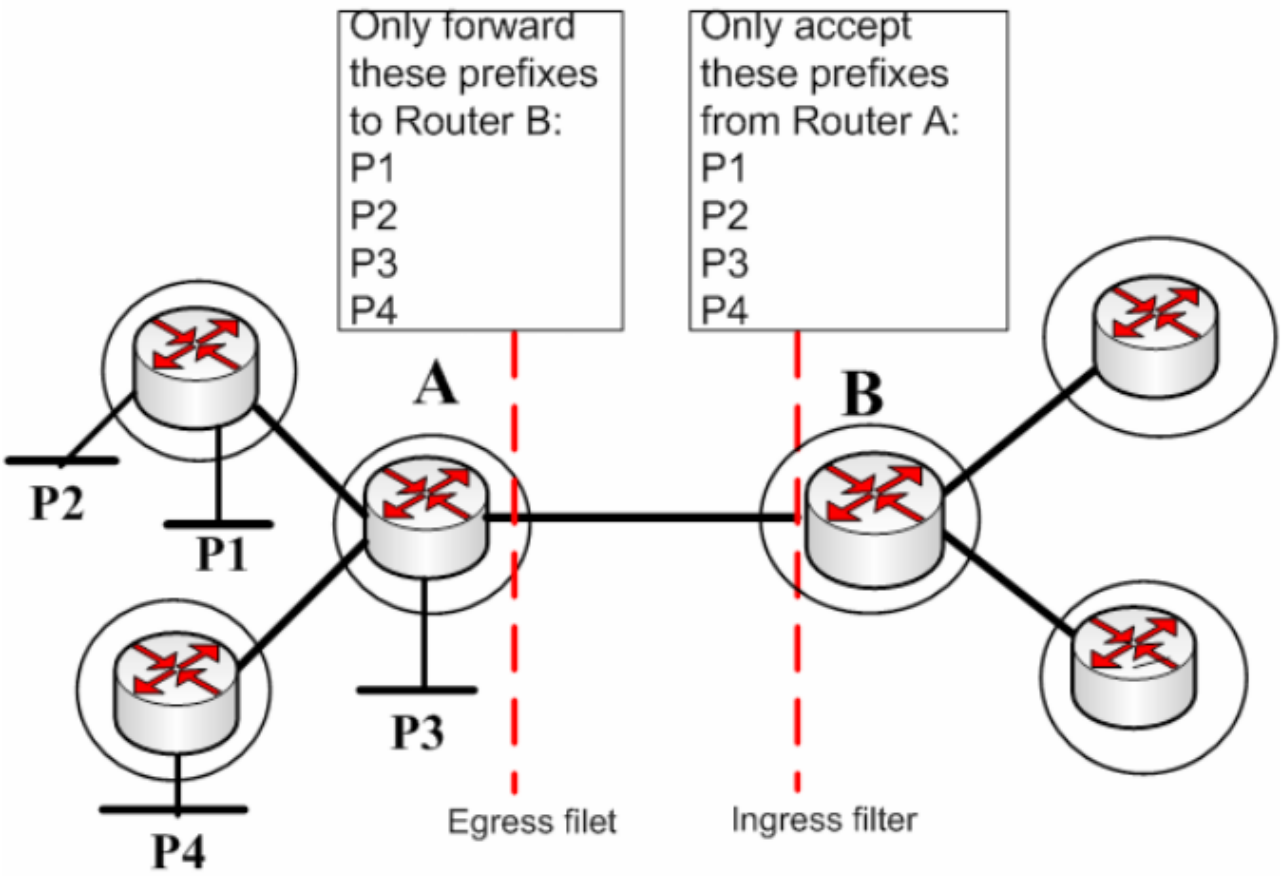
Flapping attack



- attaccante D
- $AS\,50$ si convince che il link è flapping
- $AS\,10$ diventa irraggiungibile da $AS\,50$ a causa del damping

Contromisure

La contromisura piú semplice è l'attivazione di filtri ingress e egress che scartano i path relativi a prefissi "imprevisti" Internet Routing Registry (IRR) (http://www.irr.net)



Versioni sicure di BGP

Ci sono diverse evoluzioni sicure di BGP

- S-BGP: PKI e IPsec
- Secure Origin BGP (Cisco): PKI, nuovi messaggi BGP
- IRV: indipendente dal protocollo (non solo BGP), basta un livello di trasporto sicuro

Riassumendo

Il protocollo BGP senza precauzioni è vulnerabile

- Prefix hijacking
- Prefix de-aggregation
- Flapping attack

Lezione XVIII: Reti wireless

Reti wireless

La prima rete wireless nasce nel 1971 fra le isole dell'arcipelago delle Hawaii (ALOHANET).
Dagli anni '90 hanno avuto una enorme diffusione:

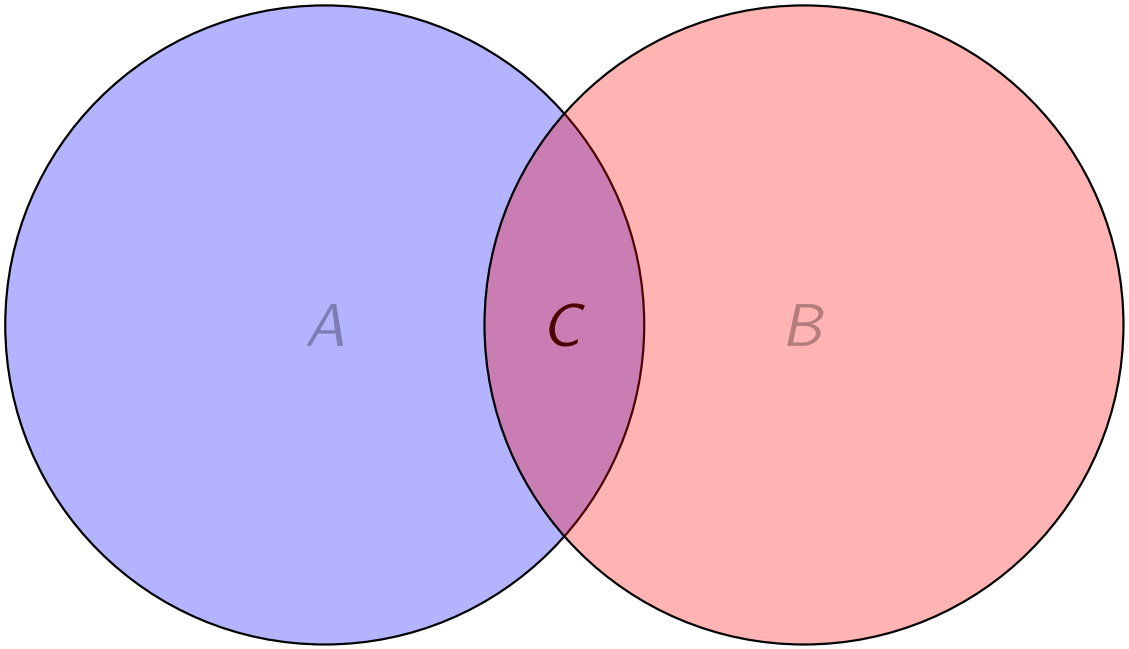
- Trasmissione via onde radio
- Bassi costi infrastrutturali
- Flessibilità nell'utilizzo

Caratteristiche salienti

Dal punto di vista della sicurezza:

- Segnali intrinsecamente broadcast
- Anche se, a differenza delle LAN cablate, il canale non è necessariamente "condiviso" fra tutti, perché ci possono essere range di ricezione diversi
- Attenuazioni e interferenze

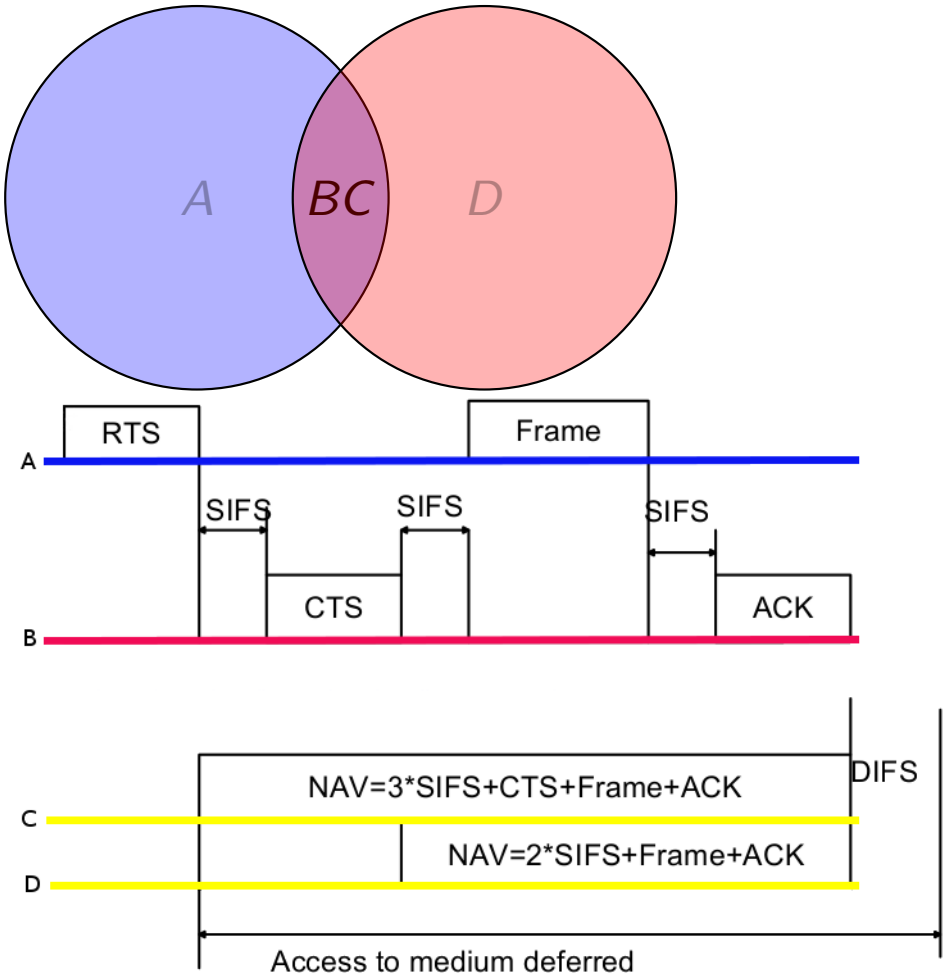
Collision detection



A non sente B, ma esiste comunque una zona di collisione dei due segnali: non si può usare il meccanismo di Ethernet.

CSMA/CA

Carrier Sense fisico + "virtuale"



- $A \rightarrow B$
- Tempi standard
  SIFS<PIFS<DIFS<EIFS
- A, ricevuto il CTS, fa partire un timer per l'ACK
- C sente RTS (e CTS), D solo CTS

802.11 (WiFi)

Gli standard IEEE per le LAN wireless sono raccolti nella famiglia 802.11

- 802.11a 5GHz, 54Mb/s teorico, circa 20Mb/s, 12 canali
- 802.11b 2.4GHz, 11Mb/s (5.9 Mb/s TCP 7.1 Mbit/s UDP)
- 802.11g 2.4GHz, 54Mb/s teorico, circa 25Mb/s
- 802.11i (WPA2) standard di sicurezza
- 802.11n standard recente che permette bande elevate e l'uso congiunto di piú antenne

Modalità di funzionamento

Access point (AP) o modalità infrastruttura: ogni nodo spedisce i pacchetti tramite un AP (che poi li gira verso una rete cablata o wireless)

Ad hoc un nodo scambia i frame direttamente con un altro

Monitor una scheda di rete riceve tutti i frame

Caratteristiche

- Ogni nodo è contrassegnato dal suo numero MAC
- Ogni AP è contrassegnato da un Service Set IDentifier (SSID)
  - L'AP annuncia il SSID con regolarità (beaconing)
  - I nodi fanno scanning di un AP, passivo o attivo (*probe request*)
  - Il MAC dell'AP è detto BSSID

Servizi 802.11

Associazione usato dai nodi per connettersi ad un AP, quando ci si trova nell'ambito della sua portata

Autenticazione usato dai nodi per avere il permesso di ricevere e spedire dati

Riassumendo

Le reti wireless 802.11

- Trasmissioni broadcast, con tecniche di *carrier sense* particolari
- La modalità infrastrutturale prevede AP cui ci si deve associare

Vulnerabilità intrinseche

- Il segnale è facile da intercettare
- Il segnale è facile da disturbare
- In alcuni casi il nodo può avere ridotte capacità di calcolo

Attacchi frequenti

- Eavesdropping
- Denial of service
- Spoofing e message replay

L'identificazione di un nodo è spesso affidata al solo numero MAC: facilmente falsificabile (e per di piú il traffico lecito è accessibile all'attaccante!)

WEP

Wired Equivalent Privacy (WEP) 1999, tutt'ora abbastanza diffuso.

- Shared key di 40, 104 o 232 bit (WEP key)
- Le chiavi sono identificate da un keyID di un byte, perché un nodo ne può avere piú di una
- Le chiavi devono essere trasmesse tramite un canale sicuro (offline a tempo di setup)
- Non c'è protezione fra coloro che conoscono la chiave (come in una LAN Ethernet)

Autenticazione

Chiave condivisa K

1. un nodo N manda una richiesta all'access point AP
2. AP: challenge $w = a_1 \dots a_{16}$ di $16 \times 8 = 128$bit
3. N: genera initialization vector IV (24 bit), calcola $m = RC4(IV|K) \oplus w$ e manda $r = IV|m$
4. AP controlla $RC4(IV|K) \oplus m = w$

Debolezza dell'autenticazione

- Sul canale viaggiano w e $r = IV|(RC4(IV|K) \oplus w)$.
- Poiché $A \oplus B \oplus B = A$, un attaccante può calcolare $RC4(IV|K)$.
- Siccome l'IV lo sceglie il nodo (e può essere riutilizzato!), l'attaccante può autenticarsi con qualsiasi challenge w' mandando $r = IV|(RC4(IV|K) \oplus w')$

Debolezza del controllo d'integrità

In WEP il controllo d'integrità dei pacchetti è ottenuto con un semplice CRC del pacchetto (senza elementi segreti).

- $CRC(A \oplus B) = CRC(A) \oplus CRC(B)$
- l'attaccante può alterare un pacchetto e iniettarne di nuovi

Tampering

Un pacchetto p sul canale è $x = (p|CRC(p)) \oplus RC4(IV|K)$

- L'attaccante intercetta x e manda $x' = (p'|CRC(p')) \oplus x$
-

$$\begin{aligned} x' &= (p'|CRC(p')) \oplus (p|CRC(p)) \oplus RC4(IV|K) \\ &= ((p' \oplus p)|(CRC(p') \oplus CRC(p))) \oplus RC4(IV|K) \\ &= ((p' \oplus p)|(CRC(p' \oplus p))) \oplus RC4(IV|K) \end{aligned}$$

- Scegliendo opportunamente p' l'attaccante manda ciò che vuole

Injection

L'injection è ancora piú facile.

- Come visto nell'autenticazione, l'attaccante può conoscere un $RC4(IV|K)$ legittimo
- A questo punto può iniettare $IV|m|CRC(m) \oplus RC4(IV|K)$ con m qualsiasi (gli IV possono essere riutilizzati)

Fragmentation attack

I primi 8 byte dei frame 802.11b sono fissi

IP	AA AA 03 00 00 00 08 00
ARP	AA AA 03 00 00 00 08 06

L'attaccante può quindi ottenere i primi 8 byte di $RC4(IV|K)$. È possibile frammentare un messaggio in frammenti di 4 byte + 4 byte di integrity check e utilizzare la chiave cosí trovata.

Riassumendo

Le reti wireless rendono il canale facilmente accessibile agli attaccanti: occorre usare contromisure crittografiche.

- Wired Equivalent Privacy
- Un protocollo mal progettato con innumerevoli vulnerabilità

WPA

WI-FI Protected Access (WPA) nato nel 2002 per superare WEP

- Utilizzabile sullo stesso hardware
- Superando le vulnerabilità di WEP

WPA

- Sostituisce CRC con un nuovo algoritmo per l'integrity check (Michael)
- Usa ancora RC4, ma impedisce replay e correlazioni con Temporal Key Integrity Protocol (TKIP).

Autenticazione WPA

Due modalità:

- Home-and-Small-Office: Pre-shared Key (PSK) analogo a WEP
- Enterprise: usa 802.1X e un authentication server connesso all'access point con una rete *wired*

Enterprise WPA

- Ogni nodo (supplicant) condivide una chiave segreta con l'authentication server (Remote Authentication Dial-In User Service, RADIUS)
- L'access point riceve le richieste del nodo e le gira al RADIUS dal quale riceve l'ok all'autenticazione

WPA

Misure di sicurezza introdotte da WPA/TKIP

- TKIP usa una *pairwise master key* (PMK) generata differentemente per ogni nodo
- la PMK viene usata per generare 4 *pairwise transient keys* (PTK) da 128 bit.
- le PTK sono diverse in ogni sessione di associazione con un AP

PTK

Le PTK vengono generate con un 4-handshake a partire da:

- un numero casuale con seme PMK
- MAC del nodo
- MAC dell'AP
- nonce generati dal nodo e dall'AP

Integrity check

- 2 PTK vengono usate da Michael per l'integrity check
- Michael è soggetto ad un attacco per cui bastano 2^{29} (invece di 2^{64}) tentativi per falsificarlo
- perciò 2 failure escludono un nodo per un minuto

Replay di IV

Per evitare che gli IV vengano riutilizzati, TKIP introduce i TKIP sequence counter (TSC).

- 48 bit divisi in 3 blocchi da 16 (con 24 bit, dopo 5120 trasmissioni è piú probabile avere collisioni che no)
- questo permette di riutilizzare RC4, spesso cablato nello hardware

DoS WPA

I pacchetti che non superano l'integrity check vengono scartati; 2 scarti portano alla dissociazione per 1 minuto.

- L'attaccante intercetta pacchetti con IV (in chiaro)
- Modifica l'IV con valori maggiori del contatore
- L'integrity check fallisce, causando DoS

Riassumendo

WPA è un protocollo nato per superare i limiti di WEP, funzionando sui medesimi device.

- RC4 based
- Algoritmo crittografico per l'integrity check
- IV non riutilizzabili

802.11i (WPA2)

WPA nasce “per mettere una pezza a WEP”. In realtà l’IEEE stava elaborando uno standard di sicurezza che è stato completato solo nel 2004

- 802.11i
- Wi-Fi Alliance ha prodotto uno standard compatibile con 802.11i chiamato WPA2

802.11i (WPA2)

Al contrario di WPA, non permette di riutilizzare lo hardware WEP.

- Crittografia basata su AES
- Autenticazione PSK o 802.1X (come WPA)
- Counter mode-CRC MAC Protocol (CCMP) usa AES-128 in counter mode per autenticazione, confidenzialità e integrità: senza IV in chiaro

Counter mode AES

Il counter mode AES permette di trasformare un block cipher in uno stream cipher usando valori successi di un "counter": il messaggio M viene spezzato in blocchi di 128 bit
$C_i = AES_K(i) \oplus M_i$
CCMP inoltre (per questo servono 2 PTK) usa il cipher-block chaining message authentication code (CBC-MAC) in cui ogni blocco dipende dalla corretta cifratura del precedente.

Sicurezza di 802.11i

CCMP è ritenuto piuttosto sicuro, ma rimangono alcune vulnerabilità generali

- DoS
- Attacchi rollback
- Dissociazioni e de-autenticazioni

Dissociazioni e de-autenticazioni

Un attaccante può forzare una dissociazione allo scopo di:

- tentare un attacco di rollback
- intercettare i pacchetti utilizzati durante l'autenticazione (per esempio per tentare un dictionary attack)

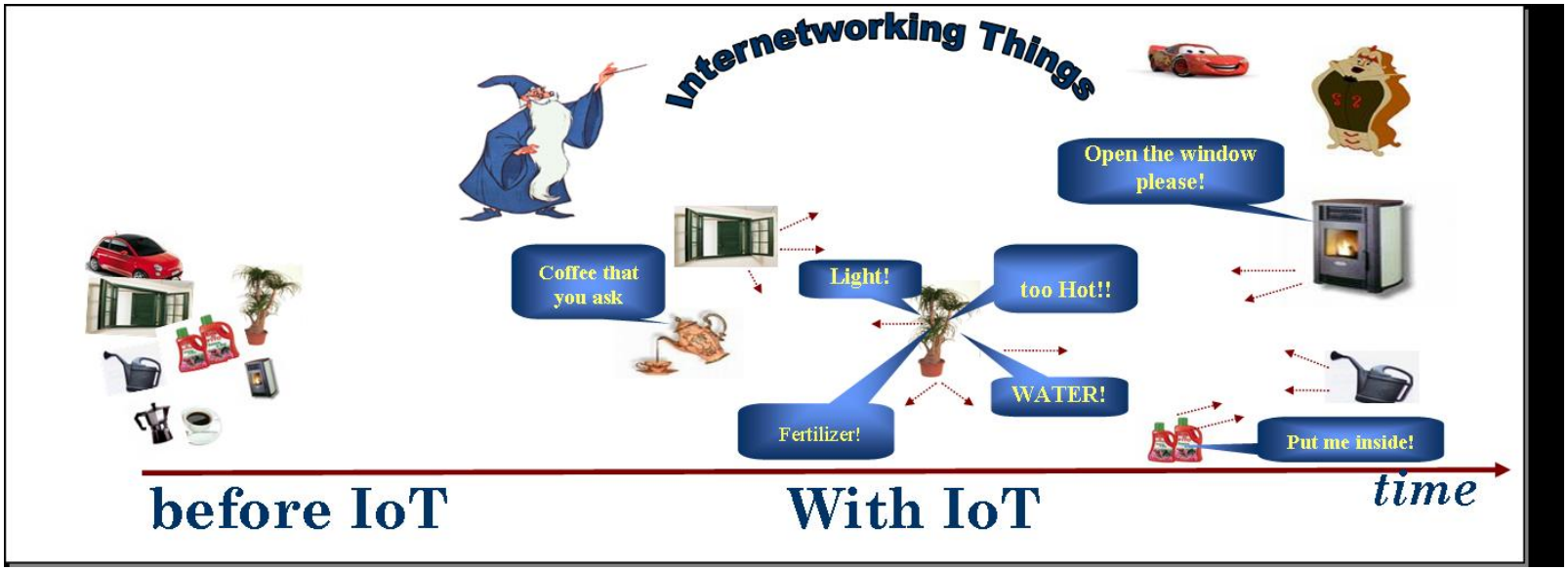
Riassumendo

WPA2 è un protocollo correntemente considerato sicuro (specie nella forma 802.1X)

- Basato su AES-128 (CCMP)
- Rimangono alcuni problemi intrinseci (DoS)
- Nel caso PSK: i dictionary attack

Lezione XIX: Wireless

Internet of Things



- Ogni oggetto dell'ambiente in cui siamo immersi potrebbe diventare un nodo intelligente di una rete di sensori.
- La realizzazione di servizi richiede lo scambio di dati e computazioni.

Problemi di sicurezza

- Le interazioni sono spesso decentralizzate
- Potenza limitata: alimentazione, capacità di calcolo, di memorizzazione, di primitive crittografiche
- Comunicazioni wireless
- malicious displacement, impersonation, and tampering

Secure data aggregation

Perché la crittografia tradizionale non è sufficiente

hop-by-hop, ma in ogni nodo è in chiaro

end-to-end, ma serve qualche segreto condiviso o crittografia asimmetrica (generalmente considerata irrealizzabile in WSN)

Castelluccia *et al.* [TOSN 2009]:

- end-to-end stream cipher: $C \oplus K = E \Rightarrow E \oplus K = C$
- usando *modular addition* $(+^m)$ invece dello *xor*: $C +^m K = E \Rightarrow (E_1 + E_2) = (C_1 + C_2) +^m K$
- In questa maniera gli aggregatori non necessitano la chiave

La locazione dei nodi

Non è sempre nota a priori.

- nodi sparsi casualmente, mobili, ecc.
- in questo caso non è un dato topologico di sistema, o semplicemente *trasmesso*
- viene *calcolato* da una *base station* con le informazioni ricevuto da nodi collaboranti

Un protocollo di localizzazione

Multilateration

- Un certo numero di landmark o ancore v_i vengono usate per la verifica
- I landmark scambiano beacon con i nodi da localizzare e trasmetto informazioni sui range

Several attacks known:

Node displacement

Wormholes (fabricated communication links)

Distance enlargement (con nodi fake)

Dissemination of false position and distance information (con nodi compromessi)

Localizzazione

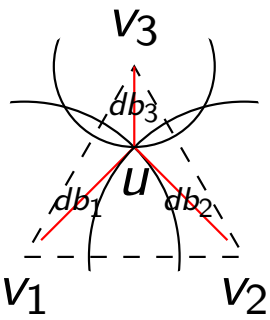
- Ogni verificatore calcola dei distance bound db_i rispetto a un nodo sconosciuto u
- Un attaccante che controlla un solo nodo può *ritardare* un beacon, ma non accelerarlo: quindi può solo apparire piú *lontano*, non piú vicino.

Secure localization

Čapkun *et al.* [IEEE JSACOMM 2006]
Servono almeno 3 verificatori di cui la base station (sink) si fida.

1. Determina u' in modo che minimizzi la somma dei delta fra i db_i e la distanza $u' - v_i$
2. Se la somma è maggiore dell'errore atteso $\rightsquigarrow$malicious
3. Altrimenti:

Se u' è contenuto in almeno un triangolo di verificatori: l'informazione è sicura, perché qualsiasi falsificazione deve accorciare un db_i



Output

Alla fine la base station può marcare le posizioni come

Robust almeno un triangolo di verificatori "certifica" il dato.

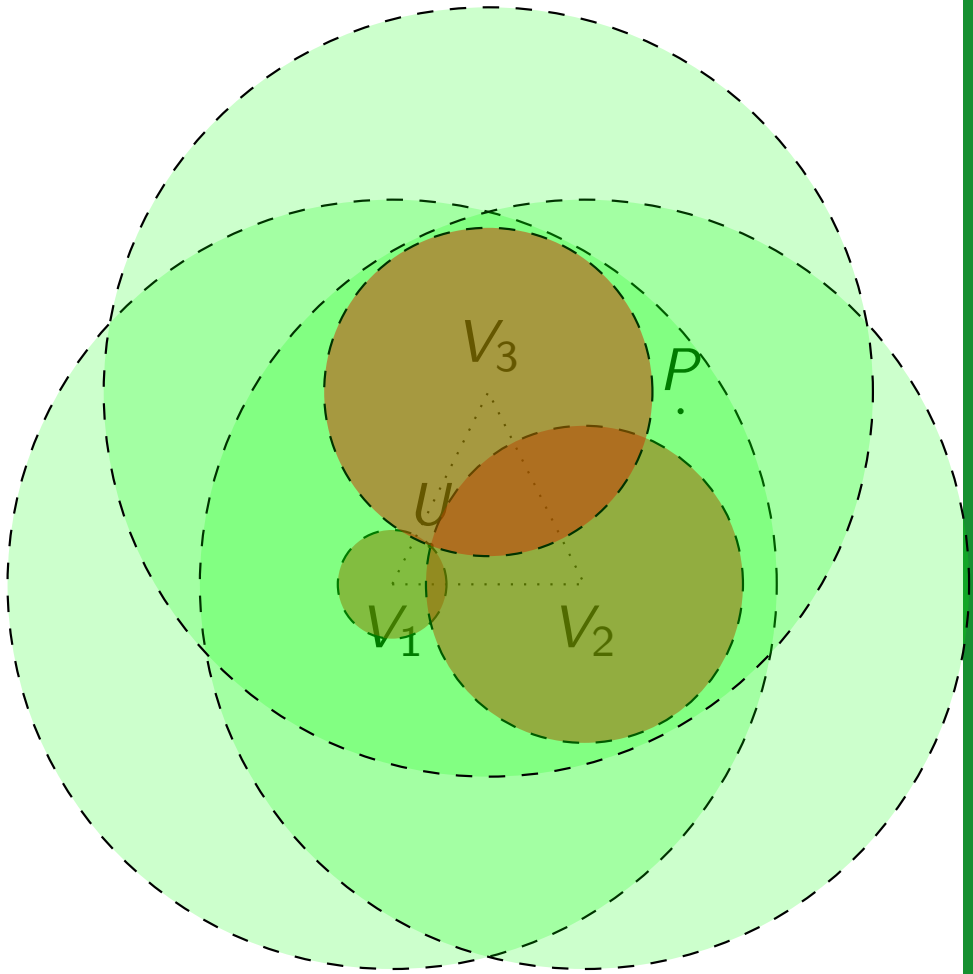
Malicious l'errore è troppo elevato perché sia casuale

Unknown

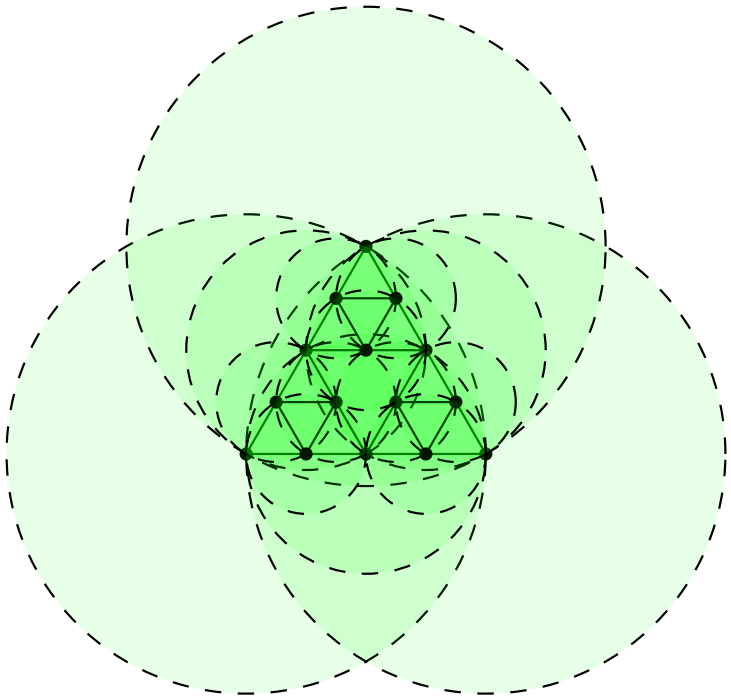
Esempio

L'attaccante può decidere dove piazzarsi (U) e quale posizione falsificare (P)

- Senza "restringere" distanze



- Power range green
- Distance bound red



# ver.	max. deception
3	$0.2516\ R$
6	$0.1258\ R$
15	$0.0629\ R$
42	$0.02145\ R$
123	$0.015725\ R$
366	$7.8625 \cdot 10^{-3} R$

Lezione XX: Censura e controllo in rete

Censura e controllo in rete

Le reti telematiche risultano essere uno strumento di libertà, ma sono esposte al rischio di controllo di massa da parte dei *carrier* e dei governi.

- Censura
- Content filtering
- Tracking delle abitudini

Anche quando ci possono essere buone ragioni, i filtri possono essere imprecisi.

Diritti fondamentali

Dichiarazione universale dei diritti dell'uomo, art. 12

No one shall be subjected to arbitrary interference with his privacy, family, home or correspondence, nor to attacks upon his honour and reputation.

Costituzione italiana, art. 15

La libertà e la segretezza della corrispondenza e di ogni altra forma di comunicazione sono inviolabili. La loro limitazione può avvenire soltanto per atto motivato dell'autorità giudiziaria con le garanzie stabilite dalla legge.

Tutela

Gli utenti delle reti hanno quindi diritto di

- veder tutelati i loro diritti dalla legge
- usare la tecnologia in modo da difendersi all'interno di una rete (quindi in potenziale conflitto con l'amministratore della rete stessa)

Privacy-Enhancing Technologies (PET)

PET: tecnologia progettata allo scopo di tutelare la privacy (privatezza, riservatezza)

- Non solo reti: le porte dei bagni sono PET...
- In campo informatico:
  - tecniche per minimizzare o eliminare i *dati personali*
  - tecniche per evitare il controllo delle attività

Privacy e sicurezza

La privacy è importante anche per la sicurezza

- identity theft
- controllo e repressione del dissenso (piú efficace della tortura, vedi *The Man in the Snow White Cell*, CIA `http://ur1.ca/6lef8`)
- le persone cambiano, ma i dati restano (diritto all'oblio)

Minimizzazione dei dati

- Ogni servizio dovrebbe richiedere e raccogliere solo l'insieme minimo di dati necessario a fornirlo
- I dati personali (o addirittura *sensibili*) dovrebbero essere raccolti solo quando strettamente necessari (e conservati in maniera adeguatamente protetta)

Sanitization

La sanitization consiste nel eliminare dai dati le caratteristiche che li rendono personali o sensibili.

- Molto difficile: anche le aggregazioni statistiche dovrebbero risultare anonime
- L'anonimato richiede spesso grandi quantità di dati (es. un nero di 30-40 abitante a Dalvík, Islanda).

Protezione

Ogni volta che dati personali/sensibili sono conservati, processati o trasmessi dovrebbero essere protetti

- Controllo degli accessi
- Crittografia e *shredding*

In Italia norme di legge piuttosto precise: vedi Decreto legislativo 30 giugno 2003, n. 196 *Codice in materia di protezione dei dati personali*.

Riassumendo

La Rete è senz'altro uno strumento di libertà d'espressione, ma si presta a un controllo sistematico e potenzialmente oppressivo.

- Censura
- Content filtering
- Privacy

Anonimato

La difesa rispetto ai pericoli di controllo è l'anonimato (non a caso tutti i tentativi di controllo politico di Internet cercano, in un modo o nell'altro, di limitare l'accesso anonimo)
Tema assai controverso, perché l'anonimato perfetto permette azioni non perseguibili (e in effetti in alcuni casi la legalità *locale* potrebbe essere in contrasto con i diritti fondamentali).

Inosservabilità

Un utente usa una risorsa senza che terze parti siano in grado di osservare l'uso
Per un evento E, Se O_A è l'insieme di eventi osservabili dall'attaccante A

Unobservability

$\forall \omega \in O_A : 0 < P(E|\omega) < 1$

Perché sia efficace $0 \ll P(E|\omega) \ll 1$

Inosservabilità perfetta

Se nessuna osservazione è in grado di cambiare la probabilità a posteriori di un evento, si parla di inosservabilità perfetta.

Perfect Unobservability

$$\forall \omega \in O_A : P(E) = P(E|\omega)$$

Incollegabilità

Un utente usa diverse risorse o servizi senza che sia possibile collegare i diversi usi.
Per due eventi E, F, con una caratteristica comune (link) $L_{E,F}$

Unlinkability

$\forall \omega \in O_A : 0 < P(L_{E,F}|\omega) < 1$

Perché sia efficace $0 \ll P(L_{E,F}|\omega) \ll 1$
Perfect Unlinkability: $\forall \omega \in O_A : P(L_{E,F}) = P(L_{E,F}|\omega)$

Incollegabilità di mittente e destinatario

Un caso particolare è l'unlinkability fra mittente e destinatario in una comunicazione.

- A, B comunicano
- A *comunicante* è osservabile, B *comunicante* è osservabile...
- ...ma non è osservabile il fatto che A comunica con B

Anonimato

Un utente usa una risorsa senza rendere nota la propria identità. Si definisce rispetto al ruolo $R_{U,E}$ dell'utente U nell'evento e un insieme W di identità

Anonymity

$\forall \omega \in O_A,\ \kappa \in W : 0 < P(R_{\kappa,E}|\omega) < 1$

In pratica deve essere $0 \ll P(R_{\kappa,E}|\omega) \ll 1$

Anonimato perfetto: $\forall \omega \in O_A,\ \kappa \in W : P(R_{\kappa,E}) = P(R_{\kappa,E}|\omega)$

Pseudonymity

Un utente usa una risorsa identificandosi con uno pseudonimo.

- Lo pseudonimo rimane costante
- ma non è possibile (o solo alcuni sono in grado di farlo) collegarlo all'identità reale
- può essere legato ad un ruolo

Riassumendo

La principale difesa rispetto ai pericoli di controllo è l'anonimato:

- Inosservabilità, unlinkability
- Anonimato e pseudonimi

Lezione XXI: Virtual Private Network

Virtual Private Network

VPN

Una rete "overlay" che costituisce un dominio amministrativo sostanzialmente indipendente dalla topologia effettiva della rete sottostante

- Le comunicazioni sono criptate
- Molto usate per permettere a utenti roaming di accedere alle risorse delle reti aziendali

Tunnel

Si basano sul concetto di tunnel: ossia incapsulano i pacchetti in un altro protocollo (che rispetta la topologia della rete "fisica" sottostante)
Differiscono per il livello di rete virtuale che offrono.

Encryption

Anche la confidenzialità e integrità dei pacchetti può essere ottenuta a diversi livelli

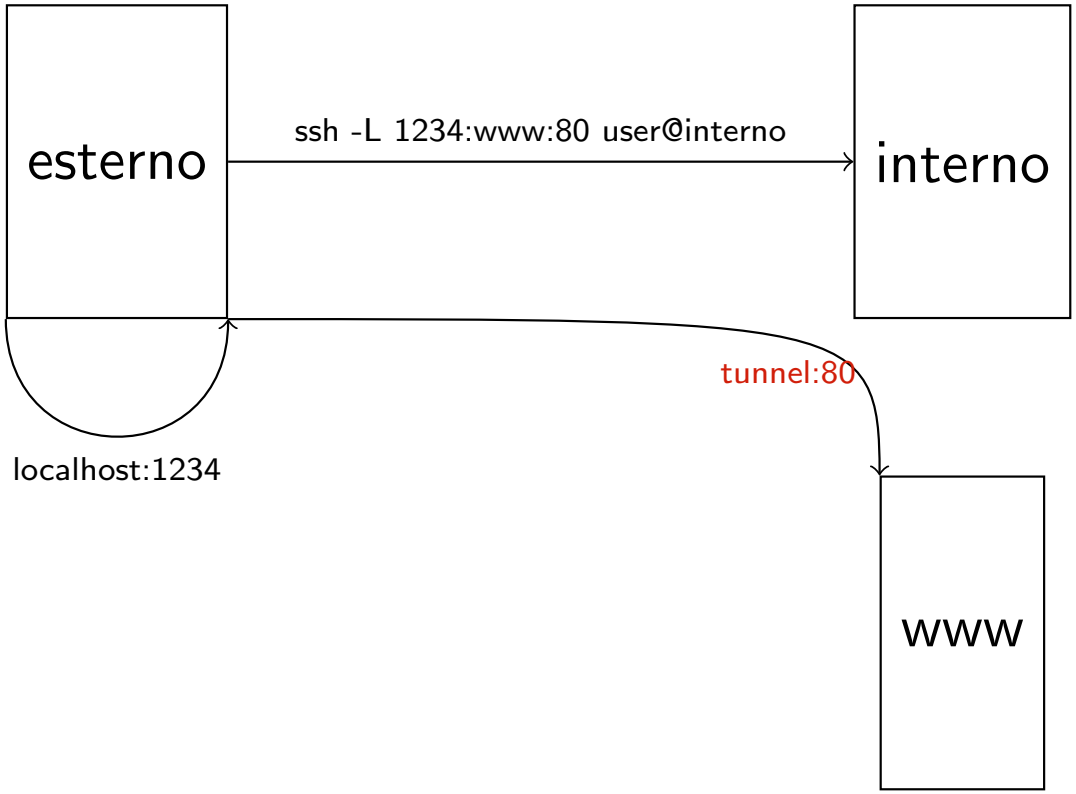
- IPSec
- SSL/TLS
- Protocolli proprietari
- SSH

SSH

OpenSSH può essere usato per fare tunneling di altri protocolli in SSH

- Port forwarding
- Vera e propria VPN

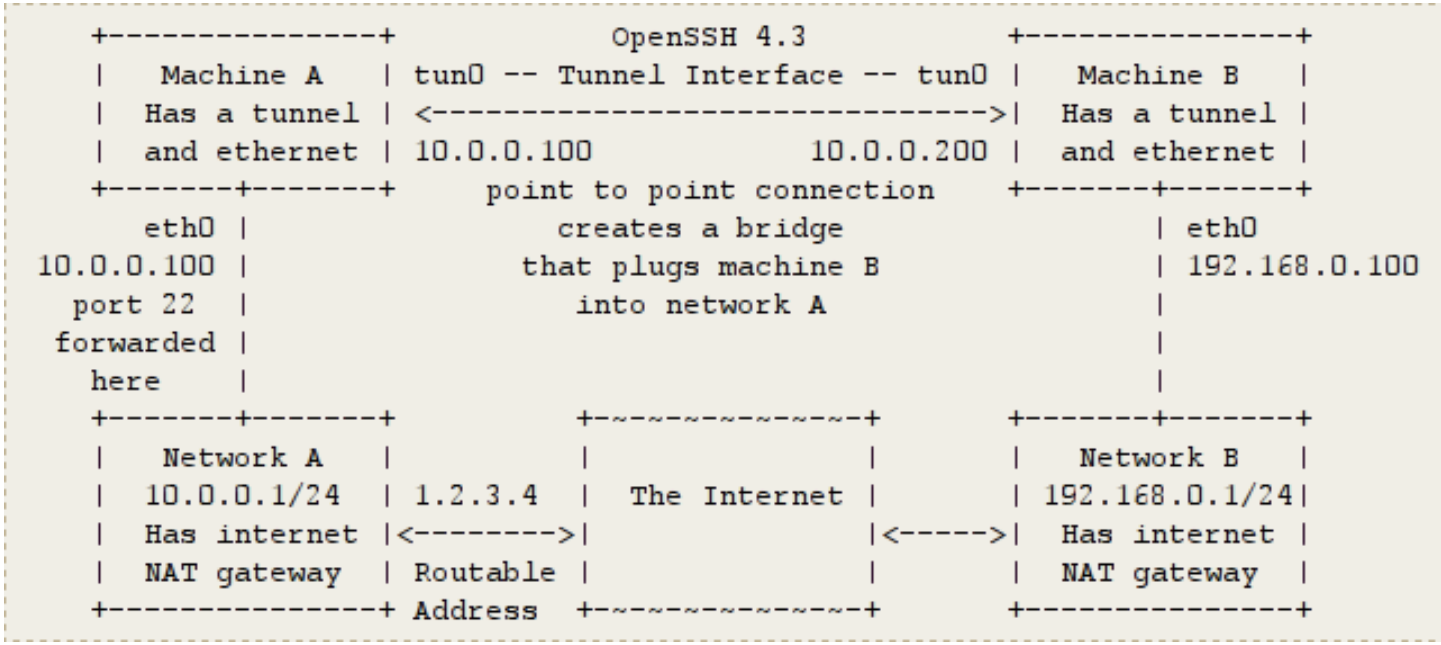
Port forwarding



www vede una connessione da interno, il pezzo criptato è solo esterno/interno

Una vera VPN con SSH

Per avere una vera VPN bisogna appoggiarsi sui livelli piú bassi dello stack. Con Linux esiste un device `tun` che serve proprio a questo scopo (Tunneling/NAT regolato da applicazioni utenti)



Setup di TUN/TAP

Se non è già attiva, bisogna creare l'interfaccia TUN/TAP
In Linux potrebbe essere necessario

```
1 mknod /dev/net/tun c 10 200
2 modprobe tun
```

Configurazione

```
# A
iface tun0 inet static
 address 10.0.0.100
 pointopoint 10.0.0.200
 up arp -sD 10.0.0.200 eth0 pub
# B
iface tun0 inet static
 pre-up ssh -f -w 0:0 1.2.3.4 'ifdown tun0; ifup tun0'
 address 10.0.0.200
 pointopoint 10.0.0.100
 up ip route add 10.0.0.0/24 via 10.0.0.200
 up ip route add 1.2.3.4/32 via 192.168.0.1
 up ip route replace default via 10.0.0.1
 down ip route replace default via 192.168.0.1
 down ip route del 10.0.0.0/24 via 10.0.0.200
 down ip route del 1.2.3.4/32 via 192.168.0.1
```

Riassumendo

Le VPN

- Permettono di usare una rete potenzialmente ostile, con alcune garanzie di confidenzialità e integrità
- Si basano sul tunnel di protocolli e possono essere realizzate in molti modi diversi

SSHuttle

Iptables + SSH tunnelling

```
1  Chain PREROUTING (policy ACCEPT 33 packets, 1604 bytes)
2   pkts bytes target prot opt in out source destination
3      33 1604 sshuttle-12300 all -- * * 0.0.0.0/0 0.0.0.0/0
4
5  Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
6   pkts bytes target prot opt in out source destination
7
8  Chain OUTPUT (policy ACCEPT 24 packets, 1568 bytes)
9   pkts bytes target prot opt in out source destination
10     48 3008 sshuttle-12300 all -- * * 0.0.0.0/0 0.0.0.0/0
11
12 Chain POSTROUTING (policy ACCEPT 48 packets, 3008 bytes)
13  pkts bytes target prot opt in out source destination
14
15 Chain sshuttle-12300 (2 references)
16  pkts bytes target prot opt in out source destination
17     0 0 RETURN tcp -- * * 0.0.0.0/0 127.0.0.0/8
18    24 1440 REDIRECT tcp -- * * 0.0.0.0/0 0.0.0.0/0 TTL match TTL != 42 redir ports 12300
```

OpenVPN

Un programma open-source che permette di costruire VPN basate su tunnel TCP o UDP. (La porta assegnata da IANA a OpenVPN è 1194)
Anche in questo caso l'implementazione sfrutta il driver TUN/TAP.

Autenticazione

L'autenticazione può avvenire in due modi

Static key pre-shared key

TLS tramite una sessione SSL/TLS (su UDP) con certificati e scambio di chiavi

Tunnel

Il tunnel può essere TCP (sconsigliato per ragioni di efficienza) o UDP (default)
Si può anche scegliere se usare l'interfaccia TUN (livello network) TAP (livello link).

Configurazione

Punto a punto, TAP (indirizzi assegnati con DHCP)

```
1 client
2 remote 172.16.0.3
3 dev tap
4 tls-client
5 ca client.crt
6 auth-user-pass
```

```
1 dev tap
2 remote 172.16.0.1
3 tls-server
4 ca server.crt
5 auth-user-pass-verify script
```

con la modalità server è possibile definire VPN con topologie complicate

Firewall

OpenVPN è facile da gestire con i firewall, perché usa un'unica porta (generalmente UDP 1194)

```
iptables -A INPUT -p udp --dport 1194 -j ACCEPT
```

E poi

```
iptables -A INPUT -i tun+ -j ACCEPT
```

(o l'equivalente con tap)

HMAC

Ogni pacchetto ha un message authentication code calcolato con HMAC, per cui i pacchetti non integri vengono scartati

$HMAC = H(K \oplus OPAD | H(K \oplus IPAD | m))$

(RFC2104: $OPAD = \texttt{0x5c5c5c} \ldots \texttt{5c5c}$, $IPAD = \texttt{0x363636} \ldots 3636$))

- Autenticazione piú forte che basata su IP sorgente
- Ogni pacchetto che arriva a tun/tap è già stato controllato

Perché HMAC

È opportuno usare HMAC e non MAC piú semplici perché essendo i pacchetti di dimensione fissa sarebbero possibili attacchi del tipo hash extension, se l'hash è di tipo Merkle-Damgård (MD4-5, SHA0-2)

- $H(K|m)$: extension attack se nota la lunghezza di m
- $H(m|K)$: meglio, ma dipende fortemente dalle tecniche di collisione di H (birthday attack)
- $H(K|m|K)$ ancora meglio, ma manca una dimostrazione del grado di efficacia

Extension attack

Reso famoso da un attacco a Flickr (2009. . . vulnerabilità nota dal 1992!)

Se si sa $len(K|m)$ è possibile generare $H(K|m|m')$ senza conoscere K. Il motivo è che gli H Merkle-Damgård spezzano lo stream in blocchi e applicano una funzione di compressione ad ogni blocco indipendentemente.

Con lunghezza si sa quale funzione applicare a blocchi aggiunti dall'attaccante (estensione).

Riassumendo

OpenVPN permette di costruire VPN

- autenticazione con chiave condivisa o certificati
- tunnel UDP o TCP

Privacy e web

Caricare una pagina web da un server HTTP espone moltissimi *dati personali* (alcuni addirittura *sensibili*)

- IP del client (pseudonym)
- IP del server (il client è interessato a quel server)
- identità (vedi `https://panopticlick.eff.org`)
- Dati del browser (history, cookie,...)
- System profile
- Dati trasmessi con form,...

Chi è il nemico?

1. Un eavesdropper può osservare il traffico: anche quando è criptato (`https`) gli IP e il system profile è accessibile
2. Il server web conserva i dati riguardanti il client
3. Il gestore della rete osserva e/o censura il traffico

Come ci si difende?

1. HTTPS
2. Proxy, cambiando proxy server (e browser!)
3. Onion Routing

Proxy

```
...
REMOTE_ADDR = 194.85.1.1
HTTP_ACCEPT_LANGUAGE = ru
HTTP_USER_AGENT = Mozilla/4.0
HTTP_HOST = www.webserver.ru
HTTP_VIA = 194.85.1.1 (Squid/2.4.STABLE7)
HTTP_X_FORWARDED_FOR = 194.115.5.5
...
```

proxy

Transparent proxy `HTTP_X_FORWARDED_FOR` è l'IP del client originale!

Anonymous Proxies IP nascosto

Distorting Proxies IP falso

High Anonymity Proxies `HTTP_VIA` vuoto

Privoxy

Privoxy è un proxy web con avanzate capacità di filtraggio progettato per la privacy

- gestisce i cookie
- javascript
- personalizzabile

Questa categoria di prodotti è detta protocol cleaner

Onion Routing

Onion Routing (OR) è una tecnica sviluppata dal Naval Research Laboratory di Washington.
Il traffico viene instradato in una serie (mutevole) di *onion router* in maniera tale da rendere difficile il tracciamento delle attività.
Gli onion router costituiscono dei mix di Chaum.

Mix di Chaum

Mix di Chaum

Un mix riceve messaggi di lunghezza fissa, li cripta, aspetta di averne in numero sufficiente da garantire un certo livello di anonimato e inoltra i messaggi (in ordine arbitrario) ad altri mix.

Onion router

- Gli onion router intermedi non hanno informazione sufficiente per tracciare mittente/destinatario e traffico
- Rimane una certa criticità degli exit node e (minore) dei nodi d'entrata

Riassumendo

L'anonimato nella navigazione web è un problema piuttosto complesso. Varie difese:

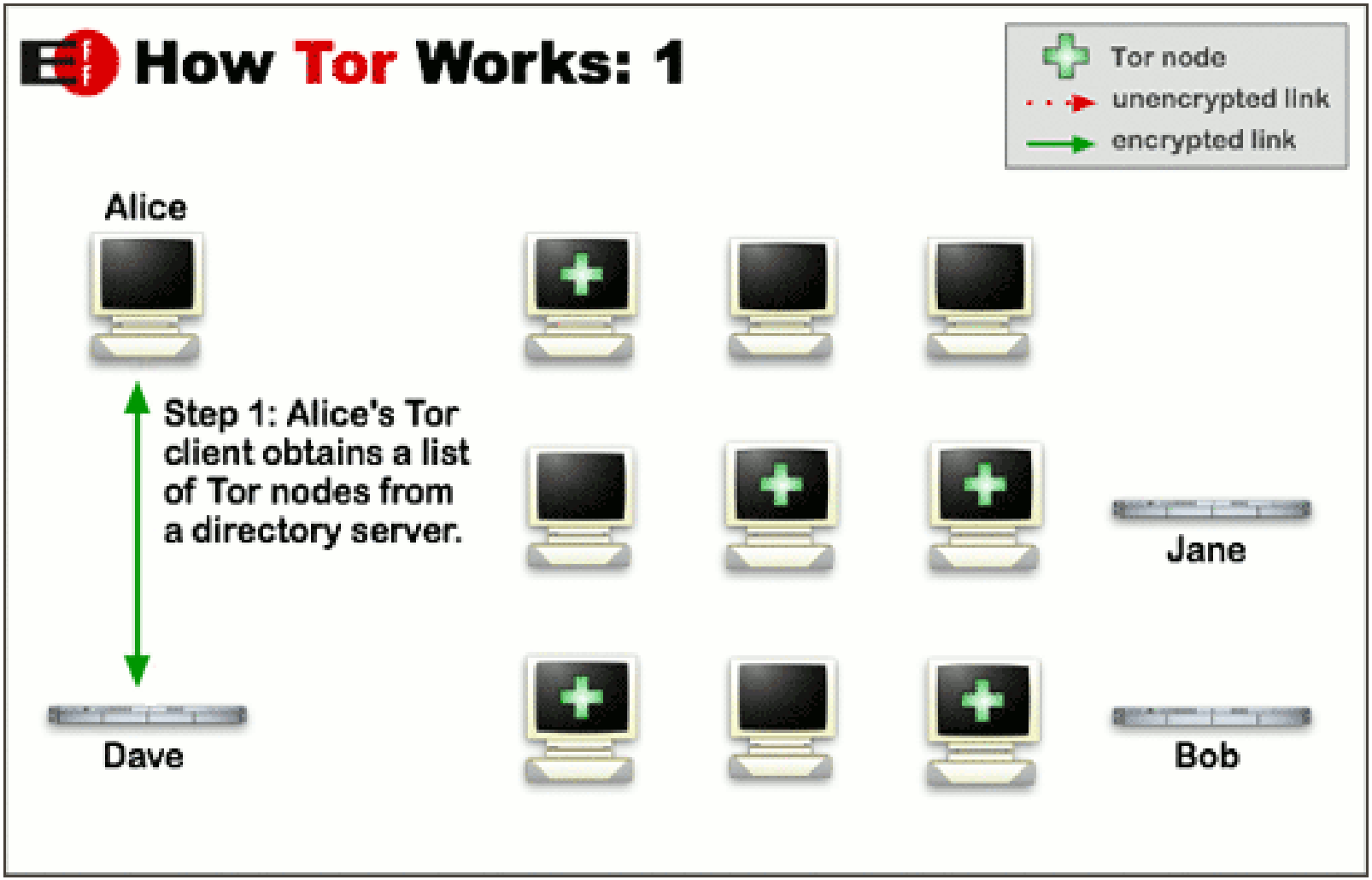
- HTTPS
- Proxy
- Onion Routing

TOR

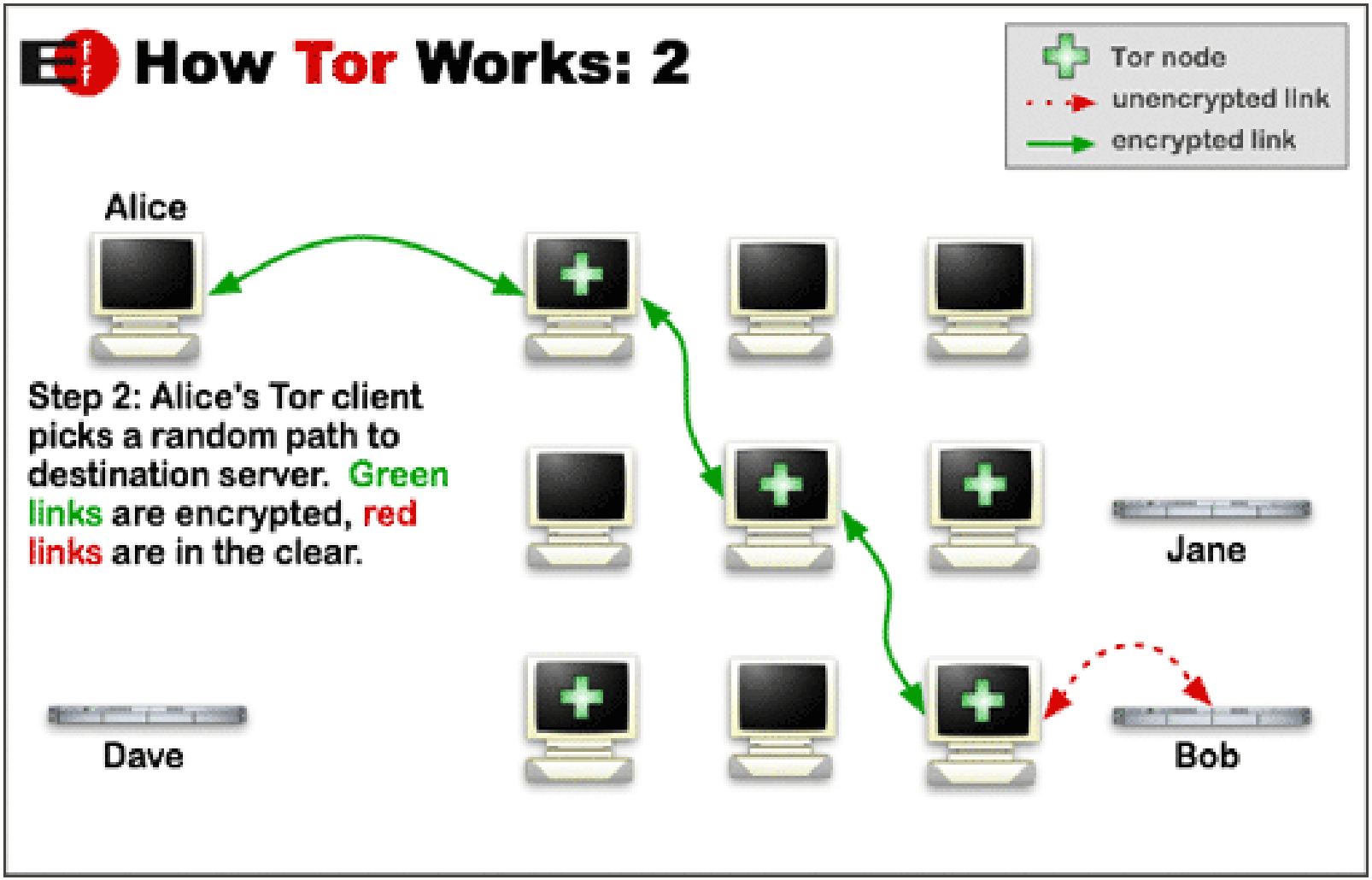
TOR (The Onion Router project) è l'evoluzione piú recente del concetto di OR. Sviluppato da NRL, open source, supportato da EFF.

Il progetto fa parecchi sforzi per rendere il prodotto comprensibile e utilizzabile anche da utenti inesperti (il numero di utenti è un valore per l'anonimato).

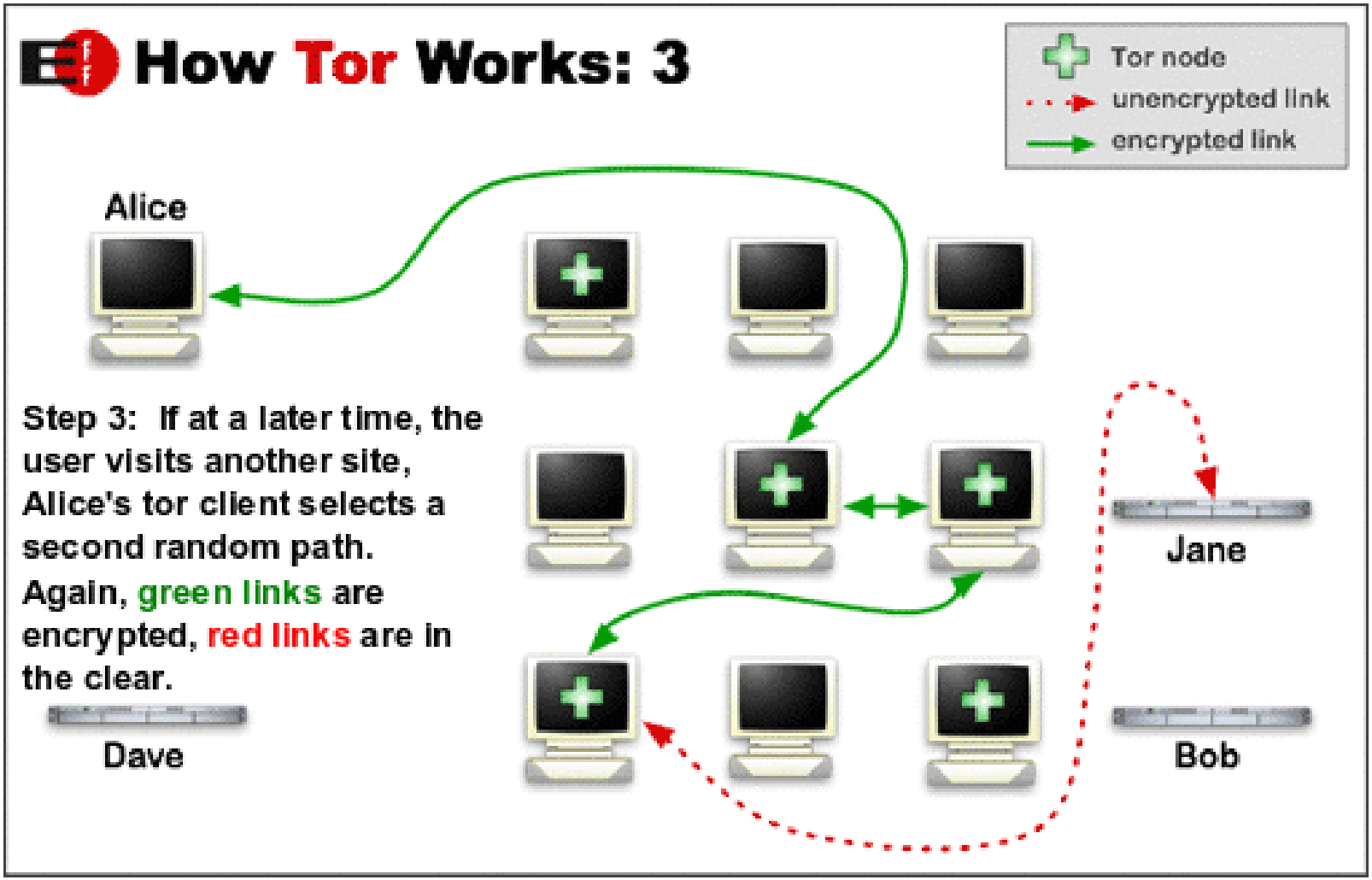
Onion routing con TOR



Onion routing con TOR



Onion routing con TOR



Circuiti

L'instradamento di ogni messaggio viene detto circuito

- Ogni nodo del circuito conosce solo il nodo precedente e successivo (non origine e destinazione)
- Molte richieste diverse vengono multiplexate in un unico circuito
- Robusto rispetto alla compromissione o l'introduzione di onion router malevoli
- Nodi trusted operano da *directory server* iniziali

Struttura di una rete TOR

- Nodi utenti hanno un Onion Proxy (OP)
- Onion Router (OR) connessi tra loro con TLS
- Gli OR hanno una long-term key e short-term "onion" key
- L'unità di trasmissione è la cella, di dimensione fissa di 512 byte

Uso delle chiavi

La *long-term identity key* viene usata per

- firmare la descrizione del router: certificati TLS, chiavi, metadati, *exit policy*
- firmare gli elenchi di router

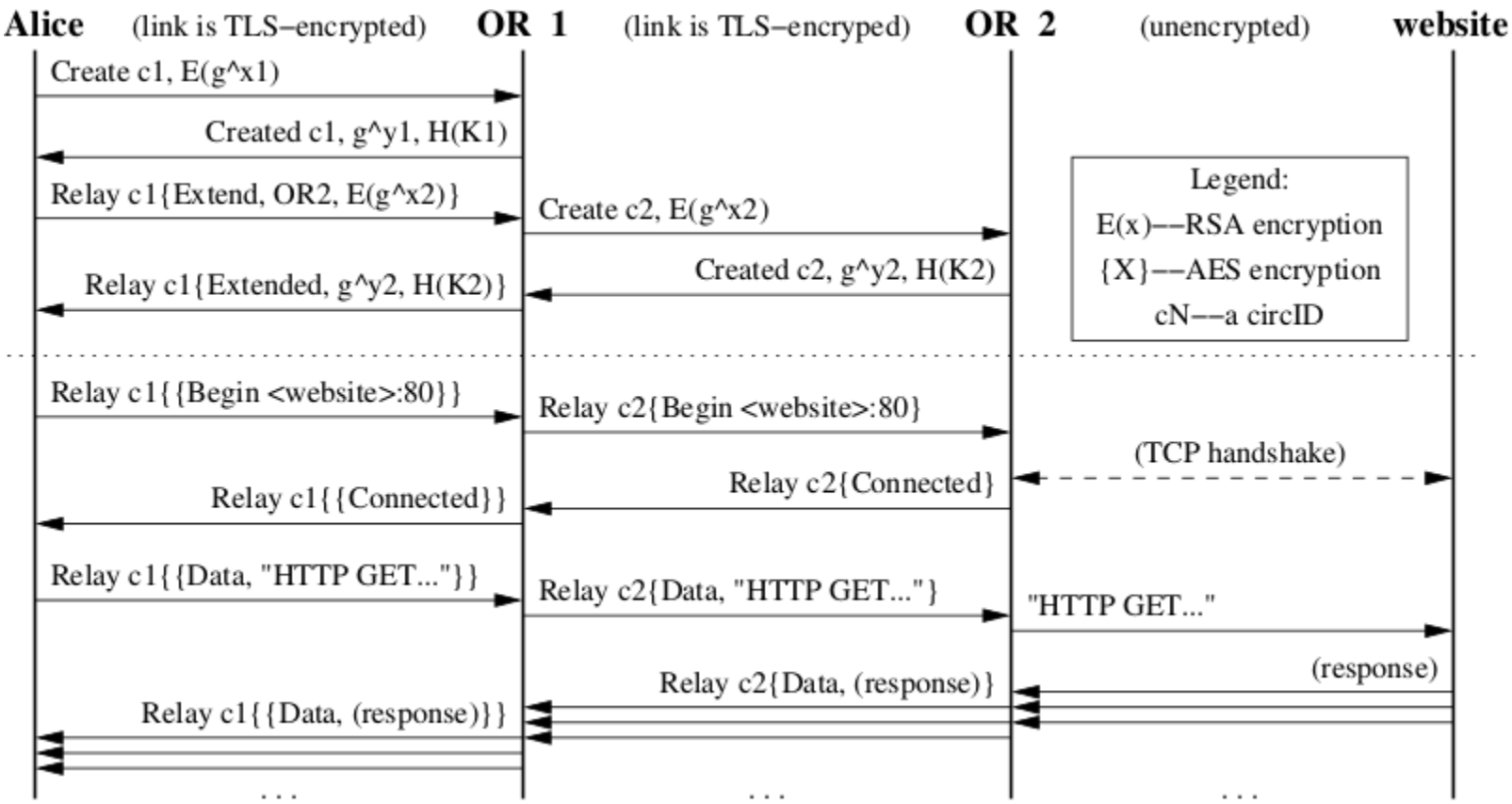
La *short-term key* (onion key) per:

- decrittare le richieste di circuiti
- negoziare chiavi una tantum (ephemeral key) che garantiscono la *forward secrecy*

Circuiti

- l'OP costruisce un circuito in background, e diversi stream utente vengono multiplexati sullo stesso circuito
- Ogni minuto viene creato un nuovo circuito

Creazione di un circuito



La prima fase *estende* il circuito, poi c'è il relay del traffico

Creazione dello stream

L'OP sceglie quale OR può fare da exit node (ognuno ha una *exit policy*)
Solo TCP stream possono essere creati (UDP, e quindi le risouzioni DNS, rimangono problematiche: vedi `http://code.google.com/p/torsocks/` per una soluzione)
Un protocol cleaner è necessario per evitare che informazioni rilevanti finiscano nello stream.

Riassumendo

TOR

- il maggior progetto di Onion Routing
- permette la creazione di circuiti anonimi
- necessita di un protocol cleaner

Lezione XXII: Reti p2p e privacy

Reti p2p

peer-to-peer

Un gruppo di nodi che opera sia come client che come server (ogni nodo è in grado di svolgere le stesse operazioni)

Napster-like	un server centrale conserva un indice dei servizi
Gnutella-like	anche l'indice è distribuito fra i peer (eccetto un elenco di bootstrapping)

Privacy delle operazioni

- L'indice (chi fornisce che cosa) è sostanzialmente pubblico
- La fruizione del servizio generalmente è HTTP (vedi privacy web)
- In alcuni casi (p.es. BitTorrent) i metadati contengono molte informazioni personali
- Potrebbero essere necessarie anche operazioni in cui non si è direttamente interessati (In Svizzera p.es., dove è permesso il download di materiale protetto da copyright, è vietato condividerlo)

Freenet

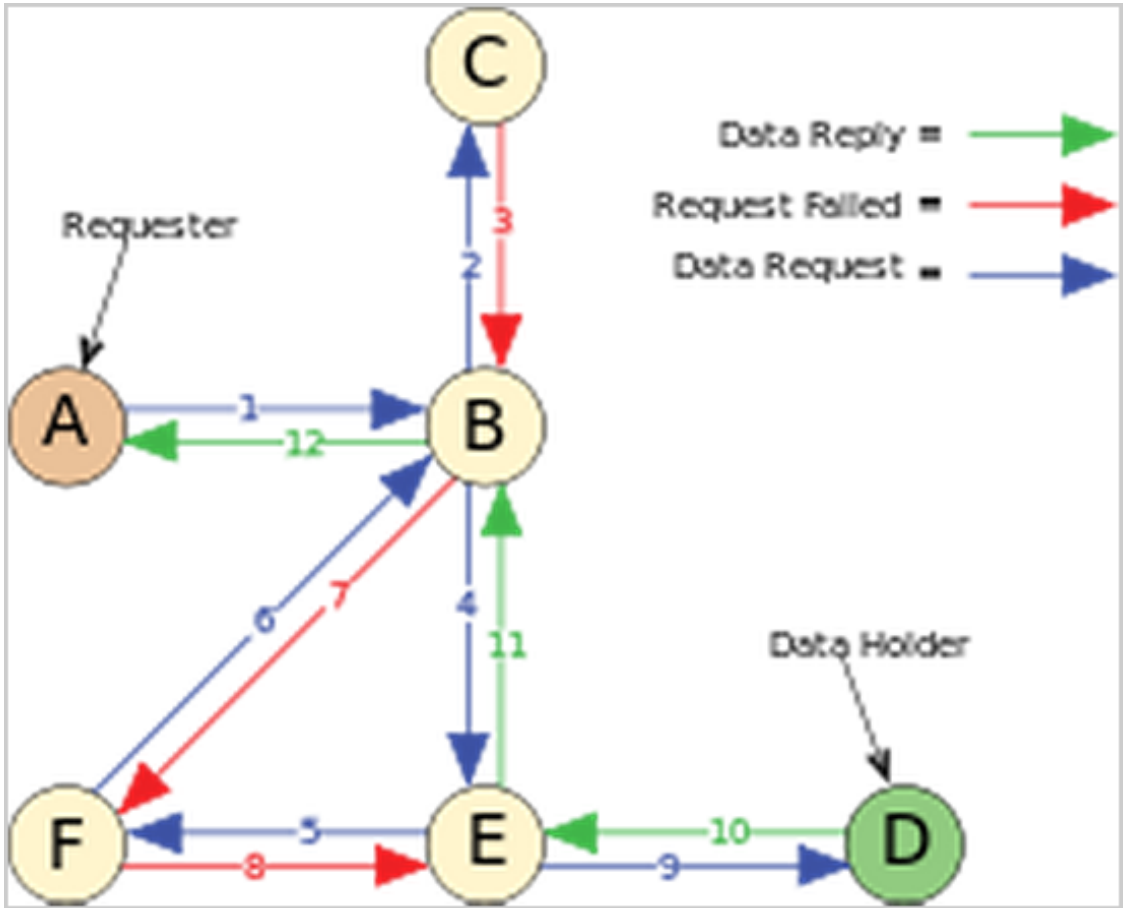
Un tentativo di realizzare un sistema di pubblicazione di contenuti resistente alle censure

- peer-to-peer e completamente decentralizzato
- i dati vengono criptati e replicati su molti nodi
- diventa estremamente difficile sapere chi ha che cosa
- i singoli nodi non hanno modo di sapere cosa mettono a disposizione

Architettura di Freenet

- Ogni contenuto è identificato solo da un hash SHA-256 (non c'è supporto diretto alle ricerche)
- Ogni nodo "conosce" solo un numero ristretto di altri nodi che può raggiungere direttamente
- I contenuti vengono passati ai vicini (e posti in una cache locale), senza sapere se è la destinazione finale
- key-based routing euristico

Freenet routing



Freenet

- Un nodo inserisce un file *nella rete*: a quel punto può anche disconnettersi, perché il file viene spezzato e conservato fra i peer attivi
- I contenuti piú richiesti vengono inseriti piú frequentemente nelle cache (mentre quelli non richiesti tendono a sparire)
- Opennet (chiunque può connettersi) e Darknet (rete fra trusted node con topologia manuale)

Freenet

Lettura obbligatoria:
Clarke, Ian, et al. “Protecting free expression online with Freenet.” Internet Computing, IEEE 6.1 (2002): 40-49.

I2P

Invisible Internet Project (I2P) è una rete per servizi anonimi (con possibilità di gateway verso l'internet tradizionale).

- Inizio nel 2003, parziale spin-off di Freenet e Invisible IRC
- Si tratta di una "overlay network": la comunicazione avviene tramite *I2Ptunnel* (equivalenti ai circuiti TOR)
- I tunnel vengono cambiati ogni dieci minuti
- Le applicazioni per usare I2P devono essere riscritte, utilizzando un'apposita API (Simple Anonymous Messaging oppure Basic Open Bridge)

eepsite

I siti web di I2P vengono chiamati eepsite e sono identificate da chiavi crittografiche (anziché numeri IP): esiste anche una forma simbolica (dominio `.i2p`).

- un *eeproxy* è necessario per collegarsi agli *eepsite* con un normale browser
- la topologia della rete e la risoluzione dei nomi simbolici avviene tramite un netDB: una base di dati distribuita gestita con modalità DHT simili a quelle viste per Freenet

Obiettivi e limiti

I2P è complementare a TOR (che prevede una modalità simile tramite gli "hidden service"): l'obiettivo è creare una rete alternativa il piú possibile anonima.

- Sono noti attacchi "Sybil" che permettono di controllare il netDB controllando una porzione di nodi (2%-20%)
- È molto facile da usare, ma non ha la massa critica (e quindi il suo potenziale di anonimato) di TOR

Bitcoin

Si tratta di una moneta scritturale.
L'obiettivo del progettista (Satoshi Nakamoto, 2008):

Bitcoin

Due soggetti possono direttamente concordare una transazione, *senza la necessità di una terza parte fidata.*

- La transazione non può essere annullata/ripudiata
- Il sistema funziona correttamente nell'ipotesi che gli "onesti" controllino collettivamente piú potenza di calcolo dei potenziali disonesti.

Come si usa

(App Android: Mycelium)

https://github.com/mycelium-com/wallet

1. Serve un indirizzo (un identificatore di una coppia di chiavi crittografiche: può essere generato autonomamente)
2. Servono bitcoin, ottenibili tramite:
   - beni, servizi, altra moneta
   - "mining": produzione di nuovi bitcoin usando potenza computazionale
3. Si indica l'indirizzo di un destinatario
4. Eventualmente proponendo un premio per la chi collaborerà alla garanzia della transazione (transaction fee)
5. Si invia la transazione che verrà validata in una decina di minuti

Come fa a funzionare

Bitcoin stabilisce un protocollo per mantenere un "log" distribuito di tutte le transazioni, in modo che sia possibile sapere se lo stato di ogni "moneta", garantendo che non venga spesa piú volte simultaneamente.
Le scritture contabili sono mantenute coerenti senza un'autorità centrale:

- Crittografia asimmetrica (firme digitali)
- Catene di hash-crittografici
- Timestamp garantiti da computazioni onerose
- Pubblicità totale (sincronizzata tramite bittorrent)

Transazioni

Un transazione è un messaggio che dice:

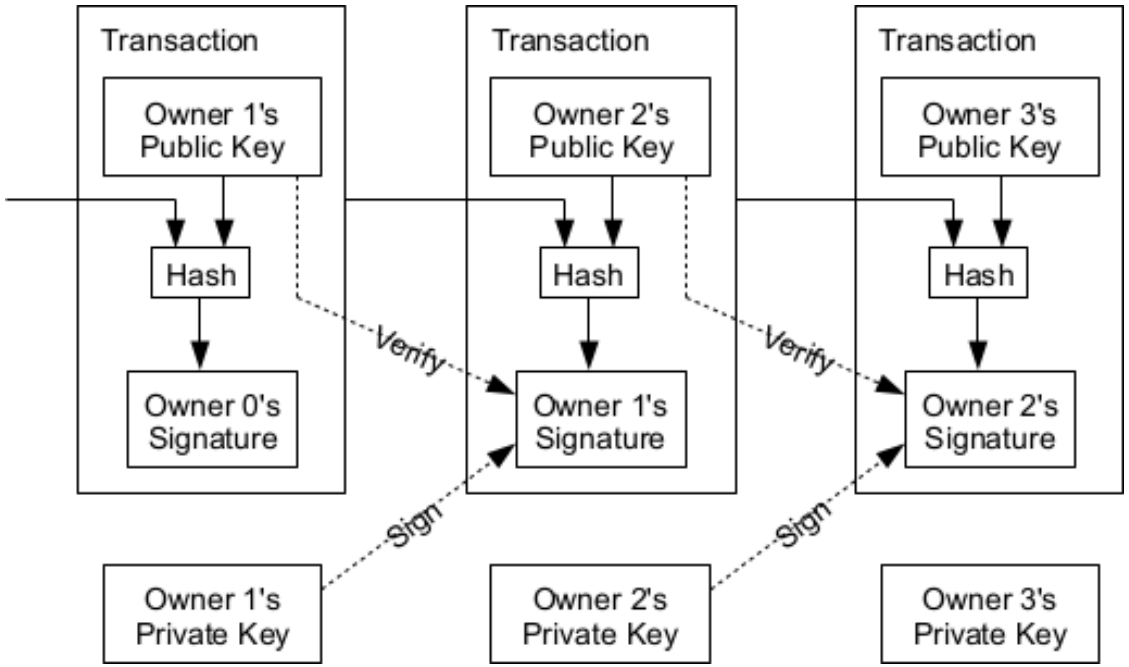
- Il soggetto A cede x bitcoin
- Il soggetto B riceve y bitcoin
- f bitcoin servono come premio per chi collabora alla validazione della transazione (*transaction fee*)

Naturalmente: $x = y + f$

Transazioni firmate

Ogni soggetto ha una coppia di chiavi asimmetriche: una (privata) serve per garantire l'autenticità (firma), l'altra (pubblica) per verificare le firme.

Owner 1 $\rightsquigarrow$ Owner 2



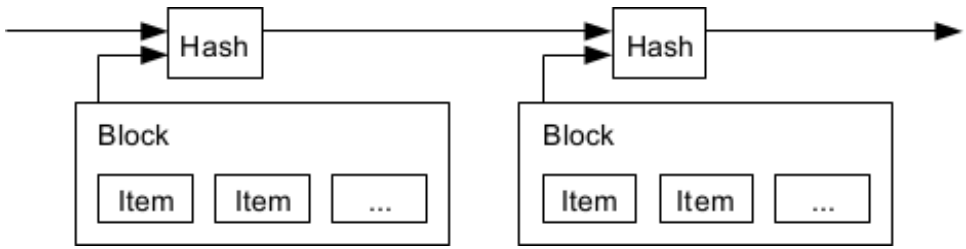
Hash chain

Ogni transazione (in realtà un blocco contiene generalmente molte transazioni) è collegata a quelle precedente perché include uno hash (256 bit che "riassumono" l'informazione in una maniera difficile da falsificare) di quelle precedenti: P.es.:

Hash SHA256 della Divina Commedia curata da G. Petrocchi →

`5b57a696ac3bdb48cb09b1d0998f9d582660f5cbd9463e2ef5d5ea4e0f6d5671`

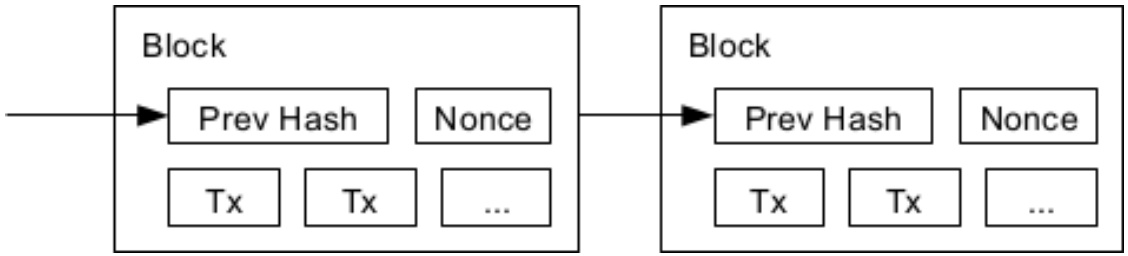
Al momento non si conosce un metodo per trovare un'altra stringa di caratteri con lo stesso hash piú efficiente del provare a caso.



Per calcolarlo devono esistere gli hash precedenti: se H_0 viene pubblicato il 1 gennaio 2014, la transazione che contiene lo hash di H_0 deve essere temporalmente successiva.

Accordo distribuito

Ma come si fa a concordare una singola storia? Un'unica block chain?
Il meccanismo con cui si risolve questo problema (in generale insolubile!) è l'introduzione di proof of work: non basta calcolare uno hash, lo si vuole anche "particolare":



Bisogna trovare un 'nonce' che dia luogo a uno hash che inizia con un certo numero (parametro di difficoltà) di zeri.

`0000000000000001237535293c120a0b9d4d4ac7bac9911c48357bf0f694d26`

Se SHA256 mantiene le sue promesse, non c'è modo migliore che quello di provare a caso... Siccome però ci provano in molti, il tempo in media col quale lo si trova è 10 minuti.

Bitcoin miners

Ma perché dovrebbero provarci in molti?

Perché c'è un premio per chi ci riesce: attualmente 25BTC, dimezzato circa ogni 4 anni.
Chi riesce a trovare un nonce che dà luogo a uno hash opportuno può intestarsi una transazione da 25BTC piú i transaction fee di tutte le transazioni nel blocco.
Nel caso (abbastanza improbabile) che ci siano piú blocchi validi, si prende il ramo con il maggior sforzo computazionale (la catena piú lunga).

Avendo sufficiente potenza computazionale è possibile accreditare transazioni false, ma l'ipotesi è che: (1) gli "onesti" siano computazionalmente piú potenti; (2) "conviene" usare la computazione per ottenere i premi di mining.

Il protocollo

1. Firmo una transazione e la annuncio *broadcast*
2. Ogni nodo disponibile al mining colleziona gli annunci in un blocco
3. Ogni miner cerca un nonce per la *proof of work*
4. Chi trova la *proof of work* la annuncia broadcast
5. L'annuncio del blocco validato viene confermato e la transazione può essere considerata genuina.

Riassumendo

- L'anonimato non è un obiettivo di progetto: anche se le transazioni avvengono fra pseudonimi
- Il numero di bitcoin è limitato ($21 \cdot 10^6$ frazionabili fino a 10^{-8})
- La difficoltà di mining è un parametro del sistema: il tasso di creazione di moneta può essere controllato (fine prevista 2140).
- Le transazioni sono irreversibili: si tutela il venditore, ma non il compratore (il contrario di quanto dovrebbe avvenire con le carte di credito...)

"È virtuale".

da Internazionale 1038

Riferimenti

- Satoshi Nakamoto, *Bitcoin: A Peer-to-Peer Electronic Cash System*, 2008
- Khan Academy: https://www.khanacademy.org/economics-finance-domain/core-finance/money-and-banking/bitcoin/v/bitcoin-what-is-it
- Esplorare la block-chain: http://blockexplorer.com
- Conviene il mining: http://tpbitcalc.appspot.com/

Lezione XXIII: Esercizi riassuntivi

Fermare un worm

- il worm apre un ftp server 6666/tcp
- il worm cerca host che ospitano un servizio vulnerabile su 0.0.0.0/8 192.168.0.0/16, porta 513

1. Scrivere regole Iptables che impediscano il traffico collegato alle attività del worm.

Analisi di traffico

Esaminare il traffico in `evidence.pcap`

1. Identificare e descrivere le interazioni avvenute
2. Identificare potenziali interazioni anomale

Iptable rules

Spiegare l'effetto delle seguenti regole

```
1  Chain PREROUTING (policy ACCEPT 33 packets, 1604 bytes)
2   pkts bytes target prot opt in out source destination
3     33 1604 sshuttle-12300 all -- * * 0.0.0.0/0 0.0.0.0/0
4
5  Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
6   pkts bytes target prot opt in out source destination
7
8  Chain OUTPUT (policy ACCEPT 24 packets, 1568 bytes)
9   pkts bytes target prot opt in out source destination
10    48 3008 sshuttle-12300 all -- * * 0.0.0.0/0 0.0.0.0/0
11
12 Chain POSTROUTING (policy ACCEPT 48 packets, 3008 bytes)
13  pkts bytes target prot opt in out source destination
14
15 Chain sshuttle-12300 (2 references)
16  pkts bytes target prot opt in out source destination
17     0 0 RETURN tcp -- * * 0.0.0.0/0 127.0.0.0/8
18    24 1440 REDIRECT tcp -- * * 0.0.0.0/0 0.0.0.0/0 TTL match TTL != 42 redir ports 12300
```