

Appunti integrativi di Sistemi e Reti

Classe 5a a.s. 2018/2019

*Maria Grazia Maffucci
2017 - 2018 - 2019*



This work is licensed under the Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License. To view a copy of this license, visit <http://creativecommons.org/licenses/by-nc-sa/4.0/>.

Contact mariagrazia@maffucci.cc

Indice

Forwarding e algoritmi di routing	14
Libro vol 2 - Il routing: protocolli e algoritmi	15
Routing e forwarding	15
Algoritmo Distance Vector	16
Fusione e generazione del Distance Vector	18
L'algoritmo Distance Vector	19
Esempio: Cold Start	20
Esempio: caduta di un link	21
Problemi del Distance Vector e possibili soluzioni	22
Video riepilogativi	24
Link State	25
Caratteristiche dell'algoritmo Link State	26
Video riepilogativi	29
Libro vol.2 - Routing gerarchico	29
Video riepilogativi	29
Il livello di trasporto	30
Libro vol.2 - Lo strato di trasporto	31
Assegnazione e traduzione degli indirizzi	32
Prefazione	33
Libro vol.2 - Assegnazione degli indirizzi tramite DHCP	34
NAT (Network Address Translation)	35
NAT e PAT (vol.2)	35
NAT	35
La NAT table	36
Porte inside e outside	36
NAT: sicurezza e amministrazione	38
Tecniche di traduzione degli indirizzi	38
NAT statico	38
NAT dinamico	40
NAT overload o PAT	41
PAT statico	43
Il livello Applicazione	45
Prefazione	46
Libro vol.3 - Il livello delle applicazioni	47
Il protocollo HTTP	47
Libro vol.3 - Il livello delle applicazioni	47
File Transfer Protocol: FTP	47
Libro vol.3 - Il livello delle applicazioni	47
Domain Name System: DNS	47
Libro vol.3 - Il livello delle applicazioni	47

DNS - Il servizio Directory di Internet	47
Servizi forniti dal DNS	48
DNS: funzioni di rete critiche con il paradigma client-server	50
Panoramica del DNS	50
Un Database Gerarchico Distribuito	51
Interrogazioni iterative e ricorsive	52
DNS Caching	54
Record DNS [leggere]	55
Vulnerabilità DNS	56
La posta elettronica	58
Libro vol.3 - Il livello delle applicazioni	58
Protocols and other networking concepts - CLIL	59
CLIL Listening and Writing - Protocols	60
CLIL Listening - And other networking concepts	61
VLAN: Virtual LAN	62
Prefazione	63
VLAN: Virtual Local Area Network	64
Libro vol.3 - Le Virtual LAN (VLAN)	64
VLAN - Definizione e ambiti di applicazione	64
Configurazione di una VLAN	68
1 - Laboratorio VLAN: simulazione con Packet Tracer	68
Creazione VLAN	69
Associazione porte - VLAN	69
Rimozione porte e VLAN	70
Rimozione di una porta da una VLAN	70
Rimozione di una VLAN	70
Some useful videos - VLAN Concepts	70
Comprehension Activity	71
2 - VLAN - Access/Untagged ports	71
VLAN Trunking	72
3 - Laboratorio: configurazione porte trunk con Packet Tracer	74
Configurazione di un link trunk	74
Some useful videos - VLAN Trunking	75
Comprehension Activity	75
4 - VLAN - Configuration of a virtual switch interface	75
5 - VLAN - Virtual switch interface protection, switch remote configuration and trunking principles	75
6 - VLAN - VLAN trunking configuration	75
Approfondimenti	76
Routing inter-VLAN	77
Router con più collegamenti fisici	77
Router-on-a-stick	78
7 - Laboratorio: inter-VLAN routing con un router-on-a-stick	79

Comprehension Activity	80
8 - VLAN - Router-on-a-stick	80
Switch Layer 3	81
Passi per configurare l'inter-VLAN routing con switch L3	82
9 - Laboratorio: inter-VLAN routing con uno Switch Layer 3	83
Comprehension Activity	85
10 - VLAN - Multilayer Switching	85
VTP (VLAN Trunking Protocol)	86
11 - Laboratorio VTP: Switch server, client e transparent	88
Comprehension Activity	90
12 - How VTP works	90
13 - VLAN - Activity Overview	90
14 - Subnetting e VLAN	91
Sicurezza informatica e crittografia	92
Prefazione	93
Sicurezza in rete	94
La sicurezza delle reti	94
Libro vol.3 - La sicurezza nei sistemi informativi	94
Crittografia simmetrica	95
Tecniche crittografiche per la protezione dei dati	95
Libro vol.3 - La crittografia simmetrica	95
Esercizi sulla crittografia simmetrica	96
Esercizi	96
Crittografia asimmetrica	97
Tecniche crittografiche per la protezione dei dati	97
Libro vol.3 - La crittografia asimmetrica	97
RSA	97
Concetto di base di RSA	99
Esempio di criptazione e decriptazione usando RSA	100
La generazione delle chiavi	101
Esempio di generazione delle chiavi di RSA	102
Esempio sull'uso di RSA	103
Esercizi sull'uso di RSA	105
Esercizi sulla crittografia asimmetrica	106
Esercizi	106
Autenticazione e affidabilità	107
Tecniche crittografiche per la protezione dei dati	107
Libro vol.3 - Certificati e firma digitale	107
Protocolli per la sicurezza della rete	109
Prefazione	109
I protocolli sicuri	111
IP Security (cenni)	114
IPSec	115

VPN (Virtual Private Network)	117
La rete VPN	117
Modalità di connessioni di una VPN	118
Remote-access VPN	118
Site-to-site VPN	119
La sicurezza nelle VPN	120
Authentication, Authorization, Accounting	121
Cifratura	121
Tunneling	122
Tipi di VPN	124
Trusted VPN	124
Secure VPN	125
Hybrid VP	126
TLS/SSL	126
La sicurezza delle reti	127
Libro vol.3 - La sicurezza delle connessioni con SSL/TLS	127
TLS - Transport Layer Security	127
Fase 1: Handshake Protocol	128
Fase 2: Record Protocol e trasferimento dati	131
S/MIME e posta elettronica	133
La sicurezza delle reti	133
Libro vol.3 - La sicurezza nei sistemi informativi	133
Sistemi di pagamento elettronico	134
La sicurezza perimetrale	136
Prefazione	136
La sicurezza delle reti: Packet Filtering Firewall	138
Libro vol.3 - Firewall, Proxy, ACL e DMZ	138
Packet Filtering Firewall o stateless firewall	138
Esempi di Packet Filtering Firewall	140
Drop di ICMP Request dall'interno della rete	140
Drop di connessioni SSH dall'esterno della rete	142
Drop di connessioni a Web esterno alla rete	144
Accept delle sole connessioni a Web esterno alla rete	147
Considerazioni sull'azione di default	149
La sicurezza delle reti: Stateful Firewall	150
Libro vol.3 - Firewall, Proxy, ACL e DMZ	150
Stateful Firewall	150
Esempi di Stateful Packet Filtering	152
Accept delle sole connessioni a Web esterno alla rete	153
Analisi di pacchetti	154
La sicurezza delle reti: Server Proxy Firewall	157
Libro vol.3 - Firewall, Proxy, ACL e DMZ	157
Server Proxy Firewall	157

Application Proxy o Application-level Gateway	158
Circuit-Level Proxy Firewall o Circuit-Level Gateway	160
Access Control List [leggere]	161
Introduzione	161
Laboratorio: ACL standard outbound	162
Laboratorio: ACL standard inbound	164
Laboratorio: ACL estese	165
Laboratorio: ACL riferite a servizi	166
Laboratorio: DMZ	168
Sistemi distribuiti e tecniche di amministrazione	171
Modelli distribuiti per i servizi di rete e amministrazione dei sistemi	172
Libro vol. 3 - Modello client-server e distribuito per i servizi di rete	172
Macchine e servizi virtuali	173
La Virtualizzazione	173
CLIL - What Virtualization is	174
Le ragioni della virtualizzazione	174
Architetture delle macchine virtuali	176
Layer fisico e layer virtuale	176
Hypervisor o VMM	176
Hypervisor 1 (Bare Metal o Native)	177
Hypervisor 2 (Hosted)	178
La gestione dell'ambiente virtuale	179
L'host fisico	179
La gestione dello storage	179
CLIL - DAS, NAS e SAN	180
Gestione delle ridondanze	180
Il virtual networking	180
I driver della macchina virtuale	182
Configurazione dell'hardware virtuale	182
Virtual data center	182
CLIL - Google Data Center	183
I data center	183
Virtual data center	184
Dal data center al virtual data center	184
Cluster di failover	185
Migrazione di macchine virtuali	185
Migrazione dello storage	186
Snapshot, clonazione e template	186
Backup del data center virtuale	187
Cloud computing	187
Le cinque caratteristiche del cloud	188
I tre modelli di servizi del cloud	190
CLIL - SaaS, PaaS and IaaS	191

Cloud privato, pubblico, comunitario e ibrido	191
CLIL - Public, Private and Hybrid Cloud	192
Cloud computing pubblico	192
Cloud computing privato	192
Cloud computing ibrido	193
Approfondimenti	194
A - Preparazione all'Esame di Stato	195
Requisiti fondamentali di Progettazione reti	196
Che cosa significa progettare una rete	197
Definizione dei requisiti	197
Progettazione e pianificazione	198
Realizzazione	198
Gestione e rilascio	198
Traccia per la progettazione di una rete	199
Inquadramento generale	199
Vincoli e progetto della rete	199
Confine con l'esterno	200
Cablaggio strutturato	200
Sistemi di sicurezza adottati	200
Gestione dei servizi	201
Gestione della rete	201
Sviluppo applicazioni	201
Prove d'esame	201
Scritto	201
Esercizi sulle prove d'esame	202
Colloquio	206
B - IPv6 Network Addresses - CLIL	212
Prefazione	212
IPv4 Issues	213
The Need for IPv6	214
Writing/Speaking Activity - The Need for IPv6	215
IPv4 and IPv6 Coexistence	215
Writing/Speaking Activity - IPv4 and IPv6 Coexistence	216
Listening/Writing Activity - What is IPv6?	217
Vocabulary	217
Activity	217
Activity – IPv4 Issues and Solutions	217
IPv6 Addressing	217
IPv6 Address Representation	218
Preferred format	218
Omit Leading 0s	219
Omit All 0 Segment	220

Activity – Practicing IPv6 Address Representations	221
Types of IPv6 Addresses	223
IPv6 Address Types	223
IPv6 Prefix Length	223
Listening Activity - IPv6 Address Types	225
Vocabulary	225
Activity	225
IPv6 Unicast Addresses	227
IPv6 Link-Local Unicast Addresses	227
IPv6 Unique local address	228
IPv6 Global Unicast Address	229
Global Routing Prefix	230
Subnet ID	230
Interface ID	230
Activity – Identify Types of IPv6 Addresses	231
Static Configuration of a Global Unicast Address	232
Router Configuration	232
Host Configuration	232
Dynamic Configuration	234
Dynamic Configuration - SLAAC	234
RA Option 1: SLAAC	235
EUI-64 Process and Randomly Generated	236
EUI-64 Process to create IPv6 address	237
Activity - EUI-64 process	238
Dynamic Link-Local Addresses	239
Static Link-Local Addresses	240
Verifying IPv6 Address Configuration	241
Activity – Link-local connectivity	244
Activity – Global unicast connectivity and SLAAC	246
C - Il livello Applicazione	248
Prefazione	248
Libro vol.3 - Il livello delle applicazioni	250
Introduzione	250
Generalità sulle applicazioni di rete	251
Architettura delle Applicazioni di rete	251
Comunicazione tra Processi	252
Processi Client e processi Server	252
L'interfaccia tra il Processo e la rete di calcolatori	253
Indirizzamento dei Processi	254
Servizi del livello Trasporto disponibili alle applicazioni	254
Trasferimento affidabile o non affidabile dei dati	255
Flusso (throughput)	255
Temporizzazione	256

Sicurezza	256
Servizi di Trasporto offerti da Internet	256
Servizi TCP	257
Sicurezza di TCP	257
Servizi UDP	258
Servizi non forniti dai protocolli di Trasporto di Internet	258
Protocolli del livello Applicazione	259
Il protocollo HTTP	259
Libro vol.3 - Il livello delle applicazioni	261
Introduzione	261
Panoramica di HTTP	261
Protocollo stateless	262
Round Trip Time (RTT)	263
Connessioni Non-Persistenti e Persistenti	264
HTTP con Connessioni Non-Persistenti	264
HTTP con Connessioni Persistenti	265
Formato dei messaggi HTTP	266
Messaggio HTTP request	266
Messaggi HTTP Response	268
File Transfer Protocol: FTP	271
Libro vol.3 - Il livello delle applicazioni	271
Funzionamento generale	271
Comandi e Risposte FTP	272
Domain Name System: DNS	275
Libro vol.3 - Il livello delle applicazioni	275
DNS - Il servizio Directory di Internet	275
Servizi forniti dal DNS	275
DNS: funzioni di rete critiche con il paradigma client-server	277
Panoramica del DNS	277
Un Database Gerarchico Distribuito	278
Interrogazioni iterative e ricorsive	280
DNS Caching	280
Record DNS	282
Vulnerabilità DNS	283
La posta elettronica	284
Libro vol.3 - Il livello delle applicazioni	285
introduzione	285
Caratteristiche principali	285
e-mail su Web	286
SMTP	287
Formati dei messaggi di posta	290
Mail Access Protocol	291
POP3	292
IMAP	294

E-Mail nel Browser	295
D - Sicurezza informatica e crittografia	296
Prefazione	296
Sicurezza in rete - 1a parte	298
La sicurezza delle reti	298
Libro vol.3 - La sicurezza nei sistemi informativi	298
Crittografia simmetrica	299
Tecniche crittografiche per la protezione dei dati	299
Libro vol.3 - La crittografia simmetrica	299
CLIL - Cryptography Introduction	300
CLIL Speaking Activity - Introduction to Cryptography	300
13 - CLIL - Homework	300
CLIL - Substitution ciphers	301
CLIL - Breaking a substitution cipher: brute force attack	301
CLIL Reading Activity - Frequency analysis	302
14 - CLIL Listening, Speaking Activity - Caesar and Vigenère cipher	303
Vocabulary	303
15 - Esercizi	303
CLIL - Transposition cipher	306
16 - CLIL Listening Activity - Transposition cipher	306
17 - Laboratorio - Cifrario monoalfabetico	307
CLIL Listening Activity - Symmetric Cryptosystems	308
CLIL Listening Activity - Correctness	308
CLIL - XOR Cipher and One Time Pad encryption	308
CLIL - XOR Cipher	308
CLIL - One Time Pad encryption method	309
18 - CLIL - Homework	310
Esercizi sulla crittografia simmetrica	311
Esercizi	311
Cenni di teoria dei numeri [da concludere]	312
Teorema di Eulero	312
La funzione toziente di Eulero	312
Esempi sulla funzione toziente di Eulero	312
Teorema di Eulero	313
Esempi sul teorema di Eulero	313
Teorema di Fermat	313
Crittografia asimmetrica	315
Tecniche crittografiche per la protezione dei dati	315
Libro vol.3 - La crittografia asimmetrica	315
Introduzione alla crittografia asimmetrica	316
Il metodo Diffie-Hellman e lo scambio delle chiavi	318
Esempio sull'uso di Diffie-Hellman	320
19 - Esercizi sull'uso di Diffie-Hellman	321

Determinazione delle chiavi identiche in Diffie-Hellman	322
Robustezza dell'algoritmo Diffie-Hellman	323
Esempio di applicazione del metodo dei quadrati ripetuti [leggere]	324
Diffie-Hellman e attacco Man-in-the-middle	325
Esempio di attacco Man-in-the-middle su Diffie-Hellman	326
CLIL - Homework	328
RSA	328
Concetto di base di RSA	330
Esempio di criptazione e decriptazione usando RSA	331
La generazione delle chiavi	332
Esempio di generazione delle chiavi di RSA	333
Esempio sull'uso di RSA	334
Esercizi sull'uso di RSA	336
Esercizi sulla crittografia asimmetrica	337
Esercizi	337
Autenticazione e affidabilità	338
Tecniche crittografiche per la protezione dei dati	338
Libro vol.3 - Certificati e firma digitale	338
Codice di autenticazione dei messaggi (crittografia simmetrica)	339
La firma digitale (crittografia asimmetrica)	340
Le funzioni hash	341
CLIL - Homework	344
I certificati digitali e Autorità di Certificazione	345
Infrastruttura a chiave pubblica (PKI)	346
Lo standard X.509 per i certificati	349
21 - CLIL - Homework	350
22 - Laboratorio su GnuPG	351
E - Protocolli per la sicurezza della rete	355
Prefazione	355
I protocolli sicuri	356
IP Security	360
IPSec	360
Security association e policy	363
Authentication Header	365
Encapsulating Security Payload	367
IKE (Internet Key Exchange)	369
VPN (Virtual Private Network)	372
La rete VPN	372
Modalità di connessioni di una VPN	373
Remote-access VPN	373
Site-to-site VPN	374
La sicurezza nelle VPN	375
Authentication, Authorization, Accounting	376

Cifratura	376
Tunneling	377
Come funziona una VPN	379
Trusted VPN	379
Secure VPN	380
Hybrid VP	381
TLS/SSL	382
La sicurezza delle reti	383
Libro vol.3 - La sicurezza delle connessioni con SSL/TLS	383
TLS/SSL	383
Fase 1: Handshake Protocol	384
Fase 2: Record Protocol e trasferimento dati	387
Sistemi di pagamento elettronico	389
S/MIME e posta elettronica	391
La sicurezza delle reti	391
Libro vol.3 - La sicurezza nei sistemi informativi	391
F - Sistemi distribuiti e tecniche di amministrazione	392
Modelli distribuiti per i servizi di rete e amministrazione dei sistemi	393
Libro vol. 3 - Modello client-server e distribuito per i servizi di rete	393
SSH Secure Shell	393
Sicurezza di SSH	394
Scambio di dati sicuro	395
Autenticazione	395
23 - Esercizi su netacad.com	398
Kerberos	398
RADIUS [rivedere, non mi piace]	403
G - Sicurezza nelle WLAN	406
Prefazione	406
Libro vol.3 - Wireless e reti mobili	408
Collegamento ad una WLAN	408
Scanning o discovery della rete	409
Passive Scanning mode	409
Active Scanning Mode	409
Autenticazione	410
Autenticazione a sistema aperto	410
Autenticazione a chiave condivisa	411
Autenticazione 802.1X	411
Associazione	412
Esercizi	412
H - La comunicazione non cablata	414
Protocolli per la comunicazione non cablata	415
Streaming	415

Sensori	415
Standard IEEE 802.15.4, ZigBee e Bluetooth	416
GSM, UMTS, LTE, 3G, 4G	416
I - Cablaggio strutturato	417
Prefazione	417
Esercizi di ripasso	418
24 - Esercizi generali su netacad.com	419
25 - Subnetting e VLAN	422
Sistemi di cablaggio strutturato	423
Fasi del cablaggio strutturato	424
Specifiche generali delle normative e degli standard	425
Struttura generica di un sistema di cablaggio	426
La topologia del cablaggio	427
Gli elementi del cablaggio	429
Dimensionamento WA	434
Fonia e dati	435
Caratteristica dei mezzi trasmissivi	440
Canale	440
Disposizione degli elementi funzionali	446
Distributori	446
Prese utente	446
Cavi	447
La progettazione del cablaggio strutturato	449
Assegnazione degli indirizzi IP	450
Esempio di cablaggio strutturato	450
Operazioni preliminari, scelte tecniche e posizionamento degli apparati	452
I canali	452
I mezzi trasmissivi	454
Elementi del sistema di cablaggio	454
Il cablaggio orizzontale	455
Assegnare gli indirizzi IP	456
Programmazione degli apparati attivi	457
Collegamenti wireless	457
Un esempio di cablaggio con postazioni wireless	459
Bibliografia e sitografia	461

Forwarding e algoritmi di routing

Libro vol 2 - Il routing: protocolli e algoritmi

STUDIARE: L. Lo Russo, E. Bianchi, Sistemi e Reti - Nuova Edizione OPENSCHOOL - Per l'articolazione INFORMATICA degli Istituti Tecnici settore Tecnologico, vol. 2, ed. Hoepli, 2016

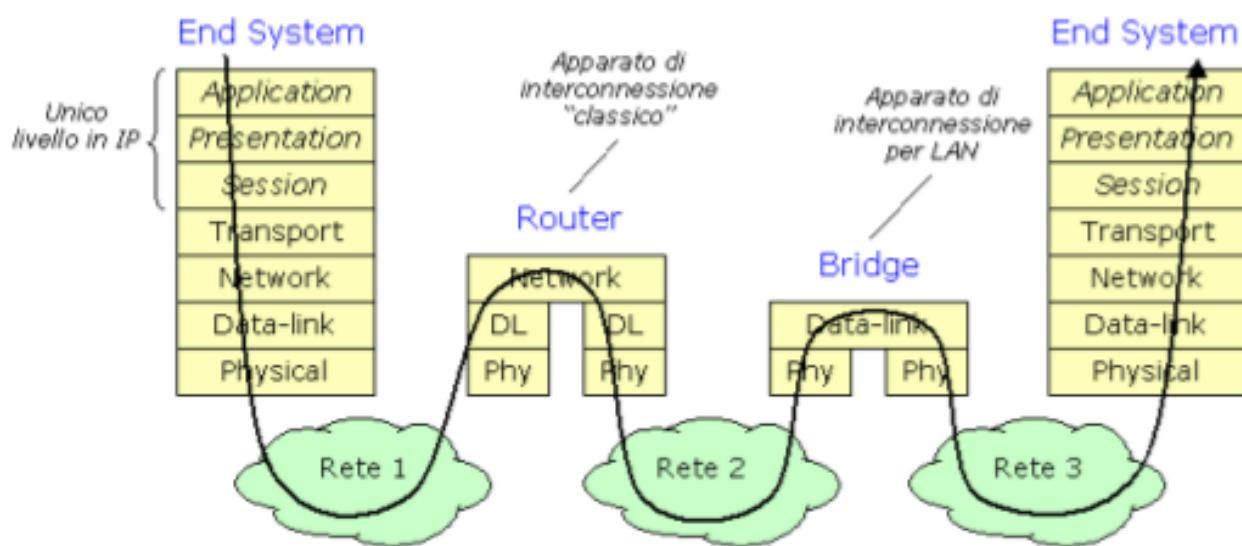
- Fondamenti di routing pp. 134-149
- Routing statico e dinamico pp. 150-155

STUDIARE: si vedano i seguenti video al fine di comprendere gli argomenti trattati e si studino i testi linkati

- [ARP Process](#) di [Sophia Danesino](#) ([video](#))
- [Subnetting vs Supernetting- IP Addressing Subnetting](#) ([video](#))
- [Supernetwork](#)

Routing e forwarding

Il **livello Network** è il livello che si occupa dell'**instradamento end-to-end** dei messaggi su una rete, utilizzando un **indirizzamento univoco**, e **localizza** gli eventuali **instradamenti alternativi** in caso di guasti.



Il **forwarding** e il **routing** sono le due operazioni fondamentali del livello Network affinché possa svolgere il proprio compito in modo adeguato.

Il **processo di routing** (instradamento) si occupa di capire quali sono le destinazioni raggiungibili, facendo in modo che tutti i nodi che partecipano alla rete si accordino su un particolare **algoritmo** (ad esempio in base alla distanza minima tra due punti) che

definisca univocamente i percorsi per giungere a qualunque destinazione a partire da qualunque sorgente. Il risultato finale del processo di routing è la creazione di una serie di informazioni locali (ad esempio la **tabella di routing** in tecnologia IP) in ognuno dei nodi della rete. È possibile immaginare la tabella di routing come ad un insieme di cartelli che specificano la direzione per ogni possibile destinazione.

Il **processo di forwarding** (inoltro) è quella di utilizzare le informazioni disseminate dal precedente processo di routing, che ha creato le tabelle di routing, per **inoltrare i pacchetti verso la destinazione** (tecnologia IP). Una caratteristica importante del processo di forwarding è la **mancanza di conoscenza del percorso globale**. Le informazioni memorizzate in ogni nodo, infatti, specificano banalmente la direzione del prossimo passo ma non il percorso globale. Tuttavia questo non è un problema in quanto i dati verranno inoltrati nella direzione opportuna nodo per nodo (il processo di routing ha già compiuto le operazioni necessarie a sincronizzare queste informazioni locali) e non è necessario conoscere esattamente in anticipo tutto il percorso.

Ambedue i processi (**routing** e **forwarding**) sono **necessari** per l'operatività di una rete. Tuttavia, mentre il **processo di forwarding deve essere obbligatoriamente specificato** da ogni architettura di rete, il **processo di routing può essere demandato all'amministratore**. In altre parole, una architettura di rete può non specificare un modo automatico di disseminazione delle informazioni, ma lasciare all'amministratore della rete la responsabilità di configurare appositamente ogni nodo (ad esempio mediante configurazione statica) per il processo di forwarding.

Per gestire aree molto grosse il processo di routing viene svolto dal **routing gerarchico** che permette di risolvere problemi di scalabilità, dato che su questo tipo di reti i normali algoritmi di routing si rivelano inadeguati. Ciò nonostante il **routing gerarchico**, pur inserendo delle nuove regole di routing tra domini, introduce al tempo stesso nuove problematiche, quali le aree partizionate.

Nonostante le varie tecniche di routing adottabili, neanche il routing gerarchico risulta sufficiente quando si deve gestire una **rete molto grossa con gestori diversificati**; in questo caso entrano in gioco anche **problematiche di policy** (ad esempio ammettere o disabilitare il passaggio di dati per alcune destinazioni in un certo dominio), che vanno affrontate con **algoritmi ad hoc**.

Algoritmo Distance Vector

L'**algoritmo Distance Vector**, noto anche come **algoritmo di Bellman-Ford**, si basa sulla comunicazione tra soli nodi adiacenti, cioè **ogni IS¹ comunica tutte le informazioni di connettività in suo possesso a tutti e solo gli IS adiacenti**.

L'algoritmo costruisce una tabella di routing costituita da due colonne, una contenente il **costo** stimato per raggiungere ogni nodo della rete, e una che specifica l'**interfaccia** da usarsi. Il **costo** può essere calcolato secondo metriche diverse, in base al protocollo in uso. Le righe della tabella diverranno, con il continuo scambio tra IS, tante quanti sono i nodi della rete.

¹ Intermediate System

Il **Distance Vector** è un algoritmo dinamico e le tabelle vengono aggiornate ad intervalli di tempo prestabiliti.

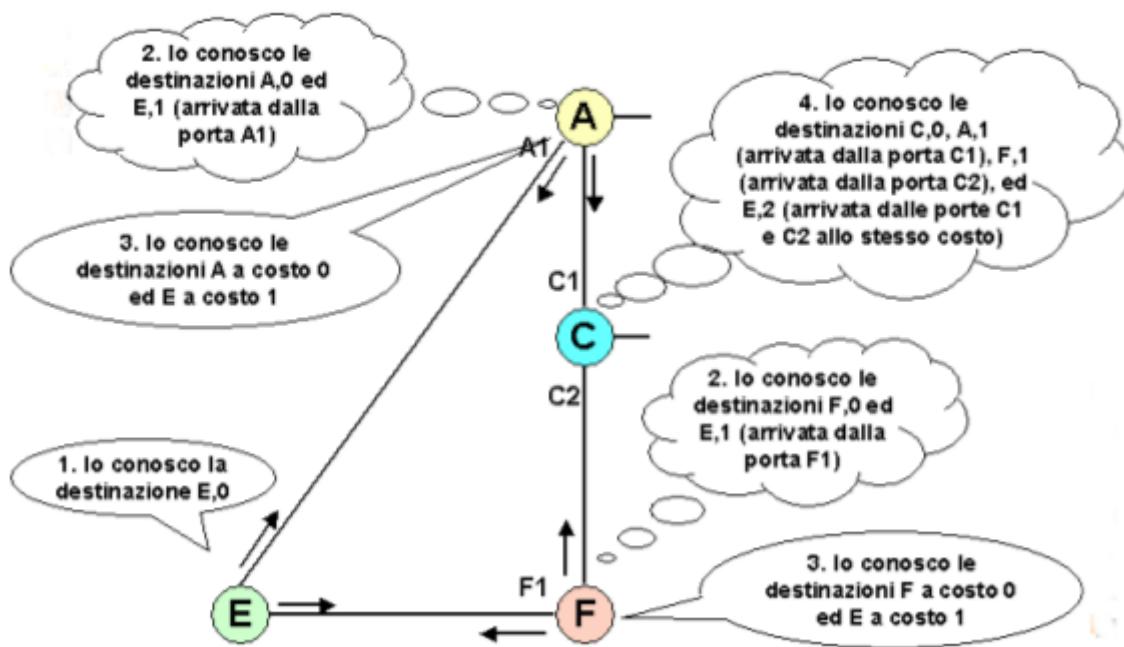
L'algoritmo prevede che inizialmente ogni router invii ai router vicini (**neighbour**²) un pacchetto di **ECHO** per **calcolare la distanza che lo separa da ciascuno di essi**, e inserisca il valore in una tabella apposita, detta **tabella delle adiacenze**.

Subito dopo i router vicini si scambiano un pacchetto contenente il **vettore delle distanze (distance vector)**, cioè **un vettore contenente le informazioni che ciascun router ha a disposizione riguardo i costi per raggiungere le varie destinazioni**. Ricevuti i vettori delle distanze dai vicini, ciascun router aggiornerà la propria tabella delle adiacenze confrontando i costi già presenti nella propria tabella e quelli ricevuti; i valori saranno modificati laddove risultino inferiori, aggiornando se necessario anche le relative interfacce di uscita.

Per capirne il funzionamento si supponga di avere la rete mostrata nella figura sottostante e che, per semplicità, i nodi vengano accesi contemporaneamente (*cold start*).

Il nodo E (analogamente gli altri nodi) comunicherà ai suoi nodi adiacenti (A, F) tutto ciò che è di sua conoscenza, ossia solo sé stesso (il bootstrap è appena concluso). I nodi A ed F riceveranno questo annuncio e introdurranno la nuova informazione fra quelle in loro possesso: esiste una destinazione E e questa si raggiunge da una precisa direzione (attraverso A1 per il nodo A e attraverso F1 per il nodo F). In questo modo sia il nodo A, sia il nodo F capiranno di non essere i soli nodi presenti sulla rete, e dato che A ed F conoscono il costo di attraversamento dei loro link (rispettivamente A-E ed F-E, ambedue pari a 1), ricaveranno facilmente che il costo di raggiungimento della destinazione E è pari a 1 per entrambi.

Principio del Distance Vector



² neighbour (UK) o neighbor (US): vicino (di casa)

L'algoritmo Distance Vector prevede ora che A ed F emettano a loro volta un pacchetto contenente un annuncio (**distance vector**) che riporti tutte le informazioni di loro conoscenza: prendendo il solo nodo A, l'annuncio (**distance vector**) dirà che A esiste e si raggiunge a costo zero, e che sulla rete esiste anche il nodo E, raggiungibile attraverso A a costo 1. Questo annuncio giungerà a C, il quale apprenderà dell'esistenza di A, F ed E, e così via.

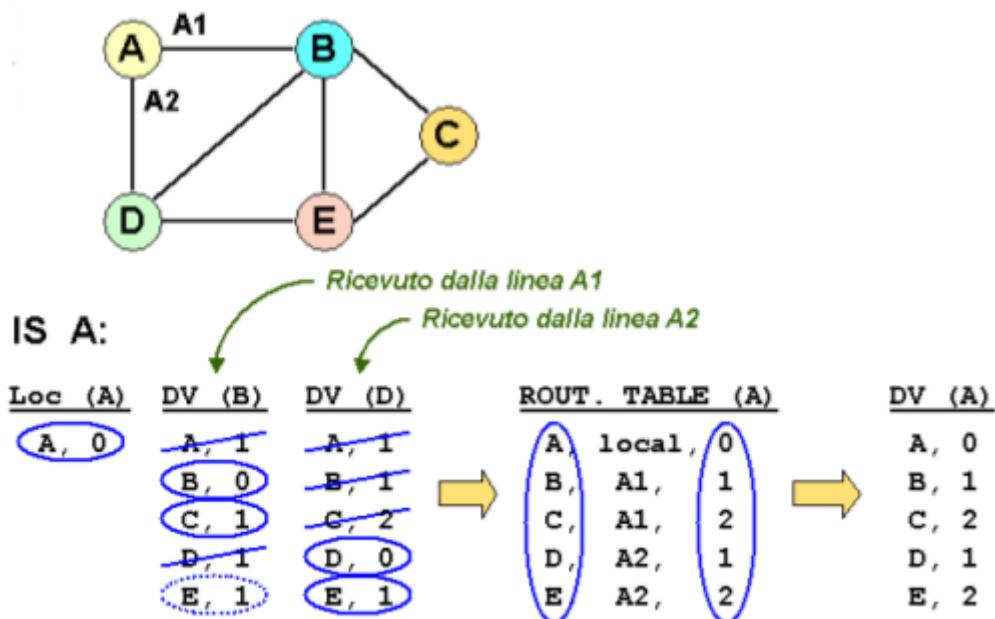
Fusione e generazione del Distance Vector

Lo scopo di un qualsiasi algoritmo di routing è la generazione della tabella di routing. Il Distance Vector ottiene questo risultato attraverso l'operazione di **fusione dei Distance Vector**, che consiste nel **selezionare, per ogni destinazione conosciuta, il percorso in base al costo migliore**.

L'operazione di fusione si basa sul fatto che ogni nodo memorizza l'ultimo DV³ giuntogli da ogni vicino adiacente, quindi ogni nodo conoscerà i costi del percorso *nodo_adiacente-destinazione* per tutte le destinazioni conosciute.

Si supponga, nella figura sottostante, di considerare il nodo A, che ha ricevuto il DV di D, e che quindi saprà che la distanza D-C è pari a 2. Da questa informazione lo IS A sarà in grado di capire che esisterà un percorso tra sé stesso e C: questo percorso passerà attraverso D ed avrà un costo pari a quello necessario a raggiungere D (ossia 1) più il costo tra D e C (ossia 2), ottenendo un costo per l'intero percorso A-C pari a 3.

Fusione e generazione di DV



Si supponga ora che A riceva anche il DV dell'altro nodo adiacente B, nel quale troverà l'informazione (C, 1), ossia che esiste un percorso tra B e C a costo 1. Il nodo A, applicando lo stesso procedimento di prima, sarà in grado di capire che esisterà un

³ Distance Vector

percorso alternativo A-C a costo 2, passando attraverso il nodo B. Il nodo A avrà due possibili percorsi per raggiungere C: il primo a costo 3 passando da D, il secondo a costo 2 passando da B. L'operazione di fusione per questa destinazione si completerà scegliendo il percorso migliore che sarà, ovviamente, quello a costo minore.

In alcuni casi, un nodo si troverà a dover **decidere tra due nodi a costo uguale**, come ad esempio il percorso A-E che può essere compiuto sia attraverso B, sia attraverso D. **In questi casi sarà facoltà del nodo decidere quale dei due sarà il prescelto.**

L'operazione di fusione dei DV verrà ripetuta per ogni destinazione conosciuta e comprenderà anche l'utilizzo di informazioni locali al nodo, che verranno utilizzate dal processo di fusione, così come i DV dei nodi adiacenti. Queste informazioni locali comprenderanno tutte le informazioni che sono conosciute direttamente dal nodo stesso: ad esempio il nodo A conoscerà sè stesso che verrà raggiunto ovviamente a costo 0. Questa informazione verrà utilizzata per determinare il percorso migliore con sé stesso per la costruzione della routing table.

Il risultato finale dell'operazione di fusione sarà la costruzione della routing table aggiornata per il nodo in esame.

Dalla routing table è immediato ricavare il DV che il nodo A emetterà verso i suoi nodi adiacenti, togliendo la colonna "next hop", che non è di alcuna utilità per la costruzione del DV.

L'algoritmo Distance Vector

L'algoritmo Distance Vector prevede che:

- un nodo emetta periodicamente il proprio DV verso i nodi adiacenti per aggiornare le informazioni di routing;
- il nodo sarà in ascolto dell'arrivo di altri DV dai nodi vicini;
- alla ricezione di un DV questo viene prima memorizzato, sostituendo un eventuale DV già in memoria ed emesso da quello stesso nodo;
- quindi viene ripetuta l'operazione di fusione e, eventualmente, aggiornata la routing table;
- se la routing table non varia rispetto a quella precedente il DV appena ricevuto non ha portato alcuna novità, quindi il nodo continuerà con il suo funzionamento normale;
- se invece la routing table subisce qualche variazione (ad esempio un nodo viene raggiunto attraverso un percorso diverso, oppure ad un costo diverso) viene ricalcolato il DV relativo al nodo in esame e propagato a tutti i nodi adiacenti, secondo la regola del "passaparola".

I DV vengono generati periodicamente, anche in presenza di una routing table che non ha subito alcuna variazione. La generazione periodica del DV è **necessaria per riconoscere l'esistenza o la mancanza di una adiacenza**. Ad esempio, la mancanza del collegamento tra il nodo A e il nodo B viene rilevata dal fatto che, per un certo periodo di tempo, A non riceve più il DV da B; allo scadere di un determinato timeout B verrà dichiarato irraggiungibile. **La generazione periodica fa sì che**

I'algoritmo DV sia indipendente dai meccanismi di segnalazione di eventuali guasti ai livelli sottostanti, infatti esistono situazioni nelle quali non è possibile rilevare la presenza o la mancanza del collegamento con un nodo, prima adiacente, per mezzo del solo segnale link-up proveniente dall'interfaccia fisica.

Il processo di ricalcolo della routing table viene attivato anche nel caso della caduta di un link direttamente connesso ad un IS, nel caso sia disponibile il segnale link-up. Ad esempio, nel caso in cui un IS rilevi che un link direttamente connesso ad esso risulti inutilizzabile, cancellerà dalla sua memoria tutti i DV provenienti da quella direzione in quanto non più attiva. Viceversa, se il segnale link-up non è disponibile, è necessario attivare l'apposito timeout associato ad ogni DV. Quando questo timeout scade, il DV viene invalidato e viene ricalcolata la routing table.

Il ricalcolo della routing table, anche in caso di guasto, avviene sempre con la stessa procedura: fusione dei DV presenti in memoria, a cui vengono aggiunte le informazioni locali. **Questo procedimento**, si vedrà in seguito, **può portare ad alcuni problemi durante i transitori⁴**.

Esempio: Cold Start

Per chiarire meglio i concetti esposti in precedenza si consideri una rete come quella nella figura sottostante e si simuli il comportamento dell'algoritmo DV, supponendo di inizializzare la rete alimentando tutti i nodi contemporaneamente (*cold start*). In questo stato, ciascuno dei nodi è caratterizzato unicamente da una conoscenza locale, che significa che ciascun nodo conosce il proprio indirizzo ma ignora totalmente la topologia della rete. La tabella di routing sarà quindi minima, caratterizzata da una singola voce, quella del nodo stesso.

⁴ Il **periodo di transitorio** è il periodo necessario alla rete per tornare alla stabilità dopo una variazione topologica o di costo

Esempio: Cold Start

RT (A)	RT (B)	RT (C)	RT (D)	RT (E)
A,loc,0	B,loc,0	C,loc,0	D,loc,0	E,loc,0

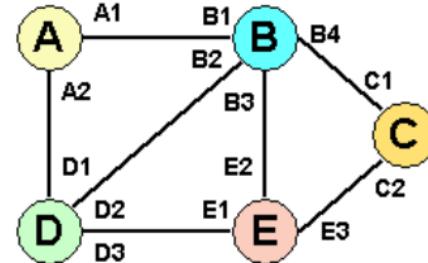
A emette il DV

RT (A)	RT (B)	RT (C)	RT (D)	RT (E)
A,loc,0	A,B1, 1 B,loc,0	C,loc,0	A,D1, 1 D,loc,0	E,loc,0

B e C emettono il DV

RT (A)	RT (B)	RT (C)	RT (D)	RT (E)
A,loc,0	A,B1, 1	A,C1 ,2	A,D1, 1	A,E2, 2
B,A1, 1	B,loc,0	B,C1 ,1	B,D2, 1	B,E2, 1
D,A2, 1	D,B2, 1	C,loc,0	D,loc,0	D,E1, 1
E,A2, 2	E,B3, 1	E,C2, 1	E,D3, 1	E,loc,0

Tutti emettono il DV



RT (A)	RT (B)	RT (C)	RT (D)	RT (E)
A,loc,0	A,B1, 1	A,C1 ,2	A,D1, 1	A,E2, 2
B,A1, 1	B,loc,0	B,C1 ,1	B,D2, 1	B,E2, 1
C,A1, 2	C,B4, 1	C,loc,0	C,D2, 2	C,E3, 1
D,A2, 1	D,B2, 1	D,C2, 2	D,loc,0	D,E1, 1
E,A2, 2	E,B3, 1	E,C2, 1	E,D3, 1	E,loc,0

Il nodo A genererà il DV che, nel caso specifico, sarà composto dall'unico valore attualmente presente nella tabella di instradamento, e lo invierà a tutti i nodi direttamente collegati (le adiacenze, ossia i nodi B e D), i quali vedranno ora crescere la loro conoscenza della rete.

B e D aggiorneranno tutte le distanze pervenute dal DV inviato da A sommando il costo del link locale, assunto pari a uno. A fronte della fusione del DV, sia B che D rileveranno una variazione nella loro tabella di routing, il che provocherà l'emissione di un loro nuovo DV verso i relativi nodi adiacenti.

Le tabelle di routing cominceranno quindi a popolarsi: i nodi A ed E conosceranno di avere due nuovi vicini (B, D), mentre i nodi B,C,D riconosceranno l'esistenza di un nuovo vicino (rispettivamente D per il nodo B, B per il nodo C e B per il nodo D). Tutti i nodi emetteranno di conseguenza il DV aggiornato, fino al raggiungimento dello stato stabile, quando i DV verranno generati solamente in base al timeout periodico.

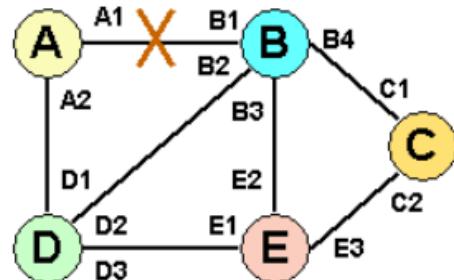
È facilmente verificabile come, in questo caso, i nodi saranno in grado di convergere in breve tempo ricavando ciascuno la propria routing table corretta, con il percorso migliore per tutte le destinazioni.

Esempio: caduta di un link

Si consideri ora il nodo A e si supponga che cada il collegamento con il nodo B, come mostrato nella figura sottostante.

Esempio: caduta di un link

- Procedura:
 - Invalidati i DV provenienti dal link A1
 - Mantenuto valido gli altri DV
 - Attivato il processo di fusione
- Efficienza
 - Il nodo A ricava la nuova RT senza scambi di DV con i nodi adiacenti



IS A:

<u>Loc (A)</u>	<u>DV (B)</u>	<u>DV (D)</u>	<u>ROUT. TABLE (A)</u>	<u>DV (A)</u>
A, 0	B, 1	A, 1		A, 0
B, 0	C, 1	B, 1		B, 2
C, 1	D, 1	C, 2		C, 3
D, 1	E, 1	D, 0	A, local, 0	D, 1
E, 1		E, 1	B, A2, 2	E, 2

Invalidation!

L'algoritmo Distance Vector prevede in questo caso che vengano invalidati tutti i DV provenienti dal link caduto (in questo caso solo da B), quindi verrà riattivato il processo di fusione.

È importante notare a questo proposito come risulti estremamente conveniente la memorizzazione dei DV dei nodi adiacenti: infatti A rileverà correttamente come il DV di B non sia più valido scartandolo, ma D non verrà interessato dal guasto e quindi si presupporrà che la routing table di D non varierà. La memorizzazione del DV di D permetterà quindi ad A di calcolare immediatamente la sua routing table senza ulteriori scambi di DV tra i nodi.

Problemi del Distance Vector e possibili soluzioni

Tutti gli algoritmi di routing distribuito hanno qualche problema durante transitori. Tuttavia, a causa di alcune caratteristiche intrinseche al Distance Vector, questo algoritmo può enfatizzarne alcuni aspetti.

Gli aspetti indesiderati più comuni del Distance Vector sono:

- **Black Hole:** i pacchetti dati inviati ad una particolare destinazione vengono ricevuti da un router sbagliato, il quale, non disponendo di una route per quella destinazione, non sa a chi inviare il pacchetto e pertanto decide di eliminarlo.
- **Bouncing Effect:** i pacchetti dati si trovano in un loop in quanto due o più router li rimbalzano tra di loro a causa della presenza di un **Routing Loop**.
- **Count to Infinity:** è un effetto perverso del **Distance Vector** che riguarda l'incremento progressivo del costo di raggiungimento di una destinazione

divenuta irraggiungibile. In questo caso i router non si accorgono che una destinazione è diventata irraggiungibile, e assumono erroneamente che sia invece ancora attiva, anche se riconoscono che la sua distanza sta aumentando sempre di più. Questo problema ha trovato una **parziale soluzione** utilizzando le tecniche dello **Split Horizon** e lo **Hold Down** descritti di seguito, anche se queste tecniche non sono mai totalmente affidabili.

I **primi due effetti** interessano i pacchetti dati che transitano in rete **durante il transitorio** e sono spesso presenti, in varie forme, in tutti gli algoritmi di routing. Il **terzo problema** è invece **peculiare** del **Distance Vector** ed è riferito al computo delle rotte, aumentando la durata del transitorio. Nel caso di **Distance Vector**, normalmente gli effetti di **Count to Infinity** e **Bouncing Effect** si manifestano **contemporaneamente**.

Il problema di fondo dell'algoritmo Distance Vector è la mancanza di conoscenza topologica. In altre parole, **nessuno dei nodi della rete ha idea di come la rete sia fatta**. L'algoritmo si basa sul meccanismo del passaparola ed ognuno dei nodi si fida ciecamente delle informazioni che gli vengono riportate dai nodi adiacenti.

Nel **Distance Vector**, che si basa su un calcolo incrementale dei costi verso ogni destinazione, **nessun nodo ha la possibilità di verificare come sia effettivamente fatta la rete** e quindi **se il calcolo dei costi sia stato fatto correttamente**.

I problemi precedenti sono chiaramente effetti non desiderati dell'algoritmo Distance Vector e si è pertanto cercato di minimizzarli mettendo a punto delle strategie che modificano il modo di operare dell'algoritmo stesso.

Le possibili soluzioni ai problemi del Distance Vector sono:

- **Split horizon:** serve a prevenire i loop tra due nodi adiacenti, facendo in modo che il router che riceve le informazioni relative a una certa destinazione da un router adiacente non può rispedire indietro informazioni su quella stessa destinazione.
- **Split horizon with poison reverse:** dal punto di vista teorico non aggiunge nulla di nuovo a quanto detto nel punto precedente, ma in questo caso, anziché omettere le destinazioni, queste vengono annunciate con distanza infinita, marcandole di fatto come irraggiungibili.
- **Route poisoning:** questo algoritmo è basato sull'osservazione che, in presenza di un Count to Infinity, il costo delle route verso una certa destinazione cresce progressivamente. L'idea è quindi quella di **bloccare l'utilizzo di tutte le route che aumentano improvvisamente di costo**: la route verrà rimessa in servizio solamente qualora due annunci successivi confermino l'esistenza di quella route, ambedue con lo stesso costo. Questo algoritmo ha un **tempo di convergenza** normalmente **rapido**, ma ha la **limitazione di scartare, a priori, anche gli annunci legittimi** che contengono un aumento di costo verso una determinata destinazione.
- **Hold down:** serve a limitare il **count to infinity** non permettendo di accettare alcun update relativamente ad un link che è stato rimosso dalla routing table, prima di aver aspettato un certo periodo di tempo (*hold down time*).

- **Triggered updates:** permette di inviare update non appena si verifica un cambiamento nella rete, senza aspettare il termine dell'intervallo di tempo previsto.

Tutti i possibili metodi applicabili non sono però definitivi a causa della mancanza della conoscenza topologica che caratterizza l'algoritmo Distance Vector, e quindi si troveranno sempre delle situazioni nelle quali gli algoritmi aggiuntivi entreranno in crisi.

In sostanza il vantaggio fondamentale del **Distance Vector** è la sua **facilità di implementazione**, ma per contro sono presenti diversi **aspetti critici** fra i quali:

- la **mancanza della conoscenza della topologia di rete** che rappresenta il limite più grosso;
- l'**elevata complessità** dell'algoritmo, che rende improponibile l'utilizzo di tale algoritmo per reti con più di un centinaio di nodi;
- la **lenta convergenza** ad un instradamento stabile;
- Presenza di numerosi casi nei quali **Count to Infinity**, **Bouncing Effect** e **Black Hole** non sono evitabili, neanche con meccanismi migliorativi all'algoritmo di base.
- Gli **algoritmi migliorativi tendono ad appesantire il protocollo** senza garantire mai un'efficacia totale.
- Necessità di definire una **soglia di "infinito" relativamente bassa per limitare il Count to Infinity**, che limita l'impiego pratico di questo algoritmo a reti piccole.

Dal punto di vista pratico i **protocolli basati sull'algoritmo Distance Vector** sono stati pesantemente **utilizzati in passato**. Attualmente **si tende a preferire algoritmi di tipo Link State**, anche se si trovano numerosissimi casi di **utilizzo di DV**, soprattutto **in reti periferiche di limitata estensione e con topologia semplice e con poche maglie**.

Una delle più vecchie implementazioni di questo algoritmo è il protocollo **RIP** (*Routing Information Protocol*) e il più recente **RIPv2**.

Un altro caso importante di routing di tipo Distance Vector è il **protocollo BGP** (*Border Gateway Protocol*), l'**attuale protocollo di routing interdominio** del mondo TCP/IP, il quale **usa una particolare variante chiamata Path Vector**.

Video riepilogativi

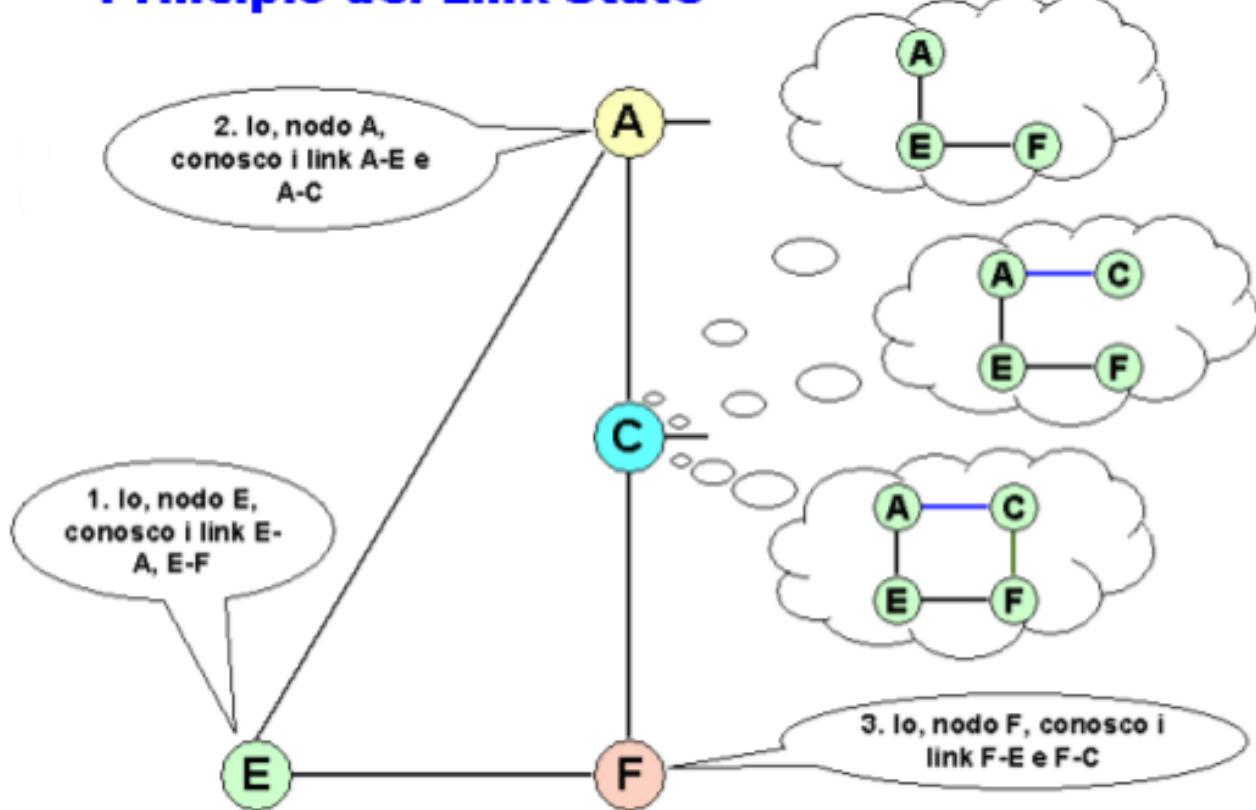
1. [Routing Dinamico - Distance Vector - Bellman Ford](#)
2. [Distance Vector Routing](#)

Link State

L'**algoritmo Link State si basa sul broadcast delle informazioni locali a tutti i nodi della rete**. Le informazioni locali, in questo caso, sono composte dallo stato di ogni collegamento tra un nodo e quelli adiacenti, informazione supposta semplice da ottenere. In altre parole, **ogni nodo comunica le informazioni locali in suo possesso a tutti gli altri nodi della rete**.

Si supponga ad esempio di lavorare nella rete mostrata in figura, e si supponga per semplicità che i nodi vengano accesi contemporaneamente (*Cold Start*).

Principio del Link State



Il nodo E comunicherà lo stato delle sue adiacenze, che avrà recuperato attraverso un altro meccanismo, a tutti i nodi della rete attraverso un meccanismo di broadcast. **Ogni nodo della rete sarà quindi in grado di ricostruire la topologia nell'intorno del nodo E**, ossia che E è adiacente ai nodi A ed F.

Se il procedimento viene ripetuto da tutti gli altri nodi (A annuncerà la sue adiacenze, quindi F, e così via), ogni nodo riceverà queste informazioni topologiche, e facendone l'unione sarà in grado di ricostruire l'intera topologia della rete.

Se inoltre **ogni nodo annuncerà non solo le adiacenze ma anche il costo per raggiungerle, ogni nodo sarà in grado di ricavare in autonomia il percorso migliore verso ogni destinazione**, ricavandolo dalla topologia della rete che si sarà costruito durante la fase di broadcast delle informazioni.

Caratteristiche dell'algoritmo Link State

L'algoritmo Link State, così come presentato in precedenza, è molto semplice. L'idea alla base è quella della **costruzione della mappa topologica della rete in ogni nodo**, aggiornata regolarmente, **al fine di determinare da essa i cammini minimi per raggiungere il nodo in esame partendo da un qualsiasi altro nodo della rete, determinando in questo modo la routing table**.

Pur sembrando semplice come algoritmo, per raggiungere il suo scopo il **Link State necessita della collaborazione di numerosi protocolli**, dimostrandosi di fatto più complicato del *Distance Vector*, ma superandone le principali limitazioni dovute soprattutto dalla mancanza di conoscenza topologica.

Il **Link State permette ad ogni router di avere una descrizione completa della topologia della rete** poiché scambia le informazioni sulle distanze direttamente con tutti i router della rete su cui viene applicato, e non solo con i vicini come invece fa il *Distance Vector*. Questo avviene tramite l'invio di pacchetti, detti **LSP** (*Link State Packet*), da parte di ogni router a tutti gli altri router della rete. La trasmissione avviene usando il protocollo di **flooding** che prevede l'inoltro di un pacchetto verso tutte le linee, tranne quella da cui è arrivato. Il pacchetto **LSP** è solitamente inviato solo quando avviene un cambiamento nella rete, come guasti o aggiunta di nuovi nodi, anche se alcuni gestori ne prevedono invece l'invio periodico.

Il pacchetto **LSP** contiene, per ogni mittente, l'elenco e la distanza da ogni vicino e, inoltre il è dotato di un numero di sequenza per permettere ai router di scartare eventuali **LSP** che dovessero giungere fuori sequenza. Nel caso il pacchetto ricevuto risulti in sequenza, viene memorizzato e ritrasmesso in **flooding**.

Tramite i pacchetti **LSP** ogni router costruisce un personale database con le informazioni sull'intera rete su cui è applicato l'algoritmo e, dopo aver ricevuto i pacchetti da tutti i router, è in grado di costruire un **grafo pesato** che rappresenta la rete stessa (**conoscenza topologica**). A questo punto viene applicato **localmente ad ogni router** un algoritmo per la ricerca dei cammini a costo minimo, come **l'algoritmo di Dijkstra** che è uno dei più noti.

L'algoritmo di Dijkstra è molto semplice ed è basato sull'idea che, dato un nodo radice, esisterà sicuramente un altro nodo la cui distanza dalla radice sarà minima. In altre parole, se A è un nodo radice, si selezioneranno tutte le adiacenze di questo nodo e si sceglierà l'adiacenza che viene raggiunta a costo minore. È evidente come non possano esistere percorsi alternativi migliori rispetto a quelli selezionati, essendo la destinazione a costo minore in assoluto. Il procedimento reitera considerando le adiacenze dell'insieme dei due nodi prescelti e selezionando la successiva adiacenza a costo minore. Il procedimento continuerà allo stesso modo, memorizzando e inserendolo ogni volta opportunamente nella routing table il percorso trovato per raggiungere un nodo.

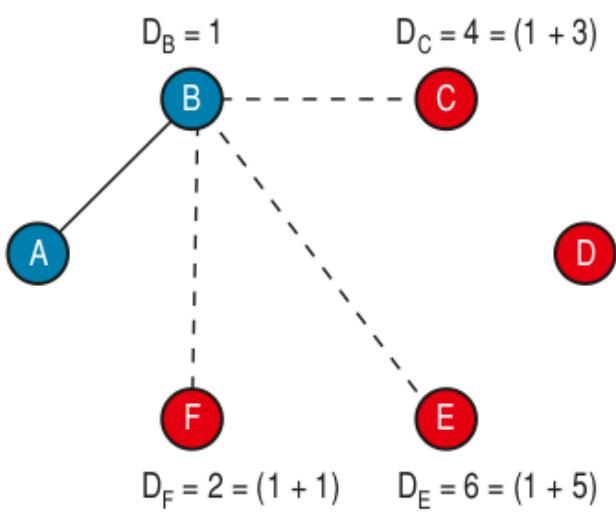
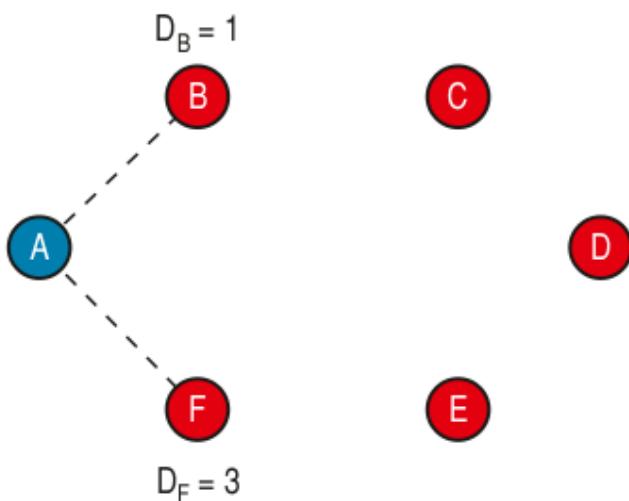
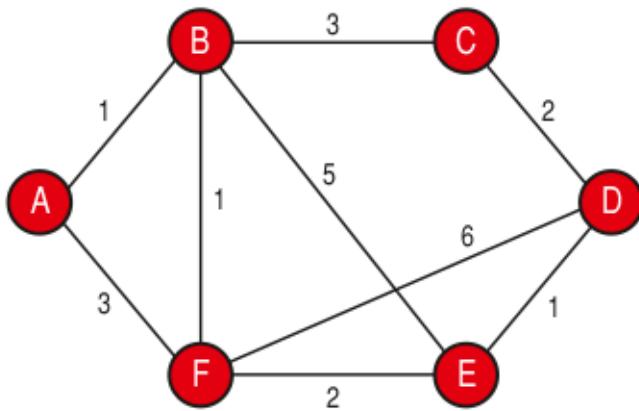
Vediamo un esempio di applicazione dell'algoritmo di Dijkstra.

Si supponga di avere la rete mostrata in figura, rappresentata mediante un grafo non orientato con i costi di ogni link indicati sugli archi.

Lo scopo dell'algoritmo è quello di trovare i **cammini minimi** per raggiungere il nodo in esame, ad esempio A, partendo da un qualsiasi altro nodo della rete.

Per riuscire a determinare i cammini minimi si procederà con la scelta del nodo adiacente ad A che avrà il costo minore per raggiungerlo. Il nodo A verrà considerato come nodo **permanente**, mentre i due nodi B e F risulteranno essere **temporanei**. Uno dei due nodi fra B ed F cambierà il suo stato in **permanente** nel momento in cui verrà selezionato come nodo sul percorso a costo minore per raggiungere A.

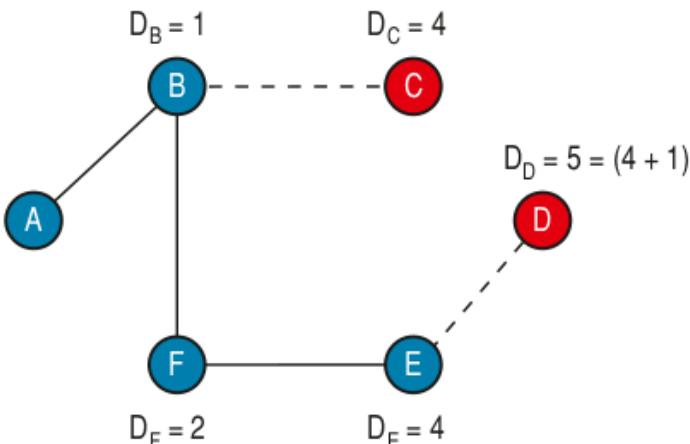
Fra B e F, il nodo che si trova sul percorso a costo minore per raggiungere A è il nodo B che quindi diventerà un nodo permanente.



Avendo scelto B come successivo nodo permanente vorrà dire che qualsiasi altro cammino a costo minimo per raggiungere A da qualsiasi altro nodo, dovrà necessariamente passare attraverso il link B-A altrimenti non sarebbe un cammino minimo.

Per trovare tutti i successivi cammini minimi si dovrà quindi procedere con il cercare quale, fra i nodi vicini a B, risulterà essere quello che lo raggiunge a costo minore, e che quindi raggiungerà anche A a costo minimo.

Fra tutti i vicini di B il nodo scelto sarà F che raggiungerà B a costo 1 e quindi A attraverso B a costo 2. Il nodo F viene quindi impostato come nodo permanente mentre tutti gli altri nodi vicini a B rimarranno etichettati come nodi temporanei.

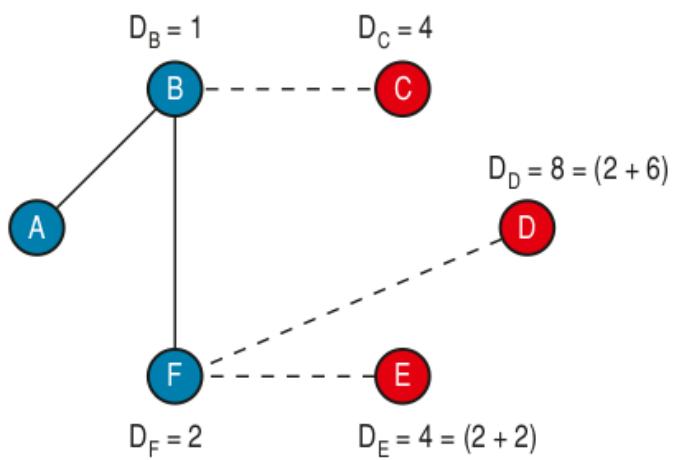


Nel momento in cui più nodi avessero dei vicini da selezionare si partira' sempre da quello che avrà il costo minore per raggiungere A. Di conseguenza l'algoritmo continuerà a scegliere i cammini minimi, prima selezionando C per raggiungere B, e poi D per raggiungere E, ottenendo alla fine il grafo mostrato in figura.

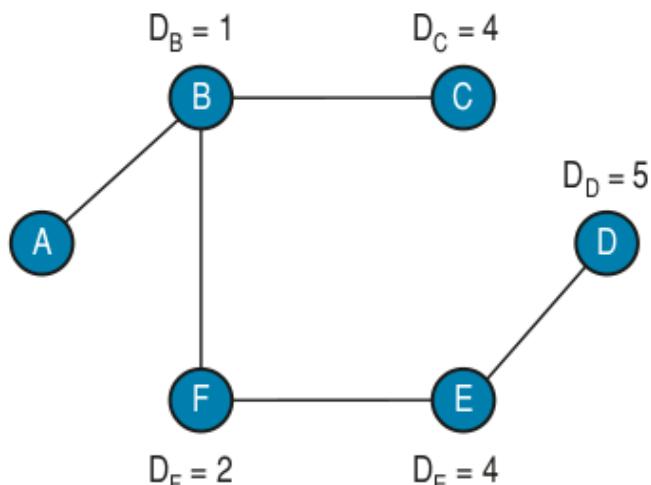
Le caratteristiche positive del **Link State** possono quindi riassumersi nei seguenti punti:

- ogni router è a conoscenza della topologia della rete;
- l'algoritmo ha una convergenza rapida poiché le informazioni sulle distanze vengono propagate con il protocollo di (**selective**) **flooding**, senza alcuna elaborazione intermedia, e **la ricerca dei cammini minimi avviene localmente** in ogni router, senza ulteriori scambi con altri router;
- l'algoritmo **non ha problemi nel transitorio** e generalmente è in grado di recuperare eventuali errori o problemi;
- l'algoritmo **dificilmente genera loop**, e comunque è in grado di identificarli ed interromperli;
- tutti i **nodi** giungeranno ad avere un **database identico**;
- è un algoritmo **facilmente scalabile**.

Il principale svantaggio del **Link State** è la complessità di realizzazione essendo l'unione di più protocolli. Inoltre richiede un notevole dispendio di memoria per la gestione del database della rete.



L'algoritmo verrà quindi riapplicato ad F, individuando fra tutti i suoi nodi vicini quale lo raggiungerà a costo minore, giungendo a scegliere il nodo E.



In generale l'algoritmo **Link State** è generalmente considerato **migliore rispetto al Distance Vector**, nonostante sia più complicato in quanto richiede molti più componenti per la sua operabilità.

Uno dei protocolli più diffusi che utilizzano l'algoritmo **Link State** è il protocollo **OSPF** (*Open Short Path First*).

Video riepilogativi

1. [Routing Dinamico - Link State - LSP](#)
2. [Link State Routing](#)
3. [Routing Dinamico - Algoritmo di Dijkstra](#)
4. [Dijkstra's Algorithm](#)
5. [Dijkstra's Shortest Path Algorithm](#)
6. [Routing Protocols Overview \(Distance Vector and Link-State\) CCNA Part1](#)

Libro vol.2 - Routing gerarchico

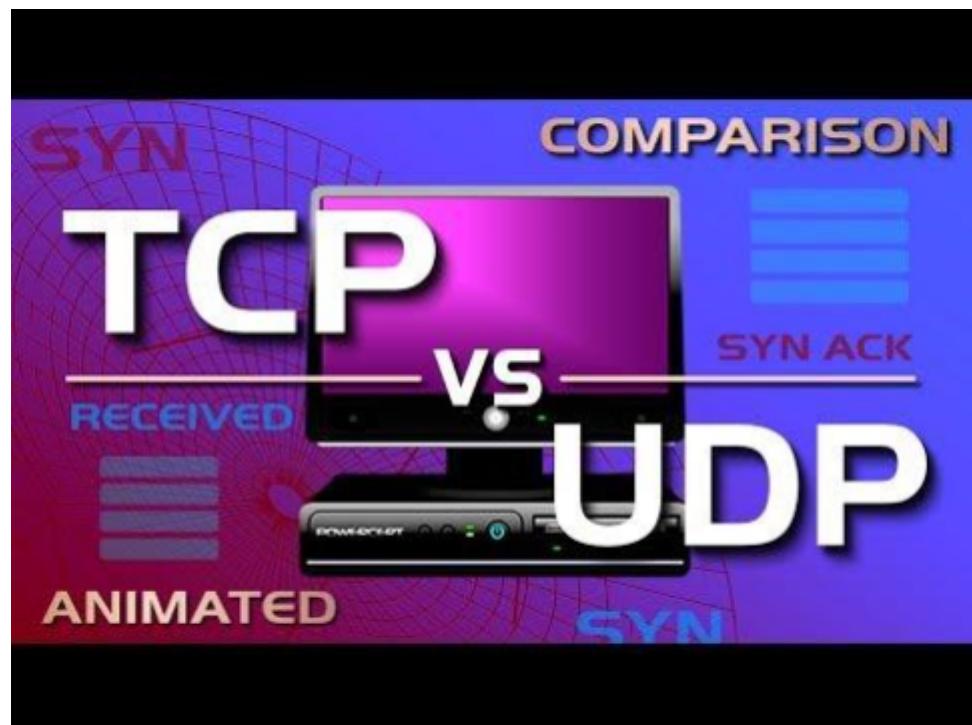
STUDIARE: *L. Lo Russo, E. Bianchi, Sistemi e Reti - Nuova Edizione OPENSCHOOL - Per l'articolazione INFORMATICA degli Istituti Tecnici settore Tecnologico, vol. 2, ed. Hoepli, 2016*

- Introduzione pp.195-197
- Tassonomia dell'internetworking pp.198-199
- Exterior Gateway Protocol (EGP) pp.210-211
 - Border Gateway Protocol (BGP) pp.211

Video riepilogativi

1. [Hierarchical Routing](#)
2. [Prefix Aggregation and Subnets](#)
3. [Routing Protocols Overview \(Cisco CCNA- RIP, RIPv2, EIGRP, OSPF\) Part2](#)
4. [Border Gateway Protocol](#)

Il livello di trasporto



Watch the video, it will help!

Libro vol.2 - Lo strato di trasporto

STUDIARE: *L. Lo Russo, E. Bianchi, Sistemi e Reti - Nuova Edizione OPENSCHOOL - Per l'articolazione INFORMATICA degli Istituti Tecnici settore Tecnologico, vol. 2, ed. Hoepli, 2016*

- Lo strato di trasporto e il protocollo UDP pp.240-259
- Il trasferimento affidabile e il protocollo TCP pp.260-276
- TCP: problematiche di connessione e congestione pp.277-286 (*leggere per cultura personale*)

Assegnazione e traduzione degli indirizzi

Prefazione

I seguenti appunti sono basati sul lavoro svolto dalla Prof.ssa Sophia Danesino, mentore, amica, appassionata di conoscenza e insegnante fuori dal comune. Sul suo [canale YouTube](#) è possibile trovare i video su molti degli argomenti trattati.

Libro vol.2 - Assegnazione degli indirizzi tramite DHCP

STUDIARE: *L. Lo Russo, E. Bianchi, Sistemi e Reti - Nuova Edizione OPENSCHOOL - Per l'articolazione INFORMATICA degli Istituti Tecnici settore Tecnologico, vol. 2, ed. Hoepli, 2016*

- Assegnazione mediante DHCP pp.44-45
- Packet Tracer: assegnazione indirizzi dinamici pp.124-132 **controllare l'es.**

STUDIARE: si vedano i seguenti video al fine di comprendere gli argomenti trattati e si studino i testi linkati

- Router DHCP
 - [DHCP: configurazione router](#) di [Sophia Danesino](#) (*video*)
 - [DHCP: fasi di assegnazione indirizzi](#) di [Sophia Danesino](#) (*video*)
- Server DHCP
 - [Cisco Packet Tracer - Server DHCP \[ITA\]](#) di [Giacomo De Liberali](#) (*video*)

NAT (Network Address Translation)

NAT e PAT (vol.2)

STUDIARE: L. Lo Russo, E. Bianchi, Sistemi e Reti - Nuova Edizione OPENSCHOOL - Per l'articolazione INFORMATICA degli Istituti Tecnici settore Tecnologico, vol. 2, ed. Hoepli, 2016

- Inoltro di pacchetti sulla rete: NAT, PAT e ICMP pp.51-55

NAT

NAT (*Network Address Translation*) è una tecnica usata per sostituire nell'intestazione di un pacchetto IP un indirizzo, sorgente o destinazione, con un altro indirizzo.

Nel suo impiego più diffuso **NAT viene usato per permettere ad una rete che usa una classe di indirizzi privata di accedere ad Internet usando uno o più indirizzi pubblici.**

NAT è stato ideato nel momento in cui ci si è accorti che lo spazio di indirizzamento IPv4 non era più sufficiente per soddisfare la crescente richiesta di indirizzi, cosa che non era stata immaginata o pianificata al momento della sua creazione. Inizialmente **NAT** si pensò fosse una soluzione temporanea, presto soppiantata da IPv6, invece il ritardo nello sviluppo di IPv6 e la rapida crescita dell'accesso ad Internet hanno di fatto reso il **NAT** una pratica molto comune.

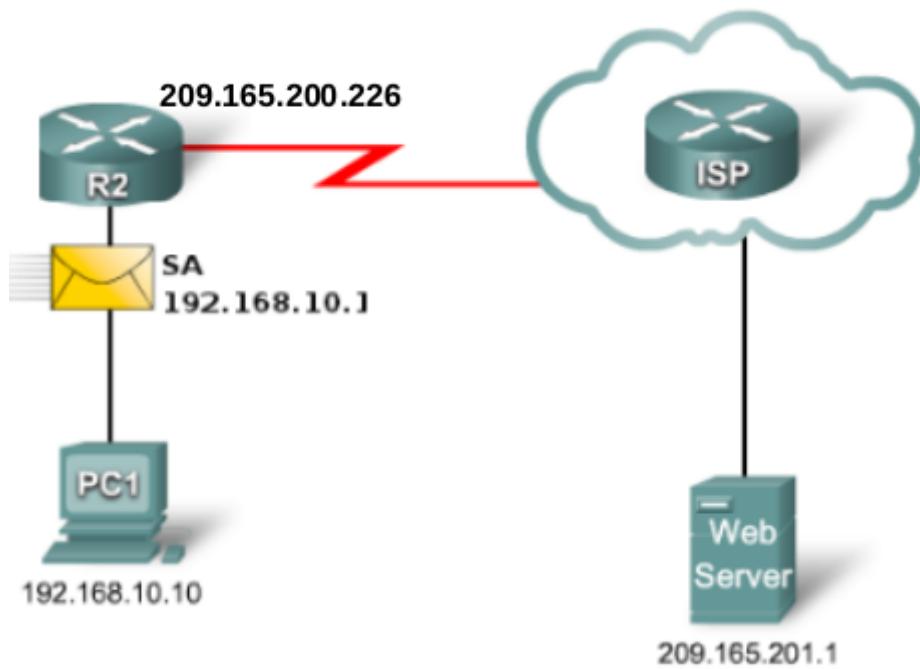
Il **NAT** fornisce i seguenti vantaggi:

- consente ad un'organizzazione di utilizzare un numero di indirizzi IP pubblici inferiore al numero degli host;
- riduce i costi di accesso ad Internet;
- **garantisce più sicurezza per i computer della rete locale che rimangono "nascosti" dietro i pochi indirizzi pubblici utilizzati.**

NAT prevede che internamente allo header del pacchetto IP avvenga una sostituzione dell'indirizzo IPv4 nel momento in cui avviene il passaggio del pacchetto dalla rete interna a quella esterna, e viceversa, tenendo traccia della sostituzione effettuata in una tabella di conversione affinché venga individuato in modo corretto lo host coinvolto nella comunicazione.

Vediamo un semplice esempio di traduzione.

Supponiamo che un host (192.168.10.10) sulla LAN voglia comunicare con un Web server (209.165.201.1). Lo host invierà il pacchetto all'interfaccia della LAN del router R2 (192.168.10.1), cioè il gateway della LAN che è stato configurato per fare il NAT.



Il router R2 legge l'IP sorgente del pacchetto e controlla se è uno di quelli previsti per la traduzione **NAT**, usando una **ACL** (Access Control List) che identifica gli host validi per la traduzione. Se l'host figura nella **ACL** il router R2 trasforma l'indirizzo locale dell'host 192.168.10.10 in quello pubblico 209.165.200.226 e memorizza tale corrispondenza in una tabella detta **NAT table**.

Il router R2 invia il pacchetto a destinazione e, quando il server web risponderà, invierà il pacchetto all'indirizzo pubblico di R2 (209.165.200.226).

Il router R2 esamina la sua **NAT table** e trasforma l'indirizzo 209.165.200.226 nel corrispondente indirizzo locale, 192.168.10.10, inviandolo allo host con questo indirizzo IPv4, cioè PC1.

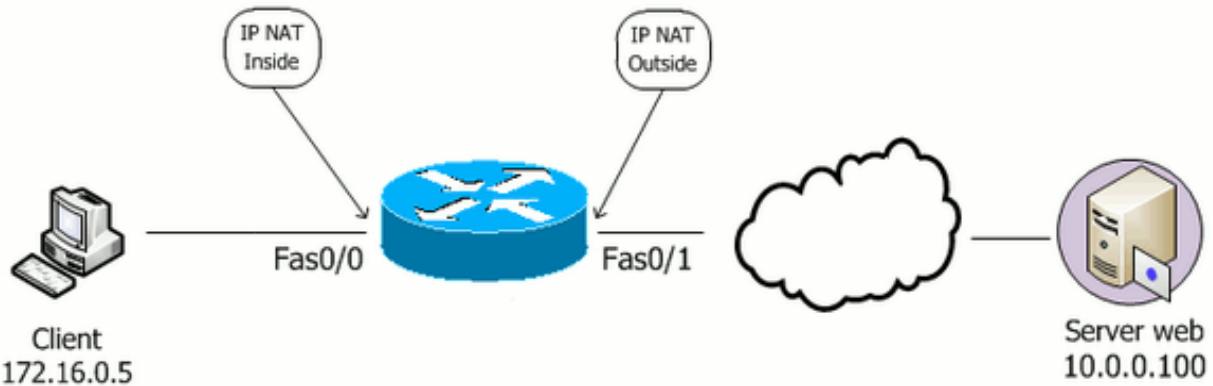
Se invece non venisse trovata alcuna corrispondenza nella **NAT table** il pacchetto verrebbe scartato.

La NAT table

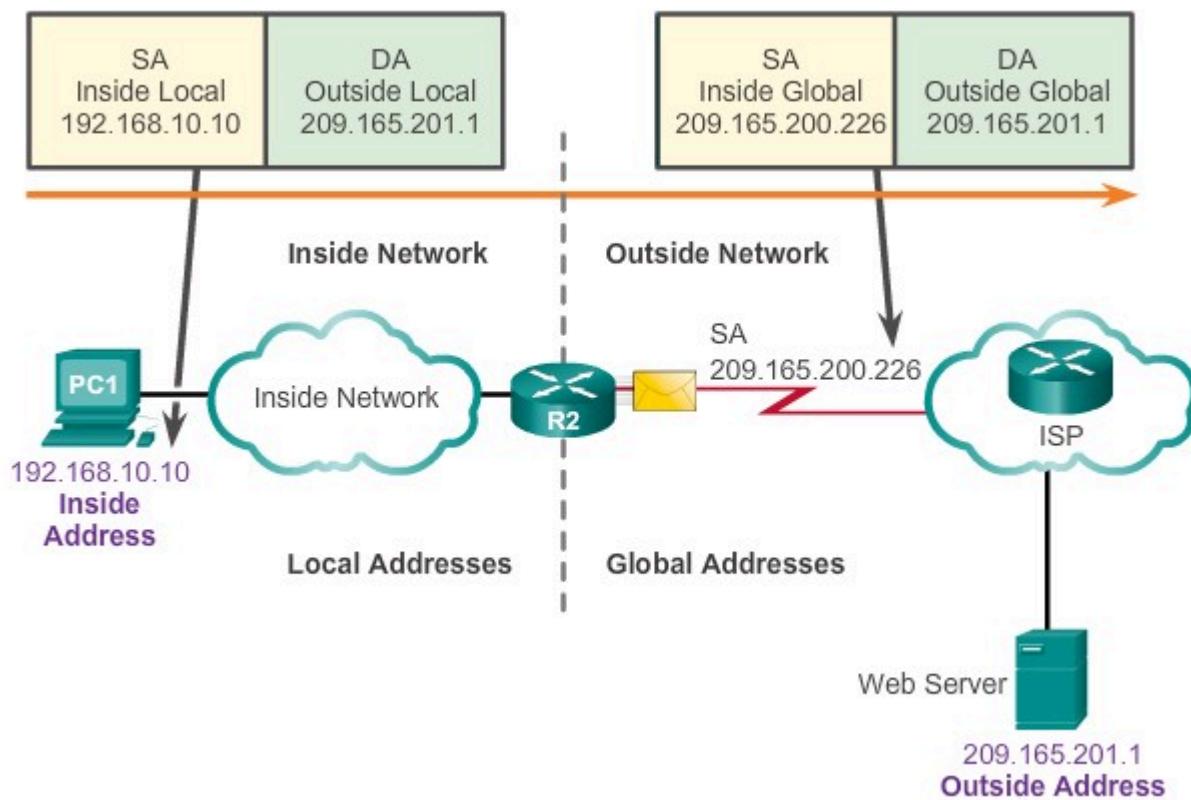
Il **NAT** suddivide la rete in due aree identificando gli indirizzi interni o **inside**, solitamente la LAN, e gli indirizzi esterni o **outside**, generalmente Internet.

Porte inside e outside

La traduzione di indirizzi viene configurata su un router e, dato che un indirizzo interno alla rete locale viene tradotto in un indirizzo visibile all'esterno della rete, è importante individuare le due interfacce del router coinvolte nel trasferimento dei pacchetti i cui indirizzi vengono usati per la traduzione. Gli indirizzi interni alla rete locale sono detti **inside**, mentre quelli esterni sono detti **outside**.

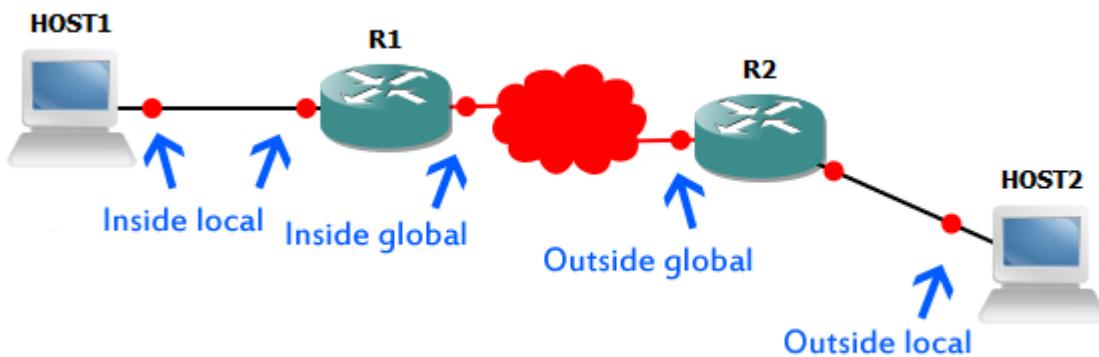


Nell'esempio seguente sono rappresentate la rete interna/**inside** e quella esterna/**outside** ai due lati del router:



- all'interno della rete (**local**) l'indirizzo sorgente (**SA - Source Address**) è quello privato 192.168.10.10 (**inside local**), mentre l'indirizzo destinatario (**DA - Destination Address**) è quello pubblico 209.165.201.1 (**outside local**);
- dopo la traduzione, nella rete esterna (**outside**), l'indirizzo sorgente (**SA**) è quello della porta esterna del router R2, 209.165.200.226 (**inside global**), mentre l'indirizzo destinatario (**DA**) è quello pubblico 209.165.201.1 (**outside global**).

In uno schema generale possiamo identificare le interfacce come in figura:



Il router gestisce a livello RAM una tabella che tiene conto delle traduzioni attive, detta **NAT table**.

NAT: sicurezza e amministrazione

L'implementazione del NAT sul confine tra LAN e WAN crea automaticamente una forma di protezione tra la rete privata e quella pubblica. Infatti **il NAT consente solo le comunicazioni che hanno origine dalla rete interna, e un computer esterno non può chiedere una connessione a un computer interno alla LAN poiché non troverebbe la corrispondenza nella tabella NAT**. La corrispondenza nella tabella dall'esterno avviene con una riga scritta quando parte la richiesta dall'interno.

Un altro vantaggio dell'utilizzo del NAT riguarda il fatto che è possibile **cambiare gli indirizzi IP privati dei nodi**, poiché essi vengono schermati all'esterno, ed è inoltre possibile **scalare** (cioè aumentare le dimensioni della rete, il numero di host connessi) senza dover richiedere ulteriori indirizzi pubblici.

Tecniche di traduzione degli indirizzi

NAT statico

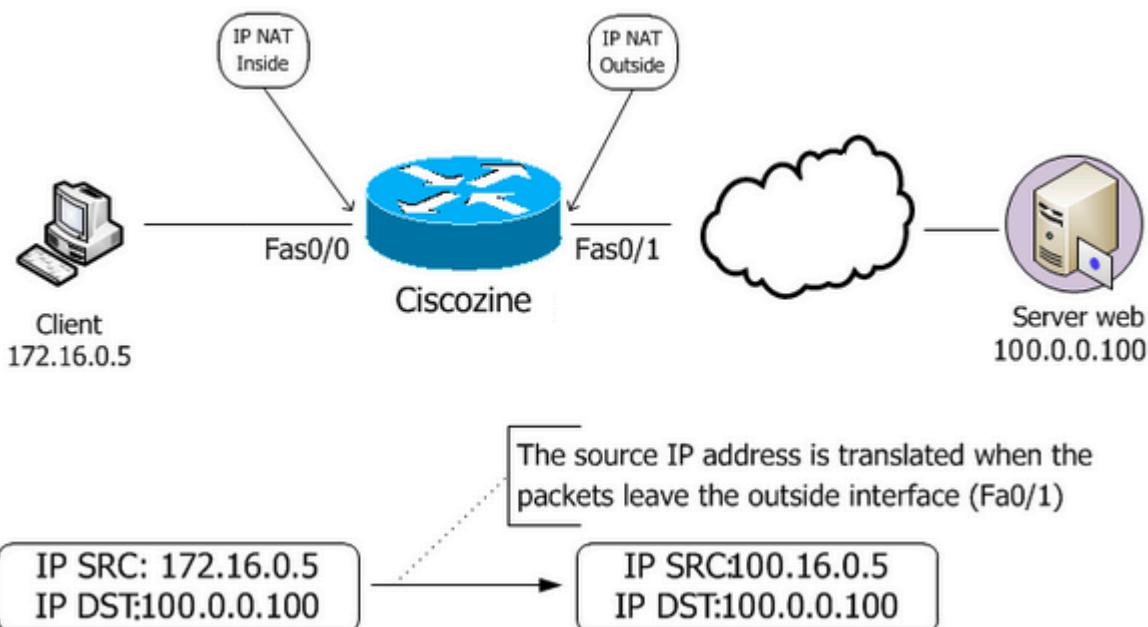
[NAT statico](#) della [Prof.ssa Sophia Danesino](#).

Il **NAT statico** permette di effettuare una **traduzione one-to-one tra indirizzi locali e globali**. E' utile per i **server web** o per altre tipologie di **server interni alla LAN che necessitano di essere raggiunti dall'esterno**, e comunque, in linea generale, per tutti gli host che devono avere un indirizzo costante accessibile da Internet.

Ai server che devono essere accessibili dall'esterno viene assegnato un indirizzo IPv4 coerente con gli indirizzi definiti nella rete interna. Nel router viene successivamente creata una mappa statica con la corrispondenza tra l'indirizzo IPv4 pubblico esterno e quello interno all'azienda.

Questa tecnica non preserva lo spazio di indirizzi IPv4 dato che a ciascun indirizzo privato che deve essere mappato deve corrispondere in modo univoco un indirizzo esterno pubblico, ma **permette** comunque **di esporre su Internet un server che si trova in una rete privata**. Le traduzioni sono presenti nella **NAT table** al momento della configurazione e vi rimangono fino a che non vengono esplicitamente cancellate.

Vediamo un esempio in cui il NAT statico venga fatto su un client, la versione su di un server è del tutto analoga:



```
Ciscozine(config)# interface fa0/0
Ciscozine(config-if)# ip nat inside
Ciscozine(config-if)# exit
Ciscozine(config)# interface fa0/1
Ciscozine(config-if)# ip nat outside
Ciscozine(config-if)# exit
Ciscozine(config)# ip nat inside source static 172.16.0.5 100.16.0.5
```

Se il client invia una richiesta HTTP ad un server web, la NAT table sarà la seguente:

```
Ciscozine# show ip nat translations
Prot. Inside global    Inside local      Outside local      Outside global
tcp    100.16.0.5:5608  172.16.0.5:5608  100.0.0.100:80   100.0.0.100:80
```

NAT dinamico

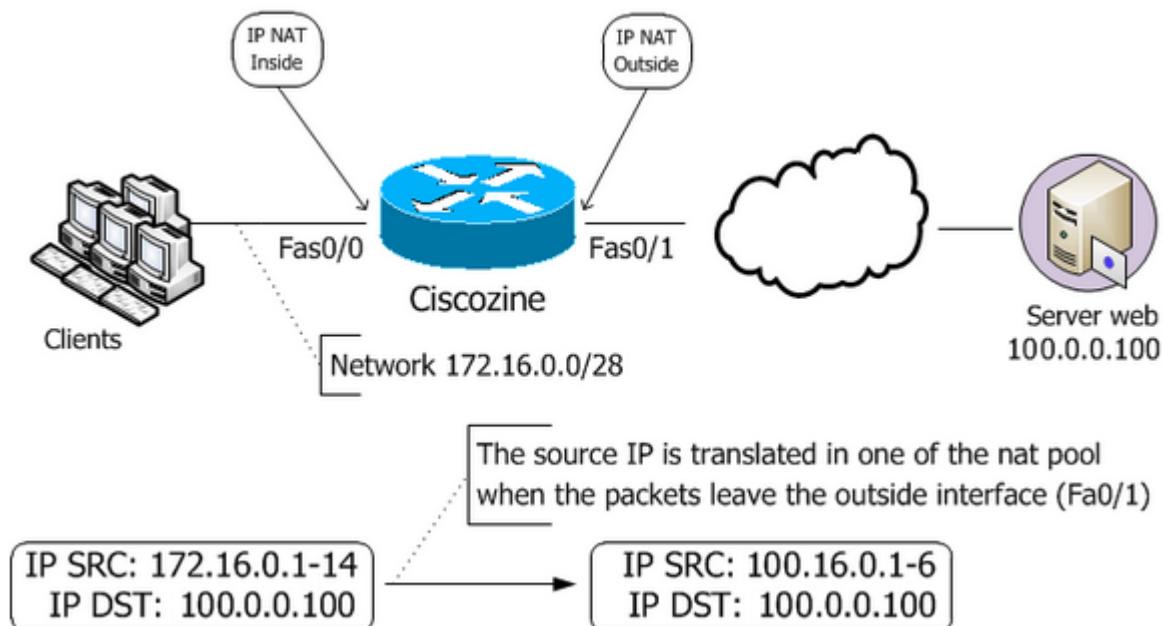
[NAT dinamico](#) della Prof.ssa Sophia Danesino.

Il **NAT dinamico** usa un **pool di indirizzi pubblici e li assegna ai vari dispositivi basandosi su una tecnica FIFO**, cioè viene assegnato il primo indirizzo IPv4 pubblico libero.

Questa tecnica elimina la necessità di avere lo stesso numero di indirizzi interni ed esterni, ma se il **NAT dinamico** terminasse il pool di indirizzi a disposizione, scarterebbe il pacchetto. Risulta quindi necessario configurare nella **NAT table** un pool di indirizzi abbastanza grande da soddisfare simultaneamente tutte le traduzioni possibili.

Gli indirizzi vengono tradotti non appena lasciano la rete interna e le traduzioni rimangono attive solo per la durata della comunicazione. Una volta ricevuta la risposta, la voce di traduzione nella **NAT table** viene eliminata, e il corrispondente indirizzo globale può essere riutilizzato dal pool NAT ed utilizzato da un altro indirizzo privato.

Vediamo un esempio.



```
Ciscozine(config)# interface fa0/0
Ciscozine(config-if)# ip nat inside
Ciscozine(config-if)# exit
Ciscozine(config)# interface fa0/1
Ciscozine(config-if)# ip nat outside
Ciscozine(config-if)# exit
Ciscozine(config)# ip nat pool fuori 100.0.16.1 100.0.16.6 netmask
255.255.255.248
Ciscozine(config)# ip access-list 10 permit 172.16.0.0 0.0.0.15
```

```
Ciscozine(config)# ip nat inside source list 10 pool fuori
```

Dopo aver definito le interfacce inside e outside sul router:

- viene definito il pool di indirizzi esterni assegnabili (fuori);
- vengono definiti gli indirizzi interni a cui il pool può essere assegnato (10);
- viene fatta l'associazione tra indirizzi interni ed esterni (10 e fuori).

Se due client (172.16.0.1 e 172.16.0.2) inviano una richiesta HTTP al server web 100.0.0.100 la NAT table sarà la seguente:

```
Ciscozine# show ip nat translations
```

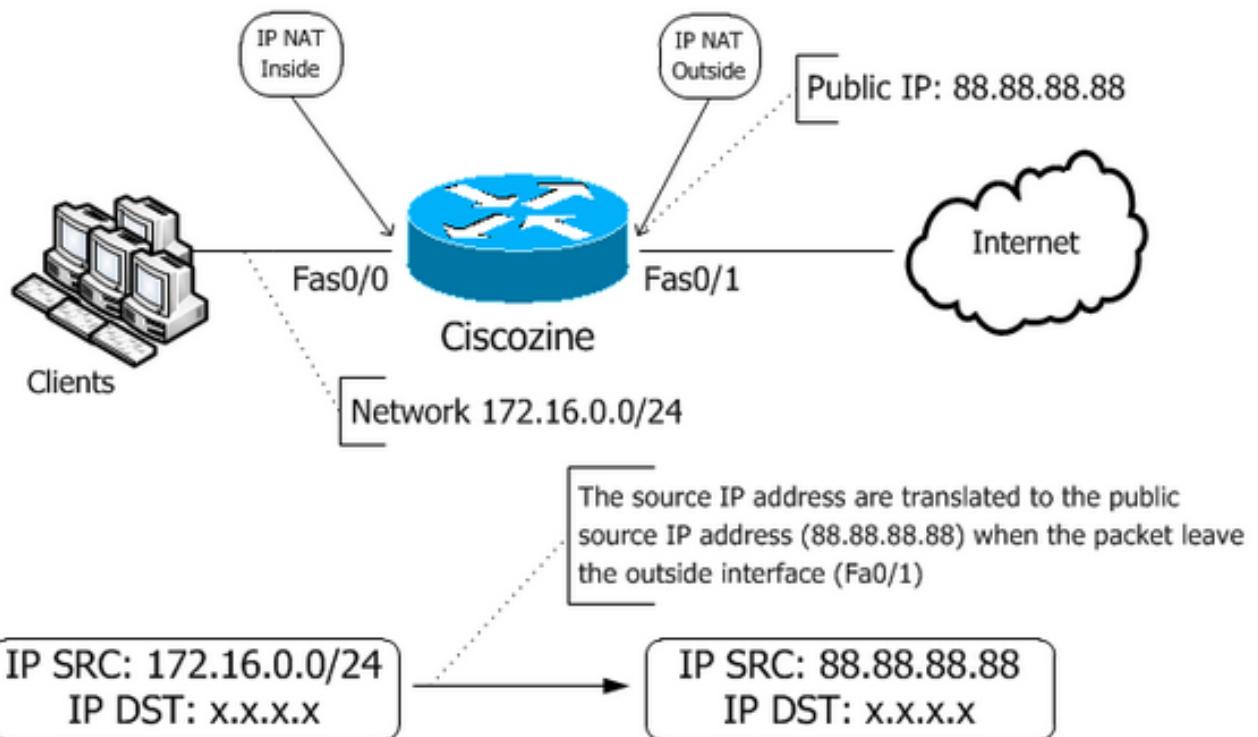
Prot.	Inside global	Inside local	Outside local	Outside global
tcp	100.0.16.2:3569	172.16.0.2:3569	100.0.0.100:80	100.0.0.100:80
tcp	100.0.16.1:5618	172.16.0.1:5618	100.0.0.100:80	100.0.0.100:80

NAT overload o PAT

[PAT - NAT overload](#) della [Prof.ssa Sophia Danesino](#). Nel video c'è un errore nella configurazione della access-list, la Prof.ssa Danesino ha usato la netmask invece della wildcard. Il video è comunque valido per la spiegazione del funzionamento generale del PAT, ma per la configurazione dei dispositivi si faccia riferimento a quanto indicato di seguito.

Il **NAT overloading** (anche detto *Port Address Translation* o **PAT**) **mappa più indirizzi IPv4 privati su un singolo IPv4 pubblico**. In questo modo più indirizzi privati possono contemporaneamente accedere ad un indirizzo pubblico.

Vediamo un esempio.



```
Ciscozine(config)# interface fa0/0
Ciscozine(config-if)# ip nat inside
Ciscozine(config-if)# exit
Ciscozine(config)# interface fa0/1
Ciscozine(config-if)# ip nat outside
Ciscozine(config-if)# exit
Ciscozine(config)# access-list 10 permit 172.16.0.0 0.0.0.255
Ciscozine(config)# ip nat inside source list 10 interface fa0/1 overload
```

Dopo aver definito le interfacce inside e outside sul router:

- viene definito l'insieme di indirizzi interni a cui l'indirizzo pubblico può essere assegnato (10);
- viene fatta l'associazione tra indirizzi interni ed indirizzo pubblico configurato sull'interfaccia fa0/1. Si noti la parola chiave **overload** al termine del comando che abilita di fatto il **PAT**.

Il **PAT** tiene traccia anche del numero di porta del sorgente, e **assicura che i client usino differenti numeri di porta per ogni sessione client con un server su Internet**. Quando il **server risponde il numero di porta determina a quale client il router debba destinare il pacchetto**.

```
Ciscozine# show ip nat translations
Prot. Inside global      Inside local      Outside local      Outside global
tcp    88.88.88.88:7921  172.16.0.2:7921  95.100.96.233:443  95.100.96.233:443
tcp    88.88.88.88:8651  172.16.0.5:8651   173.194.44.18:80   173.194.44.18:80
tcp    88.88.88.88:8652  172.16.0.11:8651  173.194.44.18:80   173.194.44.18:80
```

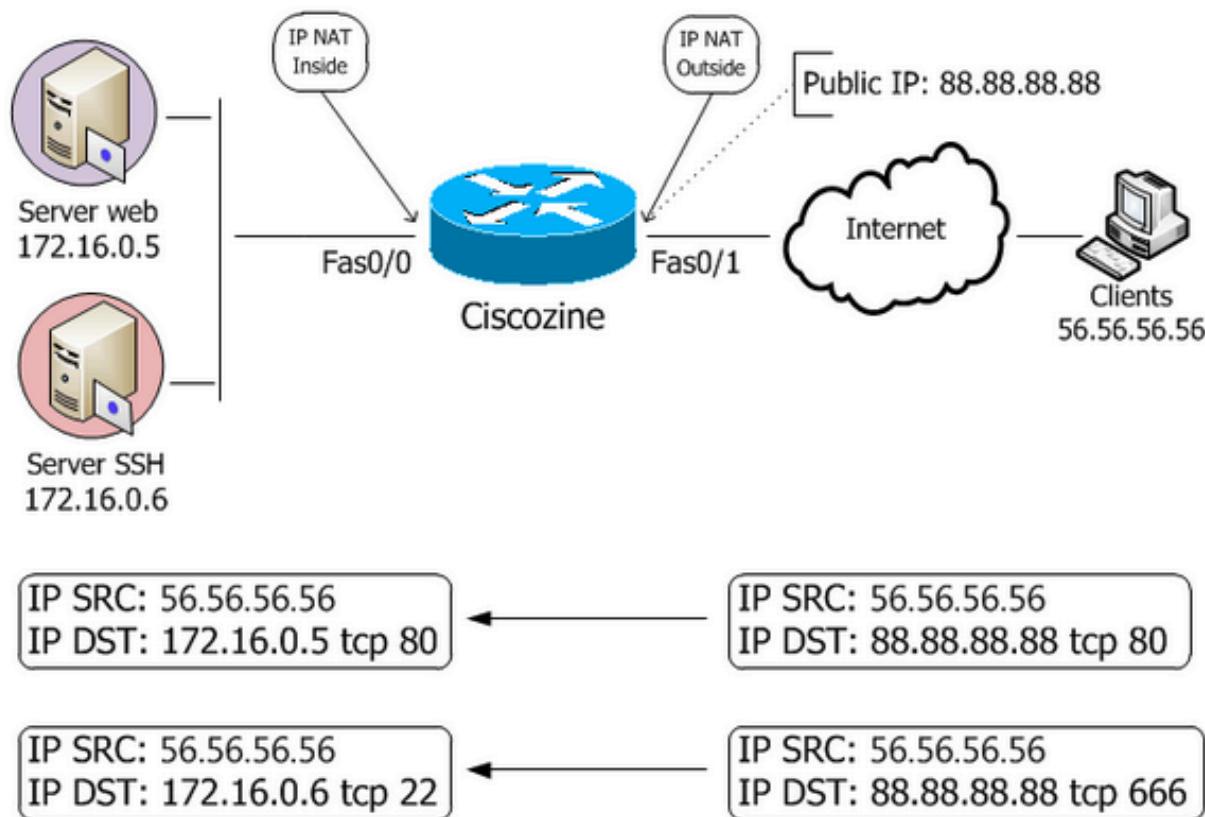
Se la porta sorgente non è disponibile (nell'esempio la 8651 è già occupata), la tecnica **PAT** inizia la ricerca di un numero di porta usabile in uno degli intervalli di porte disponibili per ciascun indirizzo globale, e assegnerà il primo disponibile. Se non dovesse trovare un numero di porta disponibile, il pacchetto verrebbe scartato.

PAT statico

Il **PAT statico** consente ad **una porta specifica UDP o TCP su un indirizzo globale di essere tradotta in una porta specifica locale**.

Nell'esempio seguente una richiesta proveniente dall'esterno all'indirizzo pubblico 88.88.88.88 sulla porta 80 deve essere diretta al server web interno 172.16.0.5 sulla porta 80.

Analogamente una richiesta proveniente dall'esterno all'indirizzo pubblico 88.88.88.88 sulla porta 666 deve essere diretta al server SSH interno 172.16.0.6 sulla porta 22.



Questo meccanismo protegge i server interni consentendone l'accesso solo per una specifica porta e tramite l'IP pubblico, nel nostro esempio 88.88.88.88, esponendo all'esterno tutti i server con lo stesso indirizzo IP.

La configurazione del router è più semplice delle precedenti dovendo solo definire la traduzione tra ip/porta esterno e ip/porta interno.

```
Ciscozine(config)# interface fa0/0
Ciscozine(config-if)# ip nat inside
Ciscozine(config-if)# exit
Ciscozine(config)# interface fa0/1
Ciscozine(config-if)# ip nat outside
Ciscozine(config-if)# exit
```

```
Ciscozine(config)# ip nat inside source static tcp 172.17.0.5 80
88.88.88.88 80
Ciscozine(config)# ip nat inside source static tcp 172.17.0.6 22
88.88.88.88 666
Ciscozine(config)# access-list 1 permit 172.16.0.0 0.0.255.255
Ciscozine(config)# ip nat inside source list 1 interface fa0/0 overload
```

Dopo aver definito le interfacce inside e outside sul router:

- vengono associati ip/porta interni con ip/porta esterni specificando il protocollo a livello di trasporto usato;
- viene definito l'insieme di indirizzi interni a cui l'indirizzo pubblico può essere assegnato (1);
- viene fatta l'associazione tra indirizzi interni ed indirizzo pubblico configurato sull'interfaccia fa0/1. Si noti la parola chiave **overload** al termine del comando che abilita di fatto il **PAT**.

La NAT table viene caricata al momento della configurazione del router:

```
Ciscozine(config)# show ip nat translations
Pro Inside global      Inside local      Outside local   Outside global
tcp  88.88.88.88:80    172.16.0.5:80    ---           ---
tcp  88.88.88.88:666   172.16.0.6:22    ---           ---
```

Il livello Applicazione

Prefazione

I seguenti appunti sul livello applicativo di rete sono basati sul lavoro svolto dal Prof. Pecoraro e pubblicati sul [suo sito](#). A lui e a tutti gli autori e appassionati di conoscenza da lui citati nel sito, vanno i miei più sinceri ringraziamenti.

Libro vol.3 - Il livello delle applicazioni

STUDIARE: *L. Lo Russo, E. Bianchi, Sistemi e Reti - Nuova Edizione OPENSCHOOL - Per l'articolazione INFORMATICA degli Istituti Tecnici settore Tecnologico, vol. 3, ed. Hoepli, 2017*

- Il livello delle applicazioni nei modelli ISO/OSI e TCP/IP pp.2-8

Il protocollo HTTP

Libro vol.3 - Il livello delle applicazioni

STUDIARE: *L. Lo Russo, E. Bianchi, Sistemi e Reti - Nuova Edizione OPENSCHOOL - Per l'articolazione INFORMATICA degli Istituti Tecnici settore Tecnologico, vol. 3, ed. Hoepli, 2017*

- Il Web:HTTP e FTP pp.9-17

File Transfer Protocol: FTP

Libro vol.3 - Il livello delle applicazioni

STUDIARE: *L. Lo Russo, E. Bianchi, Sistemi e Reti - Nuova Edizione OPENSCHOOL - Per l'articolazione INFORMATICA degli Istituti Tecnici settore Tecnologico, vol. 3, ed. Hoepli, 2017*

- Il Web:HTTP e FTP pp.17-19

Domain Name System: DNS

Libro vol.3 - Il livello delle applicazioni

STUDIARE: *L. Lo Russo, E. Bianchi, Sistemi e Reti - Nuova Edizione OPENSCHOOL - Per l'articolazione INFORMATICA degli Istituti Tecnici settore Tecnologico, vol. 3, ed. Hoepli, 2017*

- Email, DNS e Telnet pp.25-28

DNS - Il servizio Directory di Internet

Esistono molti modi per **identificare un host**, e uno di questi è **il nome** dell'host. Nomi host come `cnn.com`, `www.yahoo.com`, `frapec.edu` e `cis.poly.edu` sono mnemonici e pertanto sono più semplici da ricordare per gli esseri umani. Tuttavia **i nomi di host forniscono poche informazioni sulla posizione dell'host all'interno di Internet**. Un hostname come `www.euro.fr`, che termina con il codice del paese `fr`, dice che l'host è probabilmente in Francia, ma non fornisce altre informazioni. Inoltre, poiché i nomi degli host possono essere costituiti da caratteri alfanumerici di lunghezza variabile, sarebbero difficili da trattare dai router. Per queste ragioni, **gli host sono identificati dagli indirizzi IP**.

Un indirizzo IPv4 consiste di quattro byte ed ha una struttura gerarchica rigida. Un indirizzo IPv4 assomiglia a `121.7.106.83`, dove ogni punto separa uno dei byte espressi in notazione decimale da 0 a 255. L'indirizzo IP è gerarchico, perché quando un router lo analizza da sinistra a destra utilizzando la netmask, ottiene informazioni sempre più specifiche su dove si trova l'host in Internet, cioè la rete di appartenenza.

Servizi forniti dal DNS

Un **host** può essere **identificato** in due modi distinti, tramite un **nome** e attraverso il suo **indirizzo IP**.

Un **operatore umano** preferirà l'identificatore di host più mnemonico, cioè il **nome**, mentre i **router** preferiscono gli **indirizzi IP di lunghezza fissa e gerarchicamente strutturati**. Per conciliare queste preferenze, serve un **servizio di directory** che **traduca i nomi host in indirizzi IP**. Questo è il **compito principale del** sistema dei nomi di dominio di Internet, o **Domain Name System (DNS)**.

Il **DNS è un database distribuito** implementato in una gerarchia di server DNS, ma **è anche un protocollo del livello di applicazione** che consente agli host di interrogare il database distribuito per **risolvere il nome di un dominio, ottenendo il corrispondente indirizzo IP**. Il **protocollo DNS è eseguito su UDP e utilizza la porta 53**.

DNS è comunemente impiegato da altri protocolli del livello applicazione, tra cui **HTTP (HyperText Transfer Protocol)**, **SMTP (Simple Mail Transfer Protocol)** e **FTP (File Transfer Protocol)**, con il compito di tradurre i nomi degli host forniti dall'utente in indirizzi IP. Come esempio, si consideri cosa succede quando un browser (cioè, un client HTTP), in esecuzione sull'host di qualche utente, richiede la risorsa `index.html` presso il dominio `www.scuola.edu/`, usando l'URL `www.scuola.edu/index.html`. Affinché l'host dell'utente sia in grado di inviare un messaggio di richiesta HTTP al server Web, deve prima ottenere l'indirizzo IP di `www.scuola.edu`. Questa operazione di **risoluzione del nome di un dominio** viene svolta considerando i seguenti punti:

1. sulla macchina utente viene gestito il lato client dell'applicazione DNS;
2. il browser estrae il nome host, `www.scuola.edu`, dall'URL e passa il nome host al lato client dell'applicazione DNS;
3. il client DNS invia al server DNS una interrogazione contenente l'hostname `www.scuola.edu`;

4. il client DNS riceve una risposta contenente l'indirizzo IP corrispondente all'hostname;
5. una volta che il browser riceve l'indirizzo IP dal DNS, può aprire una connessione TCP con il processo server HTTP in ascolto sulla porta 80 a quell'indirizzo IP.

Da questo esempio si vede che il **DNS aggiunge un ulteriore ritardo per le applicazioni Internet** che lo utilizzano. Fortunatamente, come si vedrà di seguito, **l'indirizzo IP desiderato è spesso memorizzato nella cache in un server DNS "vicino"**, che aiuta a ridurre il traffico di rete e il ritardo medio di risposta del DNS.

DNS fornisce alcuni altri importanti servizi, oltre a tradurre nomi di host in indirizzi IP (risolvere il nome di un dominio):

- **Host aliasing:** un host con un nome complicato può avere uno o più nomi alternativi. Ad esempio, un hostname come relay1.west-coast.enterprise.com potrebbe avere, ad esempio, due alias come enterprise.com e www.enterprise.com. In questo caso, il nome del computer relay1.westcoast.enterprise.com si dice che è un **hostname canonico**. I nomi di host alternativi, quando presenti, sono in genere più mnemonici dei nomi host canonici. Il servizio DNS può essere richiamato da una applicazione per ottenere l'hostname canonico per un nome alternativo fornito, così come l'indirizzo IP dell'host.
- **Mail server aliasing:** per ovvie ragioni, è altamente auspicabile che gli indirizzi di posta elettronica siano mnemonici. Ad esempio, se Bob ha un account con Hotmail, l'indirizzo e-mail di Bob potrebbe essere semplice bob@hotmail.com. Tuttavia, il nome host del server di posta elettronica Hotmail è più complicato e meno mnemonico rispetto a hotmail.com (per esempio, il nome host canonico potrebbe essere qualcosa di simile a relay1.west-coast.hotmail.com). Il DNS può essere richiamato da una applicazione di posta elettronica per ottenere l'hostname canonico per un hostname alternativo fornito, così come l'indirizzo IP dell'host. Infatti, il record MX (vedi sotto) permette al server e-mail e ai server Web di una società di avere hostname identici (alias); per esempio, il server Web e il server di posta di una società possono essere chiamati entrambi enterprise.com.
- **Load distribution:** il DNS è utilizzato anche per distribuire il carico tra i server replicati, come accade spesso per i server Web. I siti con un numero molto elevato di accessi, come ad esempio cnn.com, vengono replicati su più server, con ogni server in esecuzione su un sistema terminale diverso e ciascuno con un indirizzo IP diverso. **Per i server Web replicati un insieme di indirizzi IP viene associato ad un nome host canonico. Il database DNS contiene questa serie di indirizzi IP.** Quando i client interrogano il **DNS** per un nome corrispondente ad un insieme di indirizzi, il server risponde con l'intero insieme di indirizzi IP, ma **ruota l'ordine degli indirizzi all'interno di ciascuna risposta**. Poiché un client invia tipicamente il suo messaggio di richiesta HTTP all'indirizzo IP che viene nominato per primo nell'insieme, **la rotazione DNS distribuisce il traffico tra i server replicati**. La rotazione del DNS è utilizzata anche per la posta elettronica in modo che più server di posta elettronica possano avere lo stesso nome alternativo.

DNS: funzioni di rete critiche con il paradigma client-server

Come HTTP, FTP e SMTP, il protocollo **DNS è un protocollo del livello applicazione** perché viene eseguito per far comunicare sistemi terminali utilizzando il paradigma client-server, basandosi su un protocollo di trasporto end-to-end per il trasferimento di messaggi DNS tra i sistemi terminali comunicanti. Tuttavia, il ruolo del DNS è molto diverso dalle applicazioni Web (HTTP), dal trasferimento di file (FTP), e dalle e-mail (SMTP). A differenza di queste applicazioni, il **DNS non è un'applicazione con cui un utente interagisce direttamente**. Il **DNS fornisce una funzione Internet di base traducendo i nomi degli host nei relativi indirizzi IP**, per poterli utilizzare nelle applicazioni utente e altro software in Internet.

Gran parte della complessità dell'architettura Internet si trova ai "bordi" della rete. Il DNS, che implementa il processo di risoluzione del nome di un dominio (traduzione nome-indirizzo), utilizzando un'architettura client-server, si trova ai margini della rete confermando questa filosofia di design.

Panoramica del DNS⁵

Quando una applicazione client, come un browser Web o un lettore di posta, in esecuzione su un computer di un utente necessita di **risolvere** un nome host in un indirizzo IP, deve necessariamente avviare il lato client del DNS, specificando il nome host che deve essere tradotto. Ad esempio, su molte macchine basate su UNIX l'applicazione client richiama la funzione `gethostbyname()`. Il DNS client presente sullo host dell'utente si assume quindi l'incarico di inviare un messaggio di interrogazione in rete, utilizzando la porta 53 e il protocollo di trasporto UDP.

Dopo un ritardo, che varia da pochi millisecondi a qualche secondo, il DNS client sullo host dell'utente riceve un messaggio di risposta che fornisce la corrispondenza desiderata. Questa corrispondenza viene infine passata all'applicazione che ne necessitava e che aveva invocato il client DNS.

Dal punto di vista dell'applicazione chiamante sull'host dell'utente, il DNS è un *black-box* che fornisce un servizio di risoluzione utilizzando un gran numero di server DNS distribuiti in tutto il mondo, e avvalendosi di un protocollo del livello di applicazione che specifica come comunicano i server DNS e i client DNS sugli host che emettono l'interrogazione. L'applicazione chiamante ovviamente ignora completamente questo livello di complessità, utilizzando il servizio DNS all'occorrenza.

Affinché il servizio **DNS** sia in grado di svolgere effettivamente il suo compito di risoluzione **deve necessariamente basarsi su un database distribuito su più server**.

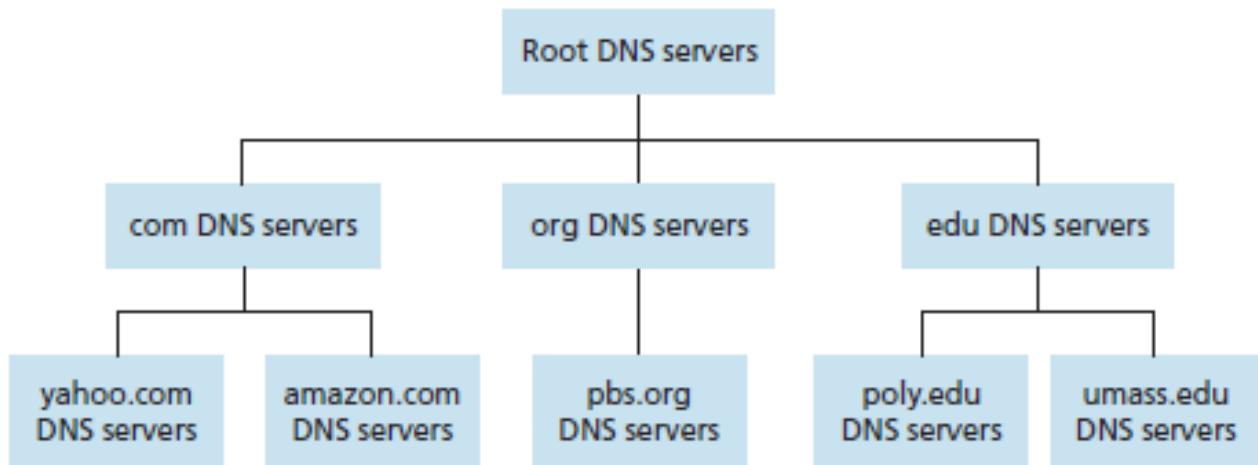
⁵ [How a DNS Server \(Domain Name System\) works.](#)

Un Database Gerarchico Distribuito

Per affrontare il problema dell'adattamento alle variazioni della rete il DNS utilizza un gran numero di server, organizzati in modo gerarchico e distribuito in tutto il mondo. In questo modo nessun server DNS ha tutte le corrispondenze per tutti gli host in Internet, distribuendo tra i diversi server DNS le associazioni nome-indirizzo.

In una prima approssimazione, i server DNS si suddividono in tre classi e ogni classe è preposta alla risoluzione di una porzione del nome di dominio, separate da un punto, considerando che ogni nome completo è terminato da un . che identifica proprio il dominio radice:

- **Server DNS root** (*Root DNS server*) - In Internet ci sono 13 server DNS root, etichettati da A a M, la maggior parte dei quali si trovano in Nord America. Un elenco degli attuali server DNS è disponibile tramite la [IANA](#) o tramite [Root Server Technical Operations Assn](#). Anche se ciascuno dei 13 server radice DNS viene immaginato come se fosse un unico server, ogni "server" è in realtà una rete di server replicati, per motivi di sicurezza e affidabilità. In totale, ci sono 247 server principali a partire dal 2011. Sono i server che ricevono per primi la query del client DNS e forniscono l'indirizzo IP del server DNS di primo livello da interrogare.
- **Server DNS di dominio di primo livello** (*TLD server - Top-Level Domain server*) - Questi server sono responsabili per i domini di primo livello come [.com](#), [.org](#), [.net](#), [.edu](#), [.gov](#) e tutti i domini di primo livello dei nomi dei paesi, quali [.uk](#), [.fr](#), [.ca](#), [.jp](#), [.it](#). La società [Verisign Global Registry Services](#) mantiene i server TLD per il dominio [.com](#) di primo livello, e la società [Educause](#) mantiene i server TLD per il dominio di primo livello [.edu](#). Presso il sito della [IANA](#) è possibile vedere un elenco di tutti i domini di primo livello. Questi server restituiscono l'indirizzo IP del server autorevole in grado di risolvere la query del client DNS.
- **Server DNS autorevole di dominio di secondo livello** (*Authoritative DNS server*) - Ogni organizzazione con dei server accessibili al pubblico (ad esempio un server Web o un server di posta) tramite Internet deve fornire i **record DNS** accessibili pubblicamente che mappano i nomi di tali host in indirizzi IP. Alcuni **server DNS di fiducia di un'organizzazione ospitano questi record DNS**. Un'organizzazione può scegliere di eseguire il proprio server DNS di fiducia per tenere questi record o, in alternativa l'organizzazione può pagare per avere questi record archiviati in un server DNS di qualche fornitore di servizi. La maggior parte delle università e grandi aziende implementano e mantengono i propri server DNS autorevoli primario e secondario (quest'ultimo di backup). Questi server risolvono il dominio di secondo livello ([yahoo.com](#) del dominio di primo livello [.com](#)) e quindi forniscono la risposta definitiva di risoluzione del dominio al client DNS.



Per capire come queste tre classi di server interagiscono, si supponga che un client DNS voglia determinare l'indirizzo IP del nome host www.amazon.com. In prima approssimazione, si svolgeranno i seguenti eventi:

- il client DNS contatta uno dei DNS root server, che restituisce gli indirizzi IP dei server TLD per il dominio di primo livello .com;
- il client DNS contatta quindi uno di questi server TLD che restituirà l'indirizzo IP di un server autorevole per la risoluzione del dominio amazon.com;
- infine il client DNS contatta uno dei server autorevoli per amazon.com che restituirà l'indirizzo IP per il nome host www.amazon.com.

Il root DNS server, il TLD server e il server DNS autorevole appartengono tutti alla gerarchia di server DNS, ma esiste un altro importante tipo di server DNS denominato **server DNS locale**, che pur non appartenendo alla gerarchia dei server DNS, è comunque **fondamentale per l'architettura DNS**. Ogni ISP - come un'università, un dipartimento accademico, aziendale o un ISP residenziale dispone di un **server DNS locale**. Quando un host si connette a un ISP, questo fornisce gli indirizzi IP di uno o più dei suoi server DNS locali, tipicamente tramite DHCP. Il **server DNS locale di un host è tipicamente vicino allo host**. Ad esempio, per un ISP istituzionale il server DNS locale può essere sulla stessa LAN dell'host, mentre per un ISP residenziale sarà separato dall'host da qualche router.

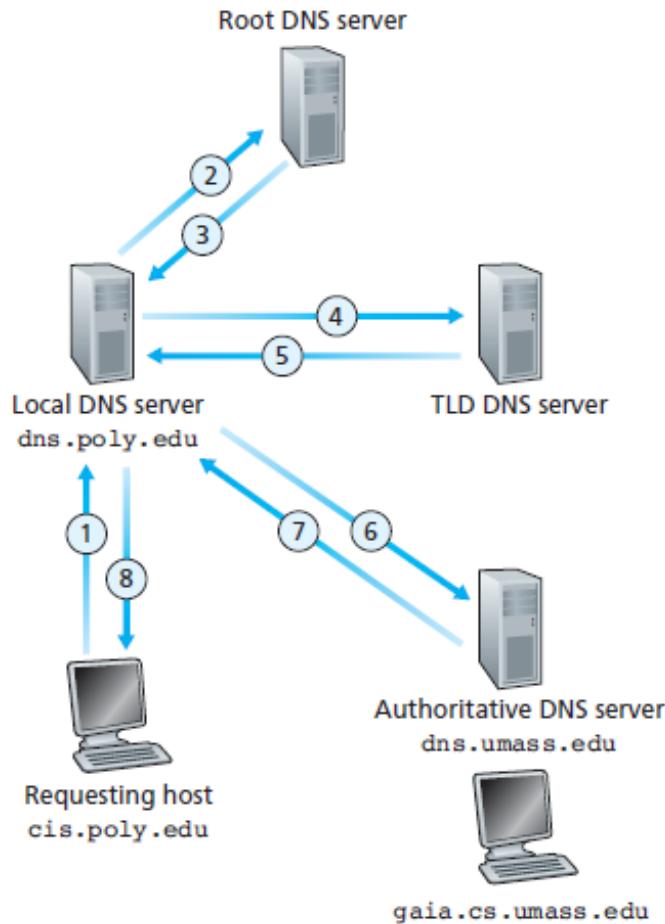
Quando un host effettua una query DNS, la query viene inviata prima di tutto al server DNS locale, che ricopre anche il ruolo di proxy. Se il server DNS locale non è in grado di risolvere il nome di dominio, non trovando un record nella sua cache per risoluzioni precedenti, allora inoltra la query alla gerarchia di server DNS.

Interrogazioni iterative e ricorsive

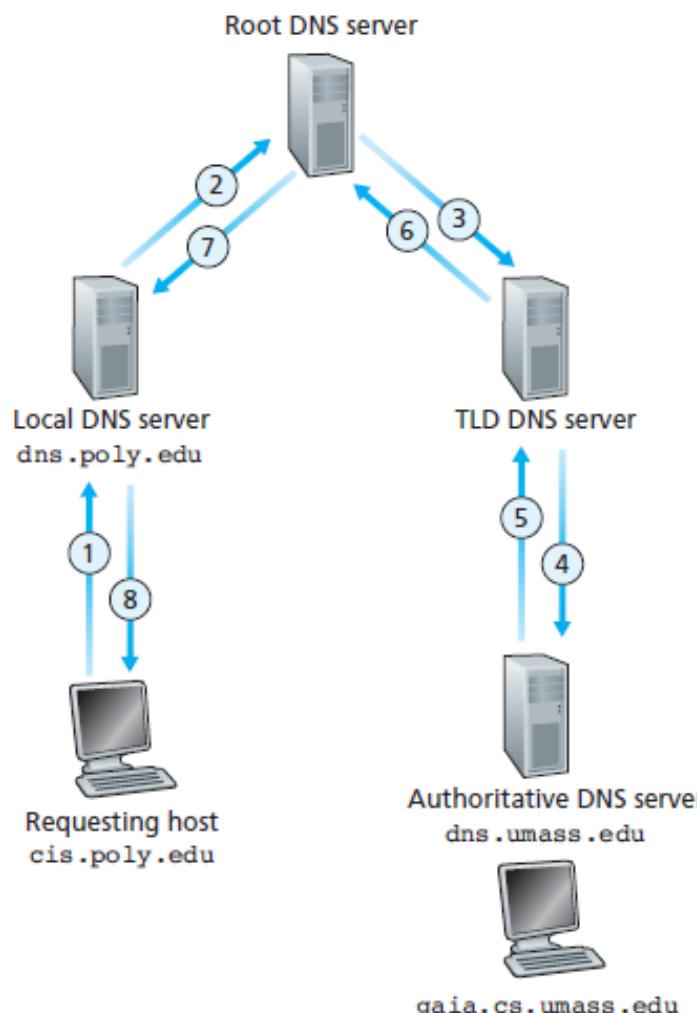
Le interrogazioni poste alla gerarchia dei server DNS possono essere risolte utilizzando due metodi:

- **iterativo**: lo host si occupa di interrogare direttamente ogni singolo livello della gerarchia dopo aver ricevuto l'indirizzo IP del livello da interrogare;
- **ricorsivo**: ogni server della gerarchia si occupa di portare avanti l'interrogazione per conto dello host.

Nella realtà viene utilizzato un metodo ibrido, utilizzando il **metodo ricorsivo tra lo host e il server DNS locale**, e avvalendosi del **metodo iterativo per tutti i server della gerarchia DNS**.

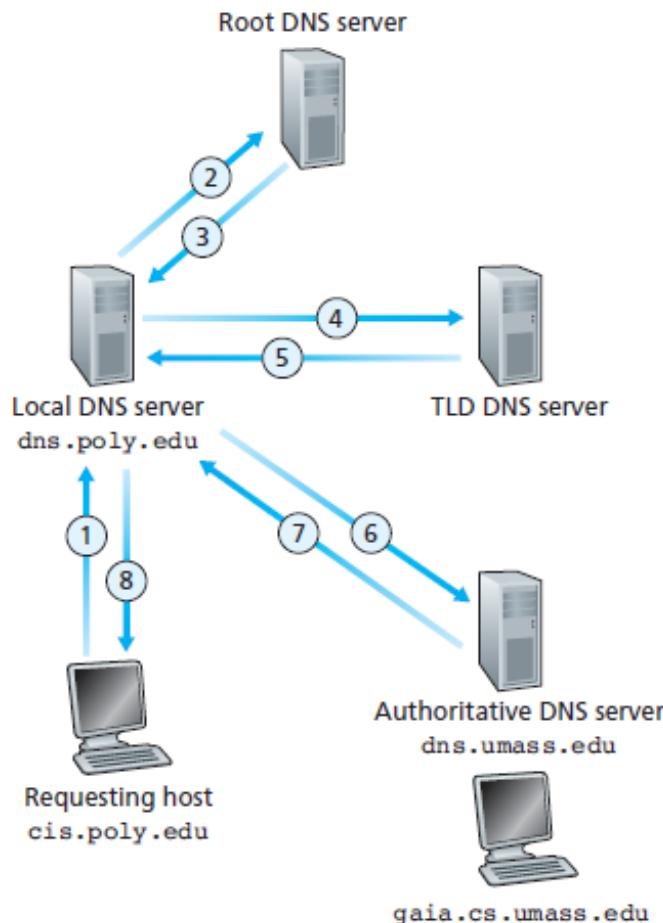


Nell'immagine seguente viene invece mostrato, per completezza, un esempio di interrogazione ricorsiva.



DNS Caching

Finora si è ignorata la presenza della **cache DNS**, una **caratteristica estremamente importante del sistema DNS**. In verità, il DNS sfrutta ampiamente il caching per migliorare le prestazioni in termini di ritardo, e allo scopo di ridurre il numero di messaggi DNS scambiati su Internet. L'idea alla base del caching DNS è molto semplice; in una catena di query, quando un server DNS riceve una risposta DNS contenente, ad esempio, una corrispondenza fra un nome host e un indirizzo IP, può memorizzare nella cache locale la corrispondenza. Per esempio, nell'esempio mostrato in figura, ogni volta che il server DNS locale dns.poly.edu riceve una risposta da qualche server DNS, può memorizzare nella cache qualsiasi informazione in essa contenuta.



Se una coppia nome dominio/indirizzo IP viene memorizzata nella cache di un server DNS locale, tutte le successive richieste per la risoluzione del medesimo dominio potranno essere soddisfatte direttamente dal server DNS locale, anche se non è autorevole per il nome di dominio. Chiaramente, dopo un certo tempo queste associazioni non autorevoli dovranno essere eliminate e sostituite in quanto le associazioni tra nomi host e indirizzi IP non sono permanenti. **Il caching delle associazioni nome dominio/indirizzi IP permette di velocizzare la risoluzione di un nome di dominio.**

Si consideri che un server DNS locale può anche memorizzare nella cache gli indirizzi IP dei server TLD, consentendo in tal modo al server DNS locale di saltare un livello nella sequenza delle interrogazioni.

Record DNS [leggere]

I server DNS che insieme implementano il database DNS distribuito, memorizzano **record di risorse** (*RR - Resource Record*), compresi i record che forniscono la corrispondenza tra nome di dominio a indirizzo IP. Ogni messaggio di risposta DNS trasporta uno o più record di risorse.

Un **record di risorsa** è un tupla che contiene i seguenti campi:

(Nome, Valore, Tipo, TTL)

dove **TTL** è il **tempo di vita del record di risorse** e determina quando una risorsa deve essere rimossa da una cache. Negli esempi seguenti questo campo verrà tralasciato per chiarezza espositiva.

Il significato dei campi **Nome** e del campo **Valore** dipende dal campo **Tipo**. Di seguito vengono elencati alcuni [tipi di record DNS](#).

- Se **Tipo = A**, allora **Nome** è un nome host e **valore** è l'indirizzo IP di quello host. Così, un **record di Tipo A** fornisce la corrispondenza standard hostname-IP. A titolo di esempio, (`relay1.bar.foo.com`, `145.37.93.126`, `A`) è un record di tipo A.
- Se **Tipo = NS**, allora **Nome** è un dominio (ad esempio `foo.com`) e **valore** è l'hostname di un server DNS autorevole che sa come ottenere gli indirizzi IP per gli host nel dominio. Questo record è utilizzato per inoltrare le query DNS nella catena di query. Ad esempio, (`foo.com`, `dns.foo.com`, `NS`) è un **record di tipo NS**.
- Se **Tipo = CNAME**, allora il **Valore** è un **nome host canonico per il nome alternativo (alias) dello hostname**. Questo record può fornire agli host il nome canonico per un ottenere l'hostname. A titolo di esempio, (`foo.com`, `relay1.bar.foo.com`, `CNAME`) è un **record di tipo CNAME**.
- Se **Tipo = MX**, allora il **valore** è il **nome canonico di un server di posta che ha come alias Nome**. A titolo di esempio, (`foo.com`, `mail.bar.foo.com`, `MX`) è un **record di Tipo MX**. I Record MX consentono ai nomi host del server di posta elettronica di avere alias semplici. Si noti che utilizzando il record MX, una società può avere lo stesso nome alias per il server di posta e per uno dei suoi altri server (come ad esempio il proprio server Web). Per ottenere il nome canonico per il server di posta elettronica, un client DNS potrebbe interrogare per un record MX; per ottenere il nome canonico per l'altro server, il client DNS interroga per il record CNAME.

Utilizzando a linea di comando il comando `nslookup` è possibile inviare una query DNS a qualsiasi server DNS (root, TLD, autorevole o non autorevole). Dopo aver ricevuto il messaggio di risposta dal server DNS, `nslookup` visualizzerà i record inclusi nella risposta, in un formato leggibile. In ambiente Linux è disponibile anche il comando `dig`, più recente rispetto al comando `nslookup`.

Vulnerabilità DNS

Il DNS è un componente fondamentale dell'infrastruttura di Internet, con molti servizi importanti, tra cui il Web e la posta elettronica, incapaci di funzionare senza un servizio di risoluzione dei nomi di dominio. Per questa ragione il servizio DNS è spesso soggetto ad attacchi di diversa natura.

Il primo tipo di attacco che viene in mente è un **attacco DDoS (Distributed Denial of Service)** di **intasamento della banda contro i server DNS**. Ad esempio, un utente malintenzionato può tentare di inviare ad ogni server DNS principale una gran quantità di pacchetti, così tanti che la maggior parte delle query DNS legittime non otterrà mai risposta. Questo attacco DDoS contro server

radice DNS ha effettivamente avuto luogo il 21 Ottobre 2002⁶. In questo attacco, gli aggressori inviarono ininterrottamente messaggi ping ICMP a ciascuno dei 13 root server DNS. Fortunatamente, questo attacco su larga scala ha causato danni minimi, avendo poco o nessun impatto sull'utilizzo della rete da parte degli utenti. Gli aggressori riuscirono a dirigere un flusso enorme di pacchetti ai server principali, ma molti dei root DNS server erano protetti da filtri di pacchetti, configurati per bloccare sempre tutti i messaggi ping ICMP diretti ai server principali. Questi server protetti furono così risparmiati e funzionarono normalmente. Inoltre, la maggior parte dei server DNS conteneva nella cache locale gli indirizzi IP dei server di dominio di primo livello, permettendo al processo di query di evitare spesso i root DNS server.

Un **attacco DDoS** potenzialmente più efficace contro il DNS sarebbe quello di **inviare un flusso di query DNS ai server di dominio di primo livello**, per esempio a tutti i server che gestiscono il dominio .com, in quanto risulta più difficile filtrare delle query legittime inviate al server DNS, e i server di primo livello non sono così facilmente agirabili come lo sono i server root. La gravità di un tale attacco sarebbe parzialmente mitigata dalla memorizzazione nella cache dei server DNS locali.

Il DNS potrebbe potenzialmente essere attaccato in altri modi. In un **attacco man-in-the-middle, l'attaccante intercetta le richieste da un host e restituisce le risposte false**.

In un attacco **DNS poisoning, l'attaccante invia risposte false a un server DNS, ingannando il server ad accettare i record falsi nella sua cache**. Uno di questi attacchi potrebbe essere utilizzato, ad esempio, per reindirizzare un utente ignaro al sito Web dell'utente malintenzionato. Questi attacchi, però, sono di difficile attuazione, in quanto richiedono di intercettare pacchetti o superare i firewall.

Un altro importante **attacco sfrutta l'infrastruttura DNS per lanciare un attacco DDoS contro un obiettivo diverso**, per esempio server di posta. In questo attacco, l'attaccante invia query DNS per molti server DNS autorevoli, mettendo in ogni query l'indirizzo sorgente contraffatto dell'host di destinazione. I server DNS quindi inviano le loro risposte direttamente all'host di destinazione. Se le query possono essere realizzate in modo tale che una risposta è molto più grande (in byte) di una query (cosiddetta amplificazione), allora l'attaccante può potenzialmente sopraffare la porta senza dover generare gran parte del proprio traffico. Tali attacchi DNS hanno avuto un successo limitato finora.

In sintesi il protocollo applicativo DNS si è dimostrato sorprendentemente robusto contro gli attacchi e, ad oggi non vi è stato un attacco che ha impedito con successo il servizio DNS. Ci sono stati attacchi efficaci, ma questi possono essere e sono affrontati con un'adeguata configurazione dei server DNS.

⁶ William F. Slater III, [*The Internet Outage and Attacks of October 2002*](#), writing del 7 novembre 2002.

La posta elettronica

Libro vol.3 - Il livello delle applicazioni

STUDIARE: *L. Lo Russo, E. Bianchi, Sistemi e Reti - Nuova Edizione OPENSCHOOL - Per l'articolazione INFORMATICA degli Istituti Tecnici settore Tecnologico, vol. 3, ed. Hoepli, 2017*

- Il Web:HTTP e FTP pp.20-25 e pp.28-33

Protocols and other networking concepts - CLIL

CLIL Listening and Writing - Protocols

The following video give an overview about what Internet protocols are. Listen each video and answer to the questions. At the end of the work send the answer to your teacher.

Protocols

1. What is a protocol? [What is a protocol?](#)
2. Why is so important that protocols are written? [What is a protocol?](#)
3. Which are the two things that a protocol has to do? [What is a protocol?](#)
4. Why are protocols so highly structured? [Why are protocols so highly structured?](#)
5. What are clients and servers and which is the difference between them? [What is a client? What is a server?](#)
6. What is the Internet protocol? [What is the internet protocol?](#)
7. Referring to IP header which is the meaning of Source IP address and Destination IP address? [What is the internet protocol?](#)
8. Which is the meaning that IP is a best-effort protocol? [What is the internet protocol?](#)
9. Is IP protocol guarantee packets ordered delivery? [What is the internet protocol?](#)
10. Is it possible to use an IP address in the browser URL bar? [What is an Internet protocol \(IP\) address?](#)
11. Which features are provided by TCP? [What is the internet protocol suite?](#)
12. Why all the TCP features are not provided into IP? [What is the internet protocol suite?](#)
13. What is the end-to-end principle? [What is the end-to-end principle?](#)
14. Which are the two benefits of the end-to-end principle? [What is the end-to-end principle?](#)
15. What is the robustness principle? [What is the robustness principle?](#)
16. Regarding Internet protocol stack, explain what the two goals "Separation of concern" and "Reuse of good ideas" are. [What is the internet protocol stack?](#)
17. Explain what encapsulation is. [What is encapsulation?](#)
18. Why is ICMP used? [What are some examples of other internet protocols?](#)
19. What is the purpose of NTP protocol? [What are some examples of other internet protocols?](#)
20. Which is the difference between FTP and BitTorrent protocols? [What are some examples of other internet protocols?](#)
21. How can UDP be used other then its normal purpose? [Can you still deploy new transport protocols?](#)
22. What is a peer-to-peer infrastructure? [What is a peer to peer system?](#)

CLIL Listening - And other networking concepts

You should have understood how to respond to a question after watching a video.

You have to watch the following playlist, they are useful to review fundamental networking concepts.

1. [Connectivity](#)

2. [Routing](#)

3. [Transport](#)

- a. The teacher speaks often about naming, referring to port number using. This concept is also referred to the concepts of multiplexing and demultiplexing.
- b. In lesson 18 is covered the concept of Network Address Translation (NAT) that we are going to cover in the near future. Grasp the concept, it will be also useful in your future laboratory assignments.

4. [Naming](#)

- a. In lesson 6 the teacher gives a brief explanation about IPv6.
- b. In lesson 7 is given a brief explanation about DHCP.

5. [Web](#)

The following videos are useful to understand network subnetting.

1. [Subnetting: the magic number](#)

2. [Basic VLSM](#)

3. [VLSM Example](#)

4. [VLSM B Class Made Easy](#)

5. [Step 4 learning VLSM with class A](#)

VLAN: Virtual LAN



Watch the video, it will help!

Prefazione

I seguenti appunti sono basati sul lavoro svolto dalla Prof.ssa Sophia Danesino, mentore, amica, appassionata di conoscenza e insegnante fuori dal comune. Sul suo [canale YouTube](#) è possibile trovare i video su molti degli argomenti trattati.

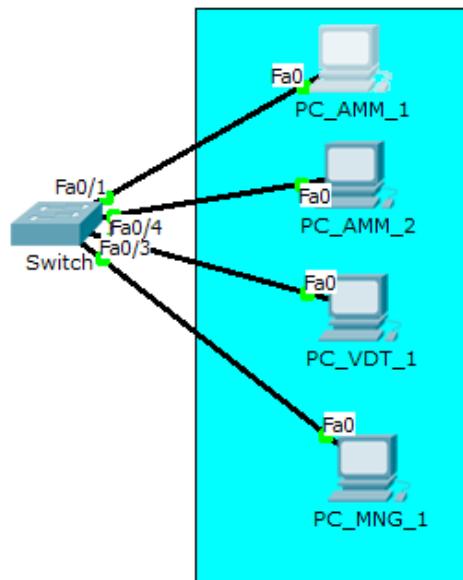
VLAN: Virtual Local Area Network

Libro vol.3 - Le Virtual LAN (VLAN)

LEGGERE: L. Lo Russo, E. Bianchi, Sistemi e Reti - Nuova Edizione OPENSCHOOL - Per l'articolazione INFORMATICA degli Istituti Tecnici settore Tecnologico, vol. 3, ed. Hoepli, 2017

- Virtual LAN pp.66-67
- Realizziamo una VLAN pp.67-71

VLAN - Definizione e ambiti di applicazione



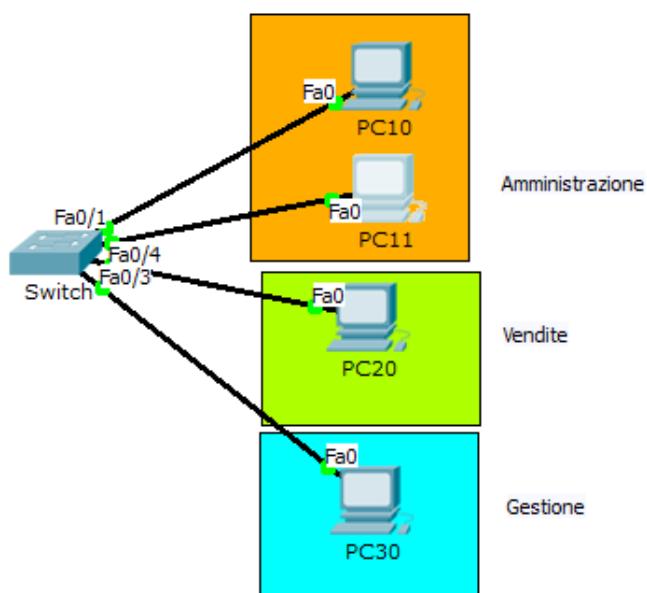
Per comprendere il funzionamento delle **VLAN** o **LAN Virtuali** consideriamo il seguente esempio. Supponiamo di avere una LAN a cui sono collegati gli host mostrati nell'immagine a sinistra.

In questa configurazione è presente **un unico dominio di broadcast poiché gli host sono attestati allo stesso switch**. Un **dominio di broadcast crea molto traffico in rete**, basti pensare alle richieste ARP usate quando il MAC address destinazione di un frame non è noto. Il traffico di tipo broadcast colpisce l'intera rete poiché ciascun dispositivo che riceve un frame

broadcast è costretto ad analizzarlo. Se la frequenza dei messaggi broadcast dovesse crescere, la banda disponibile diminuirebbe sensibilmente.

Supponiamo ora che gli host vengano suddivisi in base al gruppo di lavoro, ad esempio Amministrazione, Vendite e Gestione.

Per separare il traffico tra i vari gruppi la soluzione potrebbe essere la creazione di subnet diverse, ma gli switch, non interpretando gli indirizzi a livello 3, non risolverebbero il

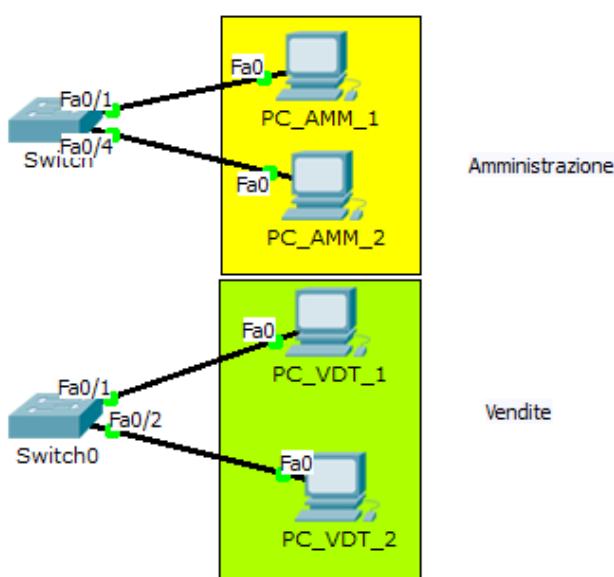


problema. Inoltre non sarebbe possibile applicare dei filtri di controllo: chi cambiasse il proprio IP potrebbe cambiare liberamente da una subnet all'altra senza alcun problema.

Per separare i domini di broadcast dovremmo separare fisicamente l'infrastruttura di rete: aggiungere degli switch e attestare fisicamente gli host su segmenti di rete separati.

In alternativa è possibile definire nell'esempio precedente, via software, **tre VLAN sullo switch, creando tre reti logiche e tre domini di broadcast sullo stesso segmento di rete**.

Per definizione, **le VLAN rappresentano un metodo per segmentare un dominio di broadcast in più domini di dimensione ridotta**. A livello 2 ogni VLAN contiene solo il traffico dei dispositivi appartenenti a quella VLAN.

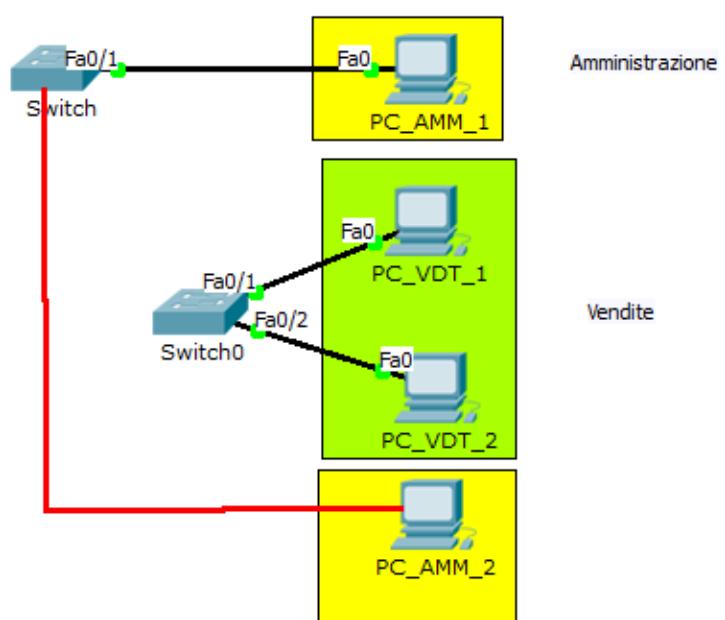


Osserviamo ora lo scenario di lato in cui sono presenti due gruppi di lavoro posti su due piani diversi dello stesso edificio e associati a due gruppi di lavoro diversi, Amministrazione al primo piano e Vendite al piano terra.

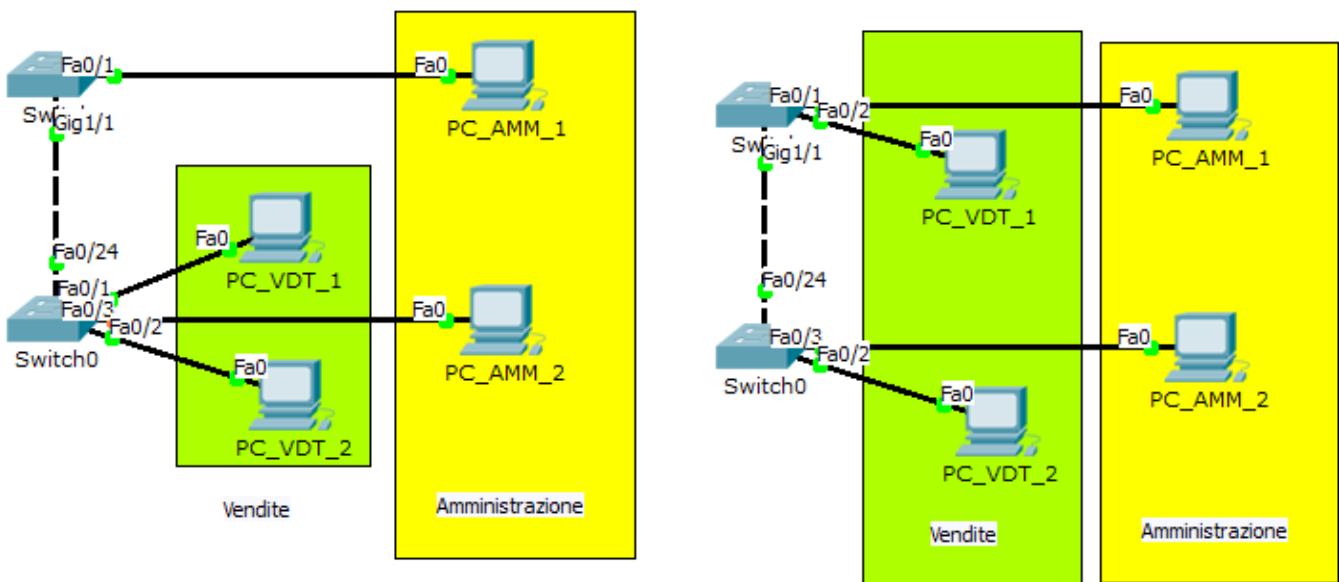
Immaginiamo che l'utente PC_AMM_2 cambi ufficio spostandosi al piano terra. Per associarlo allo switch corretto si dovrebbe ricablagare l'edificio attestando il suo host all'altro switch (all'altro dominio di broadcast).

Un processo analogo avviene se l'utente cambia gruppo di lavoro senza cambiare ufficio. Questa azione andrebbe ripetuta per ogni host che debba cambiare l'appartenenza ad un gruppo di lavoro.

Definendo delle VLAN (LAN virtuali) è possibile definire gruppi logici di host, anche se questi sono separati da switch e fanno parte di segmenti di LAN differenti. **Gli host che fanno parte di una VLAN comunicano tra loro come se si trovassero su uno stesso segmento LAN fisico**. Nell'esempio precedente



sarà quindi sufficiente un unico collegamento tra i due switch per permettere la definizione di molteplici configurazioni di VLAN, anche a fronte di spostamenti fisici dei dispositivi, come mostrato nelle figure seguenti:



In questo modo **la segmentazione di reti switched è definita in funzione delle attività svolte e dei gruppi di lavoro, a prescindere dalla collocazione fisica degli host rispetto alle connessioni fisiche della rete**. Quando si vorrà spostare un host da una VLAN ad un'altra non sarà necessario collegare fisicamente l'host ad un altro switch, sarà sufficiente assegnare la porta dello switch ad un'altra VLAN.

Le VLAN consentono quindi di definire soluzioni quasi completamente indipendenti dalle topologie fisiche, permettendo a più reti logiche di condividere la stessa infrastruttura fisica.

Le VLAN definiscono logicamente la segmentazione della rete in diversi domini di broadcast in modo da eseguire operazioni di switching solo tra le porte assegnate alla stessa VLAN. In altre parole **le porte di una VLAN condividono i broadcast, mentre VLAN differenti non condividono alcun broadcast**. Questo consente di migliorare le prestazioni complessive della rete.

I vantaggi delle VLAN possono essere sintetizzati nel modo seguente:

- **Facilità di gestione delle infrastrutture di rete:** invece di spostare cavi, riposizionare uplink, aggiungere dispositivi e ricablarne intere zone, si gestiscono le VLAN tramite strumenti software.
- **Scalabilità:** le VLAN possono essere aggiunte utilizzando le porte esistenti dello switch e quindi a costo nullo, rendendo semplice e relativamente economica l'espansione della rete.
- **Flessibilità:** le porte dello switch possono essere spostate da una VLAN ad un'altra per mezzo di semplici operazioni di riconfigurazione software, spesso effettuabili da remoto; se si desidera isolare una subnet non è necessario aggiungere uno switch e/o un router, ma sarà sufficiente riassegnare alcune porte.
- **Economicità:** con uno switch, si può effettuare routing tra le VLAN.

- **Diminuzione del traffico di rete:** tramite VLAN si confina facilmente il broadcast.
- **Sicurezza:** se un utente cambiasse il proprio IP per fare parte di un'altra subnet, resterebbe del tutto isolato poiché separato dalle altre VLAN a livello 2.

I dispositivi di una VLAN sono in grado di comunicare solo con altri dispositivi che si trovano nella stessa VLAN, a meno che nella rete sia presente un router configurato in modo da attivare il routing inter-VLAN o uno switch layer 3 che offre funzionalità a livello 3. In quest'ultimo caso lo switch gestisce una tabella di forwarding per ciascuna VLAN.

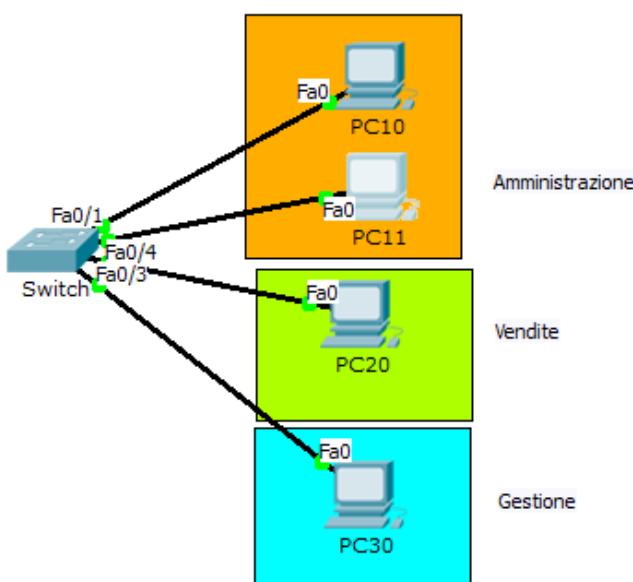
Configurazione di una VLAN

1 - Laboratorio VLAN: simulazione con Packet Tracer

[Video della Prof.ssa Sophia Danesino.](#)

Consideriamo il seguente scenario:

Nome PC	IP	Settore
PC10	192.168.0.10/24	Amministrazione
PC11	192.168.0.11/24	Amministrazione
PC20	192.168.0.20/24	Vendite
PC30	192.168.0.30/24	Gestione



A livello network abbiamo definito un'unica rete (192.168.0.0/24) quindi un ping tra i quattro host avviene correttamente. I quattro host appartengono anche allo stesso dominio di broadcast perché sono collegati allo stesso switch.

Vogliamo creare tre gruppi logici sullo stesso rete fisica creando tre VLAN: Amministrazione, Vendite e Gestione.

Per configurare le VLAN è sufficiente:

- creare le VLAN;
- associare le porte dello switch alla relativa VLAN.

Ogni **VLAN** è identificata da un identificatore **VID** o **VLAN ID**, il cui valore va da 1 a 4094, e da un nome. La **VLAN 1** è quella predefinita dal costruttore a cui appartengono di default tutte le porte dello switch (**default VLAN**).

La **Default VLAN** ha le seguenti caratteristiche:

- esiste sempre e non può essere eliminata;
- esiste perché lo switch ha bisogno di almeno una VLAN a cui assegnare le sue porte, ed inoltre è utilizzata per protocolli speciali quali VTP e DTP⁷;
- inizialmente tutte le porte sono untagged member della VLAN di default;

⁷ VLAN Tunking Protocol (VTP) è un protocollo proprietario di trunking, sviluppato dalla Cisco Systems e sarà illustrato successivamente. Il Dynamic Trunking Protocol (DTP) è un protocollo proprietario di trunking, sviluppato dalla Cisco Systems. È utilizzato per la negoziazione dinamica di collegamenti in trunk (che trasportano più VLAN), generalmente tra due switch (fonte: Wikipedia).

- la **VLAN di default ha ID 1**;
- l'ID della VLAN di default può essere cambiato, ma è discutibile il farlo;
- se una porta non è più membro di alcuna VLAN, automaticamente diventa untagged member della VLAN di default (una porta non appartiene più a una determinata VLAN quando è rimossa dalla VLAN stessa, oppure se quella VLAN viene eliminata).

Nel seguito sono definiti i comandi⁸ per creare la VLAN di cui si è illustrata precedentemente la configurazione.

Creazione VLAN

```
Switch>enable  
Switch#configure terminal
```

Entro in modalità configurazione VLAN identificando la nuova VLAN con VID 10.

```
Switch(config)#vlan 10
```

Assegno un nome alla VLAN 10.

```
Switch(config-vlan)#name Amministrazione  
Switch(config-vlan)#exit
```

Ripeto lo stesso procedimento per tutte le altre VLAN.

```
Switch(config)#vlan 20  
Switch(config-vlan)#name Vendite  
Switch(config-vlan)#exit  
Switch(config)#vlan 30  
Switch(config-vlan)#name Gestione  
Switch(config-vlan)#exit  
Switch(config) #
```

Associazione porte - VLAN

A questo punto è necessario assegnare le VLAN a un certo numero di porte. **Una porta può appartenere a una sola VLAN alla volta.** Queste porte sono dette **Access Port** o **Untagged Port**. I frame che attraversano una porta untagged (o access) sono privi del tag VLAN⁹. In questa modalità l'interfaccia può essere untagged member di una sola VLAN.

Assegno la porta ad una VLAN utilizzando il VID che la identifica.

```
Switch(config)#interface FastEthernet0/1  
Switch(config-if)#switchport access vlan 10  
Switch(config-if)#exit  
Switch(config)#interface FastEthernet0/4  
Switch(config-if)#switchport access vlan 10
```

⁸ Durante le esercitazioni è stato consentito l'uso degli appunti e la consultazione dei manuali Cisco per quanto riguarda la sintassi dei comandi.

⁹ Nei frame tagged che attraversano un link impostato come TRUNK viene aggiunto un campo iniziale che permette allo switch di capire qual è la VLAN di appartenenza, ma nei frame untagged, come in questo esercizio, questo campo addizionale non è presente.

```
Switch(config-if)#exit
Switch(config)#interface FastEthernet0/2
Switch(config-if)#switchport access vlan 20
Switch(config-if)#exit
Switch(config)#interface FastEthernet0/3
Switch(config-if)#switchport access vlan 30
Switch(config-if)#exit
```

Le macchine non associate alle VLAN 10, 20, 30 appartengono alla VLAN 1.

Per **esaminare** quali **VLAN** sono state **definite e a quali porte** sono **associate** è possibile usare il comando

```
Switch#show vlan brief
```

Non sarà possibile effettuare un ping tra host che appartengono a VLAN diverse nonostante appartengano alla stessa rete IP. **Un ping in broadcast è limitato alla VLAN da cui è stato inviato.**

E' anche possibile associare gruppi di interfacce ad una VLAN:

```
Switch(config)#interface range FastEthernet0/1 - 5
Switch(config-if-range)#switchport access vlan 10
```

Rimozione porte e VLAN

Rimozione di una porta da una VLAN

Quando una **porta** viene **rimossa da una VLAN** viene **automaticamente riassegnata alla VLAN 1** di default. Di seguito vengono mostrati i comandi per la rimozione di una porta da una VLAN:

```
Switch(config)#interface FastEthernet0/2
Switch(config-if)#no switchport access vlan 20
Switch(config-if)#exit
```

Rimozione di una VLAN

Di seguito vengono mostrati i comandi per la rimozione di una VLAN:

```
Switch(config)#no vlan 20
```

Some useful videos - VLAN Concepts

[VLAN Concepts](#)

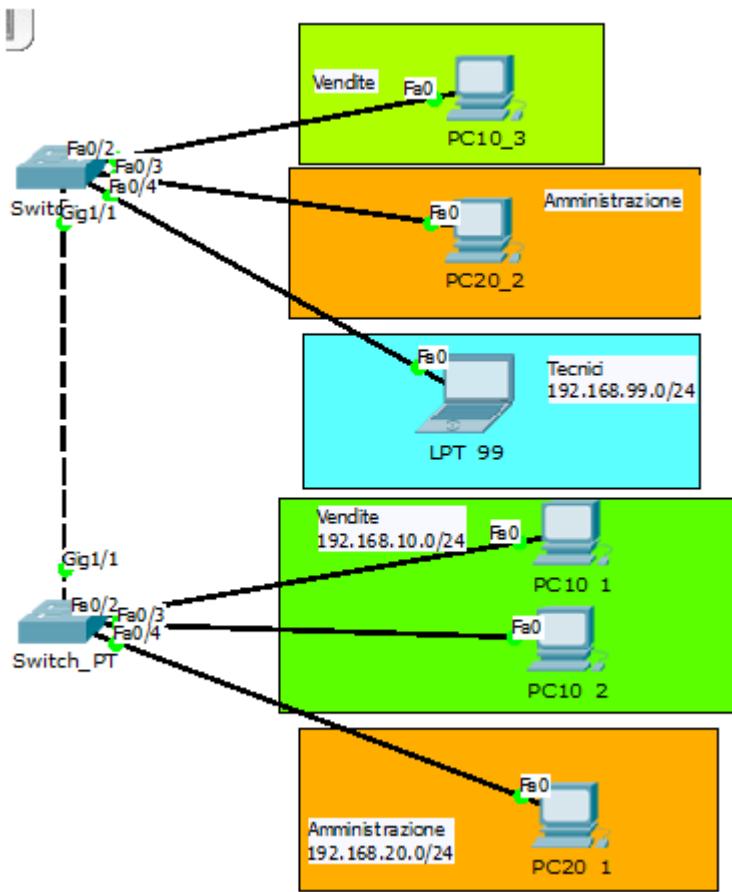
Comprehension Activity

2 - VLAN - Access/Untagged ports

Using Packet Tracer create the scenario shown in [VLANs and Trunks for Beginners - Part 1](#) video. Talk with your classmate about new concepts and new commands [*you will have found*](#)¹⁰ in the video, such as switchport mode access and show vlan, and add all of them in your personal CLI manual. In order to be coherent with names, you can call the VLAN 50 using the name **data** instead of **student**, or vice versa, or you have only to remember that the speaker uses the label **data** for the VLAN named **student**.

¹⁰ Grammar - Future Perfect

VLAN Trunking



In pratica un solo link fisico tra due switch è in grado di supportare il traffico tra qualsiasi VLAN.

Un LINK impostato come TRUNK non appartiene ad una VLAN particolare, ma è un canale che le VLAN possono utilizzare per collegare switch e router per lo scambio di frame tra VLAN diverse.

Per gestire il traffico tra qualsiasi VLAN tramite un link trunk viene aggiunto un tag a ciascun frame inviato al link, che consente di identificare la VLAN di appartenenza. Il **VLAN trunking** si basa su una serie di specifiche standard: uno dei protocolli più utilizzati è **IEEE 802.1Q**.

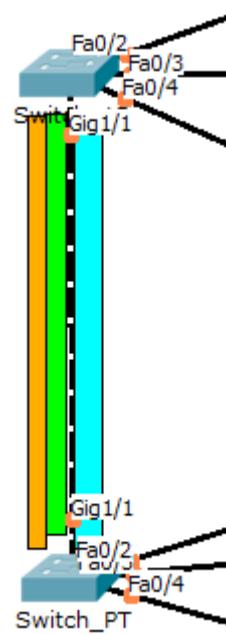
Affinché un link di collegamento tra due switch venga usato in modo promiscuo per il trasporto dei frame di tutte le VLAN definite, è necessario impostare l'interfaccia in modalità **trunk** invece di **access**.

Quindi, da quanto emerso finora, è possibile configurare le porte dello switch in una delle modalità seguenti:

Immaginiamo ora lo scenario mostrato di lato in cui a due switch sono collegati host appartenenti a VLAN diverse. Per far **colloquiare host appartenenti alla stessa VLAN** ma collegati a switch diversi, **sarebbe necessario un collegamento crossover tra i due switch per ogni VLAN**.

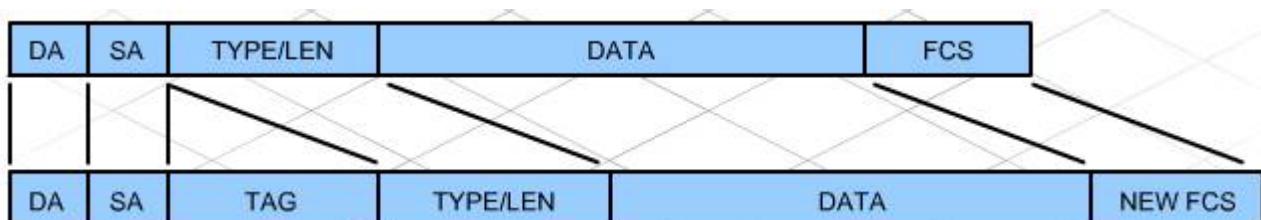
Poiché questo metodo è molto laborioso è stata sviluppata una tecnica, detta **VLAN trunking**, che **consente di trasferire i frame appartenenti a VLAN diverse collegando i due switch con un'unica connessione fisica**.

Un trunk è una connessione fisica e logica tra due switch (link punto-a-punto) in grado di supportare la presenza di più VLAN. In



- **Access o untagged** - in questa modalità l'interfaccia può essere untagged member di una sola VLAN. Access Port o Untagged Port sono definizioni equivalenti. I pacchetti che attraversano una porta untagged (o access) sono privi del tag VLAN, come quello definito dallo standard 802.1Q.
- **Trunk o tagged** - l'interfaccia è tagged member di una o più VLAN in quanto usata in modo promiscuo per trasportare i frame di molteplici VLAN. Una porta tagged è in grado di ricevere pacchetti "taggati", e su queste porte possono essere collegati solo dispositivi in grado di interpretare i VLAN tag, come switch, router e firewall compatibili con il protocollo 802.1Q.

Il protocollo IEEE 802.1Q aggiunge al frame Ethernet un tag contenente il VLAN ID (VID o VLAN Identifier) corrispondente alla VLAN interessata al trasferimento. Dato che le trame Ethernet hanno una dimensione minima di 64 byte e una dimensione massima di 1518 byte, l'aggiunta del tag VLAN porta la dimensione della trama Ethernet a 1522 byte. Pertanto gli switch che vogliono trattare le trame di questa dimensione devono essere necessariamente compatibili 802.1Q.



Lo switch compatibile 802.1Q che riceve questo frame deve essere in grado di interpretare i 2 byte del VLAN TAG¹¹ come l'identificatore della VLAN di appartenenza, mentre il resto del frame dovrà essere interpretato come un normale frame, ad esempio 802.3. Qualsiasi switch che riceve il frame deve leggere l'identificatore e decidere come comportarsi, ma prima che il frame esca dall'ultimo switch e venga consegnato al destinatario, l'etichetta deve essere rimossa così da ricostruire il formato originale.

Occorre sottolineare che non è necessario cambiare le schede di rete degli host poiché questi nuovi campi sono utilizzati solo dagli switch compatibili 802.1Q e non dalle schede di rete degli host. Appena la trama entra nello switch compatibile 802.1Q, a cui l'host è collegato, gli viene aggiunto il campo che individua la LAN di appartenenza. Sarà l'ultimo switch di destinazione a rimuovere questo campo e a dare al destinatario il frame nella classica struttura con cui è stato inviato dall'host mittente.

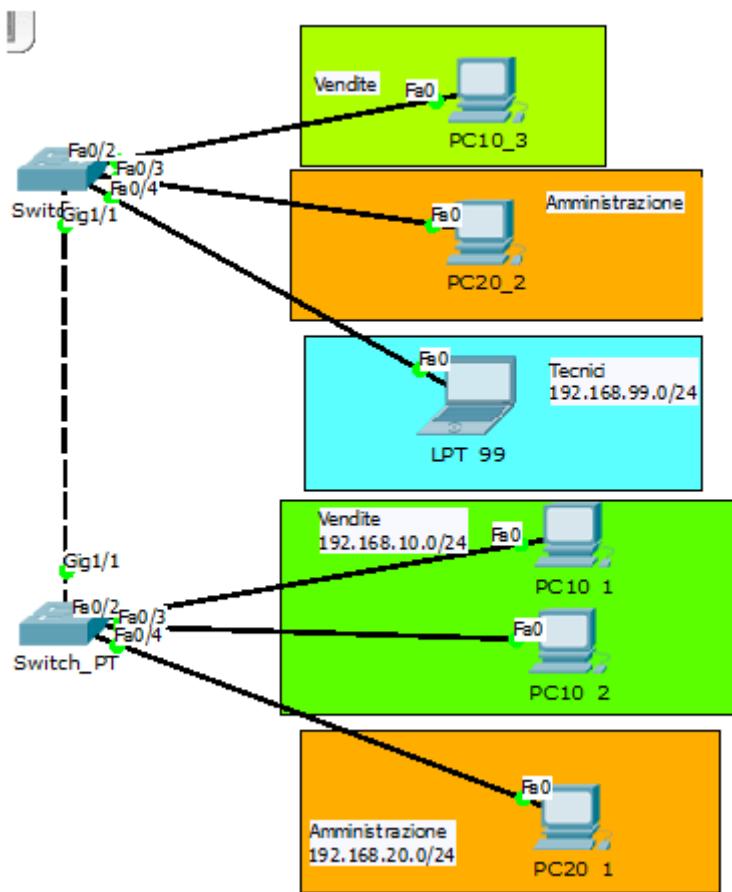
Si osservi però che è buona norma non utilizzare le VLAN per isolare le diverse zone della rete, come ad esempio per ospitare una DMZ, perché il traffico tra le VLAN è *spoofabile*, cioè facilmente falsificabile. È quindi sempre meglio affidarsi a un firewall per isolare le zone tra le quali la sicurezza del traffico è un fattore critico in quanto il tag VLAN 802.1Q potrebbe essere aggiunto volontariamente da qualcuno che desidera saltare tra le VLAN, violando la sicurezza delle reti.

¹¹ I byte aggiunti dal protocollo 802.1Q sono quattro, i primi due vengono usati per indicare con un codice che il formato del frame è cambiato, mentre gli ultimi due rappresentano il VLAN TAG.

3 - Laboratorio: configurazione porte trunk con Packet Tracer

Si provi a creare lo scenario mostrato nell'immagine seguente, verificando la corretta connettività fra i dispositivi dopo aver creato tutte le VLAN e definito il link di collegamento tra i due switch come trunk. Analizzare con attenzione le informazioni del PDU di ingresso e di uscita dallo switch, come mostrato nel video.

[Video della Prof.ssa Sophia Danesino.](#)



Configurazione di un link trunk

Per configurare un link come trunk è necessario configurare le porte dello switch a cui è collegato come segue:

```
Switch(config)#interface gigabitEthernet 1/1
Switch(config-if)#switchport mode trunk
Switch(config-if)#switchport trunk allowed vlan 10-20
Switch(config-if)#exit
```

Le due porte che si trovano ai lati di un trunk sono assegnate a una VLAN nativa.

Per rimuovere una VLAN (ad esempio la 10) da un link trunk si dovrà usare il seguente comando:

```
Switch(config)#interface gigabitEthernet 1/1
Switch(config-if)#switchport trunk allowed vlan remove 10
Switch(config-if)#exit
```

Some useful videos - VLAN Trunking

[Virtual LANs \(VLANS\)](#)

Comprehension Activity

4 - VLAN - Configuration of a virtual switch interface

Using Packet Tracer create the scenario shown in [VLANS and Trunks for Beginners - Part 2](#) video. Talk with your classmate about new concepts and new commands you will have found in the video, such as **how to remotely manage a switch giving an IP address to a VLAN interface**, and add all of them in your personal CLI manual. In order to be coherent with names, you can call the VLAN 50 using the name **data** instead of **student**, or vice versa, or you have only to remember that the speaker uses the label **data** for the VLAN named **student**.

5 - VLAN - Virtual switch interface protection, switch remote configuration and trunking principles

Using Packet Tracer create the scenario shown in [VLANS and Trunks for Beginners - Part 3](#) video. Pay attention to the following elements that you will see in the video:

1. the speaker made a mistake configuring the IP address of the PC 192.168.1.101; do not make the same error, but think about it and try to understand which host was pinged;
2. the speaker was able to create a new VLAN without using the command `vlan VID`. How did he do this?

Talk with your classmate about new concepts and new commands you will have found in the video and add all of them in your personal CLI manual.

6 - VLAN - VLAN trunking configuration

Using Packet Tracer create the scenario shown in [VLANS and Trunks for Beginners - Part 4](#) video. Talk with your classmate about new concepts and new commands you will have found in the video, such as **how to allow trunk mode to a VLAN group using CLI** or **how to show trunks interfaces**, and add all of them in your personal CLI manual.

Approfondimenti

[VLANs and Trunks for Beginners - Part 5](#) native VLAN

[VLANs and Trunks for Beginners - Part 6 VOIP](#)

[VLANs and Trunks for Beginners - Part 7 VOIP](#)

[VLANs and Trunks for Beginners - Part 8 DTP](#)

Routing inter-VLAN

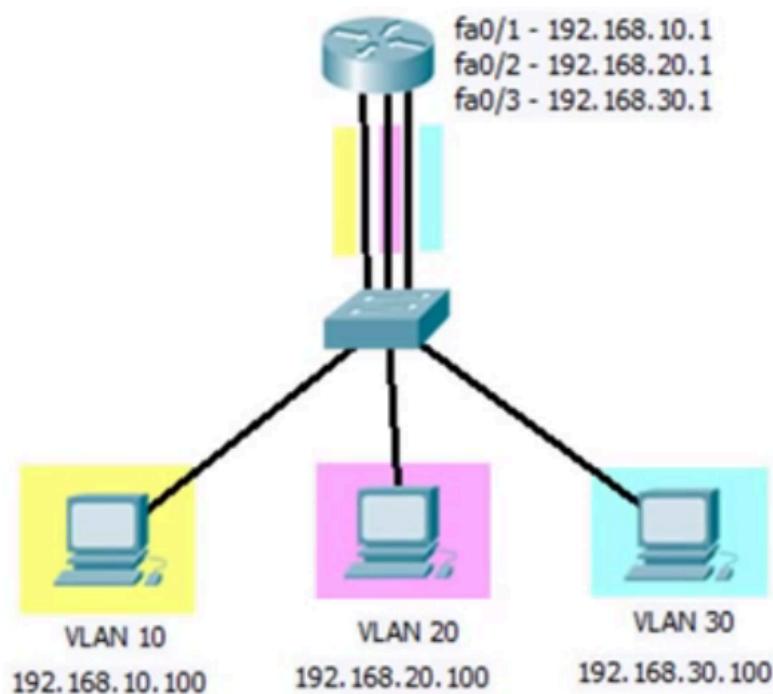
In un ambiente VLAN si esegue lo switching dei frame solo tra porte che appartengono ad uno stesso dominio di broadcast, cioè alla stessa VLAN. Le VLAN eseguono le operazioni di partizionamento della rete e di separazione del traffico a livello 2, quindi **le comunicazioni inter-VLAN non possono avvenire senza un dispositivo di livello 3**, ad esempio un router o uno switch di livello 3. Esaminiamo tre soluzioni possibili.

Router con più collegamenti fisici

Il **routing inter-VLAN tradizionale** avviene utilizzando **un router che dispone di tante interfacce quante sono le VLAN che deve collegare**.

Il router dovrà essere connesso ad uno degli switch della LAN e dovrà avere tante interfacce fisiche collegate allo switch quante sono le VLAN che devono comunicare fra loro. Ogni interfaccia fisica del router sarà associata ad una VLAN e dovrà quindi avere un indirizzo IP appartenente alla VLAN.

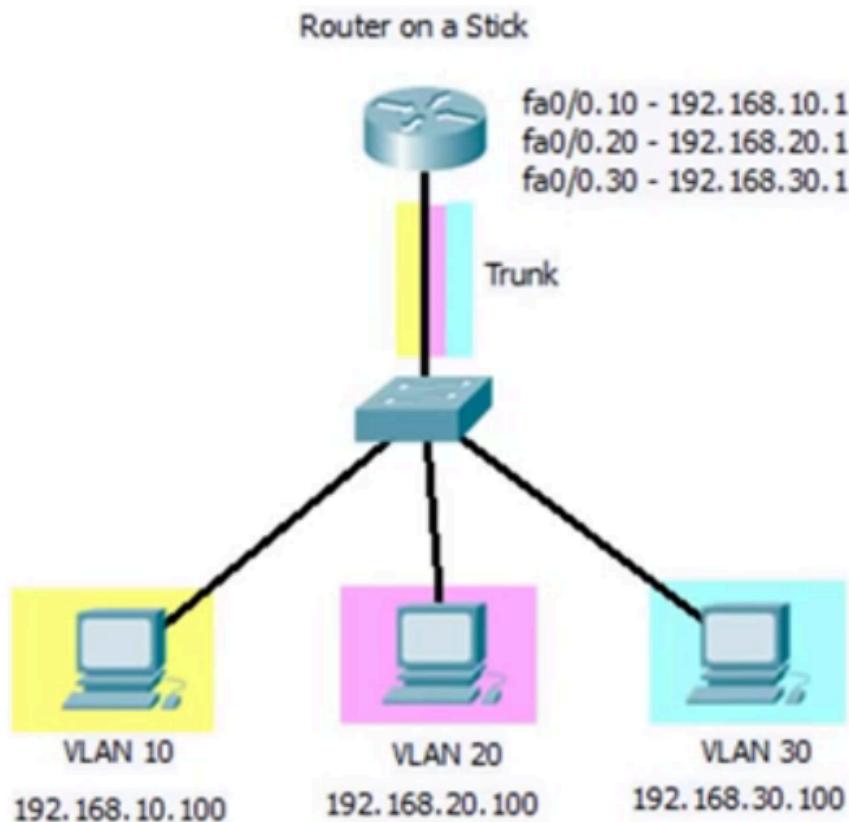
Ad esempio supponiamo di voler far comunicare la VLAN 10 e la VLAN 30:



Se il PC 192.168.10.100 che si trova sulla VLAN10 volesse comunicare con il PC 192.168.30.100 sulla VLAN30, dovrà farlo attraverso il router collegato allo switch. Il messaggio verrà inviato allo switch, di qui verrà inviato all'interfaccia fa0/1 del router e il router lo invierà nuovamente allo switch attraverso l'interfaccia fa0/3, che si occuperà di effettuare la consegna finale. Chiaramente questa soluzione risulta estremamente dispendiosa visto il numero di interfacce coinvolte nel router.

Router-on-a-stick

Una configurazione più interessante è nota con l'espressione di **router-on-a-stick**.



Una sola interfaccia del router è configurata in modo da operare con trunk ed è connessa ad una porta di uno switch anch'essa configurata in trunk mode.

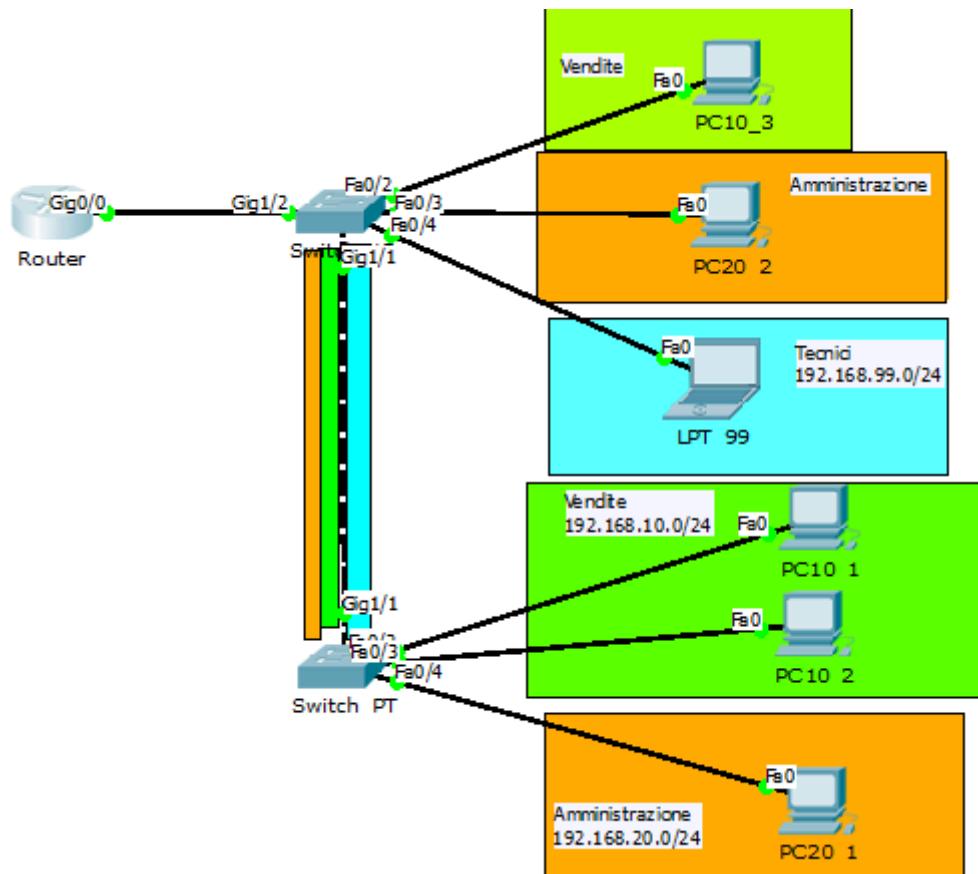
Il router può ricevere pacchetti da una VLAN ed eseguire il forwarding verso un'altra VLAN. Per far questo è necessario **suddividere l'interfaccia fisica del router in più interfacce logiche e indirizzabili**, una per ciascuna VLAN, e **abilitare il trunking sulla connessione fisica**.

Le **interfacce logiche** che derivano da questa impostazione **prendono il nome di sottointerfacce**. Il router effettua l'inter-VLAN routing accettando il traffico con tag proveniente dallo switch ed effettuando il routing tra le VLAN utilizzando le sottointerfacce. Il router infine inoltra il traffico con tag pari alla VLAN di destinazione usando sempre la stessa interfaccia fisica.

In mancanza di questa suddivisione diventa necessario dedicare a ciascuna VLAN una distinta interfaccia fisica.

7 - Laboratorio: inter-VLAN routing con un router-on-a-stick

Svolgere l'esercizio proposto nel [Video della Prof.ssa Sophia Danesino](#). Di seguito sono elencate tutte le configurazioni dei dispositivi. Nel vostro esercizio completate la configurazione anche per la VLAN 99.



Host

Nome host	IP	Gateway	VLAN	Interfaccia switch a cui è collegato
PC101	192.168.10.1/24	192.168.10.254/24	10	SWPT Fa0/2
PC102	192.168.10.2/24	192.168.10.254/24	10	SWPT Fa0/3
PC103	192.168.10.3/24	192.168.10.254/24	10	SW1P Fa0/2
PC201	192.168.20.1/24	192.168.20.254/24	20	SWPT Fa0/4
PC202	192.168.20.2/24	192.168.20.254/24	20	SW1T Fa0/2
LPT99	192.168.99.1/24	192.168.99.254/24	99	SW1T Fa0/4

Router

IP	Sottointerfaccia	VLAN	Interfaccia switch a cui è collegato
192.168.10.254/24	Gig0/0.10	10	SW1T Gig1/2
192.168.20.254/24	Gig0/0.20	20	SW1T Gig1/2

Switch

Switch	VLAN 10	VLAN 20	VLAN 99
SWPT	Fa0/2, Fa0/3	Fa0/4	
SW1T	Fa0/2	Fa0/3	Fa0/4

1. Abilitare il trunking sull'interfaccia dello switch rivolta verso il router

```
Switch(config)#interface Gig1/2
Switch(config-if)#switch port mode trunk
Switch(config-if)#exit
```

2. Abilitare l'interfaccia del router verso lo switch

```
Router(config)#interface Gig0/0
Router(config-if)#no shutdown
```

3. Per ogni sottointerfaccia logica: definire l'incapsulamento VLAN e assegnare un indirizzo IP all'interfaccia¹²

```
Router(config-if)#interface Gig0/0.10
Router(config-subif)#encapsulation dot1Q 10
Router(config-subif)#ip address 192.168.10.254 255.255.255.0
Router(config-subif)#interface Gig0/0.20
Router(config-subif)#encapsulation dot1Q 20
Router(config-subif)#ip address 192.168.20.254 255.255.255.0
Router(config-subif)#interface Gig0/0.99
Router(config-subif)#encapsulation dot1Q 99
Router(config-subif)#ip address 192.168.99.254 255.255.255.0
Router(config-subif)#exit
Router(config)#
```

Comprehension Activity

8 - VLAN - Router-on-a-stick

Watch the following three videos and using Packet Tracer create the scenario shown about Router-on-a-stick configuration. Talk with your classmate about new concepts and new commands you will have found in the video, and add all of them in your personal CLI manual.

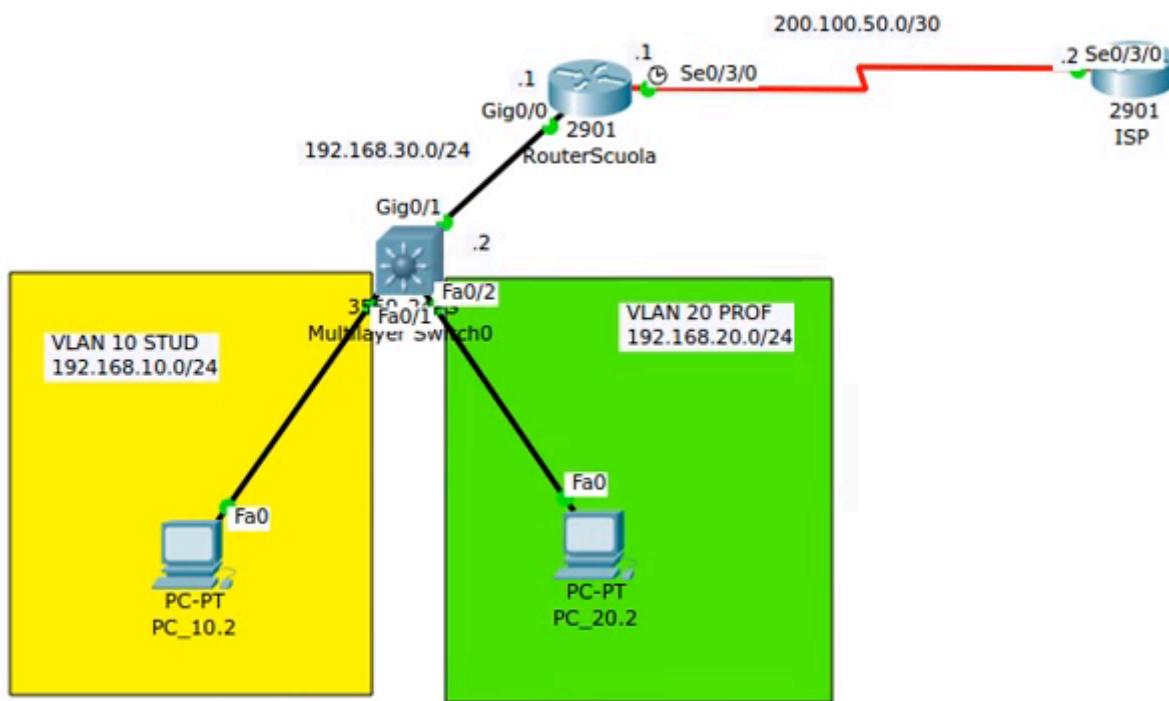
¹² La sigla **dot1Q** specifica un incapsulamento di tipo 802.1q

[Router on a Stick, Inter-VLAN Routing - Part 1](#)[Router on a Stick, Inter-VLAN Routing - Part 2](#)[Router on a Stick, Inter-VLAN Routing - Part 3](#)

Switch Layer 3

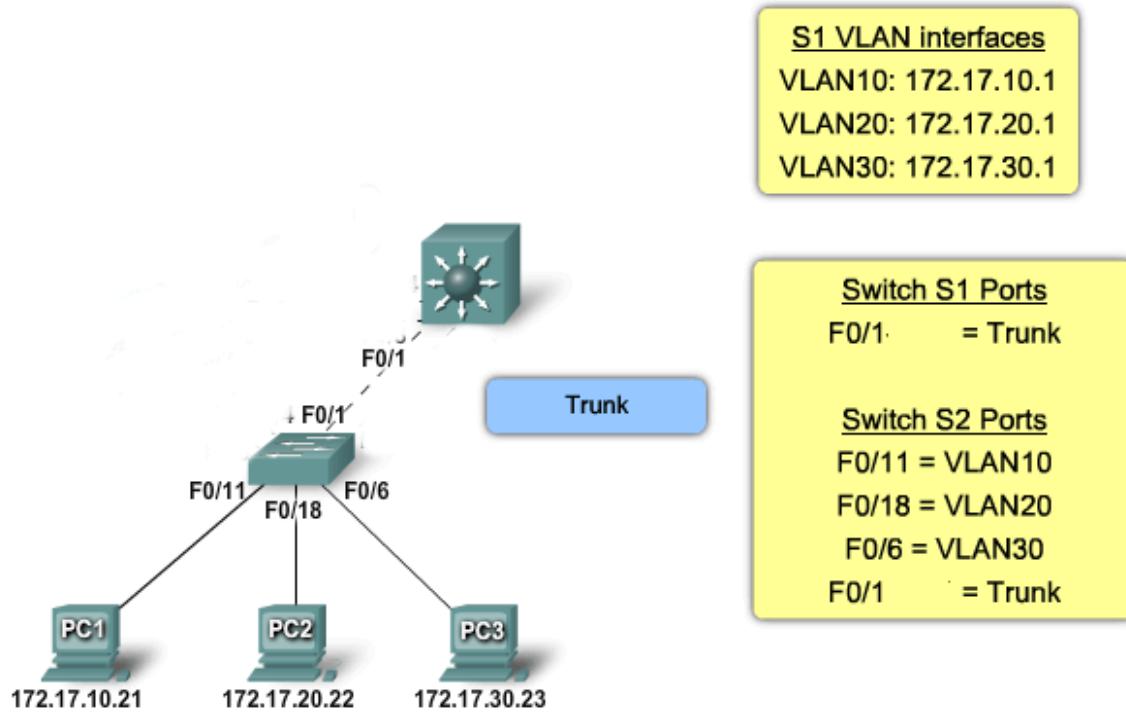
Alcuni **switch** possono effettuare funzioni **Layer 3**, evitando la necessità di utilizzare un router. Questi switch, detti “**Multilayer switch**” sono in grado di effettuare lo **inter-VLAN routing**. L’uso di questi switch permette, ad esempio, di limitare il traffico di rete che si ha sul collegamento nel caso di un router-on-a-stick.

Nel seguente scenario i due PC sono connessi ad uno **switch L3** che, oltre a svolgere operazioni di livello 2, è anche in grado di operare a livello 3. Usando un **multilayer switch** si ha la possibilità di **assegnare alle interfacce un indirizzo IP** creando una **interfaccia virtuale**, utilizzabili **per** effettuare a tutti gli effetti **operazioni di routing**.



Lo scenario seguente invece mostra la combinazione di un collegamento trunk fra i due switch, e l’uso di un multilayer switch che potrà eventualmente essere collegato con un router per permettere una connettività anche con l’esterno.

Nello scenario il multilayer switch svolge di fatto le veci di un router-on-a-stick.



In questo caso PC1 sulla VLAN10 (switch S2 porta F0/11) invia un messaggio a PC3 che appartiene alla VLAN30 (switch S2 porta F0/6) come segue:

- PC1 invia il messaggio unicast a S2;
- lo switch S2 appone il tag 10 al messaggio e lo invia tramite il link trunk allo switch S1;
- S1 rimuove il tag, effettua il routing del traffico ponendo il tag 30 al messaggio e inviandolo nuovamente sul trunk;
- lo switch S2 rimuove il tag e lo invia a PC3 sulla porta F0/6.

Passi per configurare l'inter-VLAN routing con switch L3

Per configurare l'inter-VLAN routing in uno switch L3 bisogna seguire i passi sintetizzati di seguito e che vedremo applicati nei prossimi esercizi.

1. **configurare le VLAN**

```
vlan xx
```

```
name yy
```

2. **associare le porte dello switch alle VLAN in modalità access**

```
switchport mode access
```

```
switchport access vlan xx
```

3. **configurare le virtual interface**

```
interface vlan xx
```

```
ip address a.a.a.a s.s.s.s
```

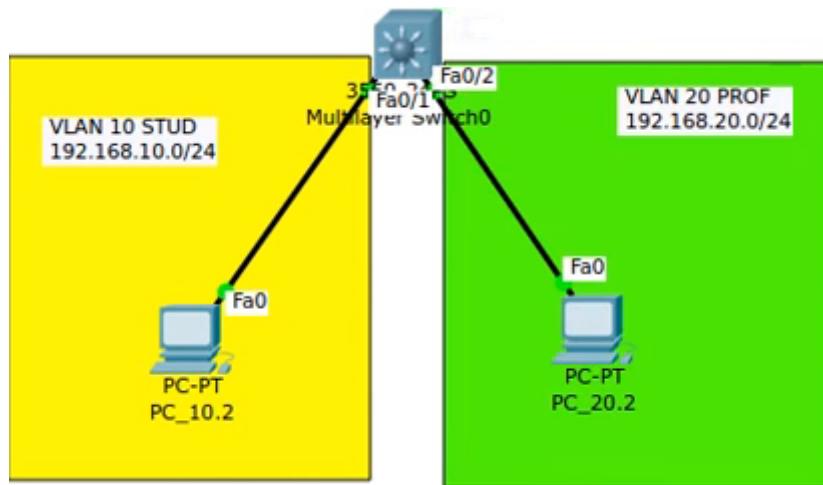
4. **abilitare l'inter-VLAN routing**

```
ip routing
```

9 - Laboratorio: inter-VLAN routing con uno Switch Layer 3

[Video della Prof.ssa Sophia Danesino.](#)

In una rete molto trafficata il routing-on-a-stick può essere inadeguato a causa del traffico presente sul link di collegamento con il router (stick). In questo caso è preferibile una soluzione basata su un **Multi-layer switch**, detto anche **switch layer 3** poiché **offre sia i servizi di livello 2 che di livello 3**. Vediamo come si configura l'inter-VLAN routing su uno switch Layer 3 nello scenario seguente.



Sullo switch 3560 sono state create due VLAN:

```
Switch>enable
Switch# configure terminal
Switch(config)# vlan 10
Switch(config-vlan)# name STUD
Switch(config-vlan)# exit
Switch(config)# vlan 20
Switch(config-vlan)#name PROF
Switch(config-vlan)# exit
```

Successivamente sono state **configurate in modalità access** le due porte **Fa0/1 e Fa0/2** dello switch L3 come **appartenenti** rispettivamente alla **VLAN 10 e alla VLAN 20**:

```
Switch(config)# interface FastEthernet0/1
Switch(config-if)# switch port mode access
Switch(config-if)# switch port access vlan 10
Switch(config-if)# exit
Switch(config)# interface FastEthernet0/2
Switch(config-if)# switch port mode access
Switch(config-if)# switchport access vlan 20
Switch(config-if)# exit
```

Per consentire l'**inter-VLAN routing** è necessario che **la rete 192.168.10.0/24 e la 192.168.20.0/24 vedano lo switch come il loro gateway**. A tal scopo è necessario **creare due interfacce virtuali nello switch**, una per ogni VLAN, e **assegnare loro l'IP del gateway**. Le interfacce virtuali sono chiamate **SVI**, acronimo di **Switch Virtual Interface**. **L'IP delle SVI non è assegnato ad**

un'interfaccia fisica, ma bensì ad una interfaccia virtuale il cui numero deve corrispondere al VLAN tag 802.1Q:

```
Switch(config)# interface vlan 10
Switch(config-if)# ip address 192.168.10.1 255.255.255.0
Switch(config-if)# exit
```

```
Switch(config)# interface vlan 20
Switch(config-if)# ip address 192.168.20.1 255.255.255.0
Switch(config-if)# exit
```

Infine è necessario **abilitare esplicitamente l'inter-VLAN routing** con il comando seguente:

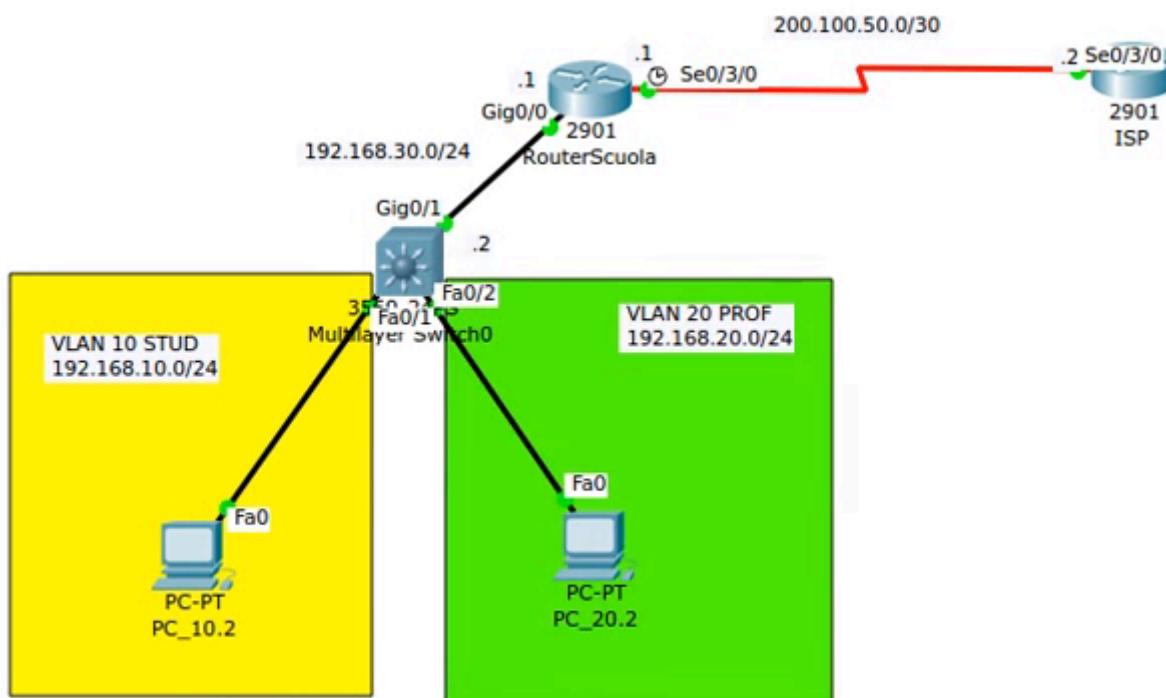
```
Switch(config)# ip routing
```

Si analizzi la **running configuration** dello switch L3 usando il comando

```
Switch# show running-config
```

sarà possibile verificare la presenza di tutte le configurazioni impostate finora.

Immaginiamo ora che gli host presenti nelle due VLAN abbiano la necessità di uscire dalla propria rete locale, come nello schema seguente:



In questa situazione **sul multilayer switch** andrà configurata la **porta Gig0/1** con un **indirizzo IP** appartenente alla rete **192.168.30.0/24** e andranno **configurate le rotte per consentire l'instradamento verso il router** (è possibile, ad esempio, prevedere un gateway of last resort verso il router RouterScuola). Tutti gli switch ([Catalyst](#)) gestiscono 2 tipi di interfacce L3:

- **switched**: operano a livello 2 in base ai MAC (smistano frame Ethernet o 802.1Q);

- **routed**: operano a livello 3 in base agli IP (consentono il routing dei pacchetti).

Nel caso in esame la **porta Gig0/1** dovrà essere **configurata** come **routed**, con i seguenti comandi:

```
Switch(config)# interface GigabitEthernet0/1
Switch(config-if)# no switchport
Switch(config-if)# ip address 192.168.30.2 255.255.255.0
Switch(config-if)# no shutdown
Switch(config-if)# exit
```

Infine devono essere impostate nei due router, RouterScuola e ISP, tutte le rotte per raggiungere tutte le reti non direttamente connesse.

Comprehension Activity

10 - VLAN - Multilayer Switching

Using Packet Tracer create the scenario shown in [Multilayer Switching in Packet Tracer 6.1](#) video. Talk with your classmate about new concepts and new commands you will have found in the video, such as **how the speaker sets the gateway of last resort on the router** and **the default gateway on the switch**, and add all of them in your personal CLI manual.

VTP (VLAN Trunking Protocol)

All'interno di un **dominio (gruppo logico di utenti e risorse)** ciascuna VLAN deve essere configurata manualmente su ogni switch. Se l'organizzazione cresce e vengono aggiunti nuovi switch, ognuno di essi dovrà essere configurato con le informazioni di tutte le VLAN. Una sola configurazione errata provoca problemi di connessione tra VLAN.

Il **protocollo VTP (VLAN Trunking Protocol)**, proprietario della Cisco, consente di ridurre la complessità della gestione di un dominio offrendo la possibilità di **inserire, cancellare, modificare le VLAN su un unico switch e automaticamente distribuisce tali informazioni agli altri switch appartenenti al dominio**. Il protocollo **lavora a livello 2**.

Un **dominio VTP è un dominio di amministrazione costituito da uno o più switch interconnessi**. I messaggi VTP garantiscono che le informazioni VLAN risultino sempre sincronizzate all'interno del dominio VTP. Quando si aggiunge/modifica/cancella una VLAN, le nuove informazioni si propagano agli altri switch del dominio.

Le configurazioni vengono fatte su uno switch server VTP. Il protocollo VTP invia messaggi solo alle porte trunk.

VTP funziona in tre modalità (che vanno impostate su uno switch):

1. **server** (default): in questa modalità è possibile creare, modificare o cancellare delle VLAN e qualsiasi modifica si propagherà immediatamente a tutti gli switch del medesimo dominio. Queste informazioni vengono memorizzate nella NVRAM (Non Volatile Random-Access Memory) dello switch server in un file denominato `vlan.dat`;
2. **client**: in questa modalità non è possibile creare, modificare o cancellare le VLAN, ma comunque si esegue il forwarding dei messaggi VTP e le informazioni ricevute da uno switch VTP server sono memorizzate nella NVRAM, nel file `vlan.dat`, sincronizzandole con gli altri client e server VTP;
3. **transparent**: questo tipo di switch effettua il forwarding dei messaggi VTP originati da uno switch VTP server, pur non usandoli per configurare le proprie VLAN. Uno switch transparent può avere una propria configurazione di VLAN, ma non considererà la configurazione ricevuta da uno switch VTP server, si occuperà solo di inviare ad altri switch client i messaggi di advertisement.

I **messaggi VTP** vengono **invia**ti agli switch del dominio VTP **ogni 5 minuti o** immediatamente **dopo la modifica di qualche configurazione VLAN**. In ogni **messaggio VTP** è presente un **numero di revisione della configurazione**. Uno switch che riceve un messaggio VTP aggiornerà la propria configurazione VLAN solo in presenza di un numero di revisione superiore al proprio.

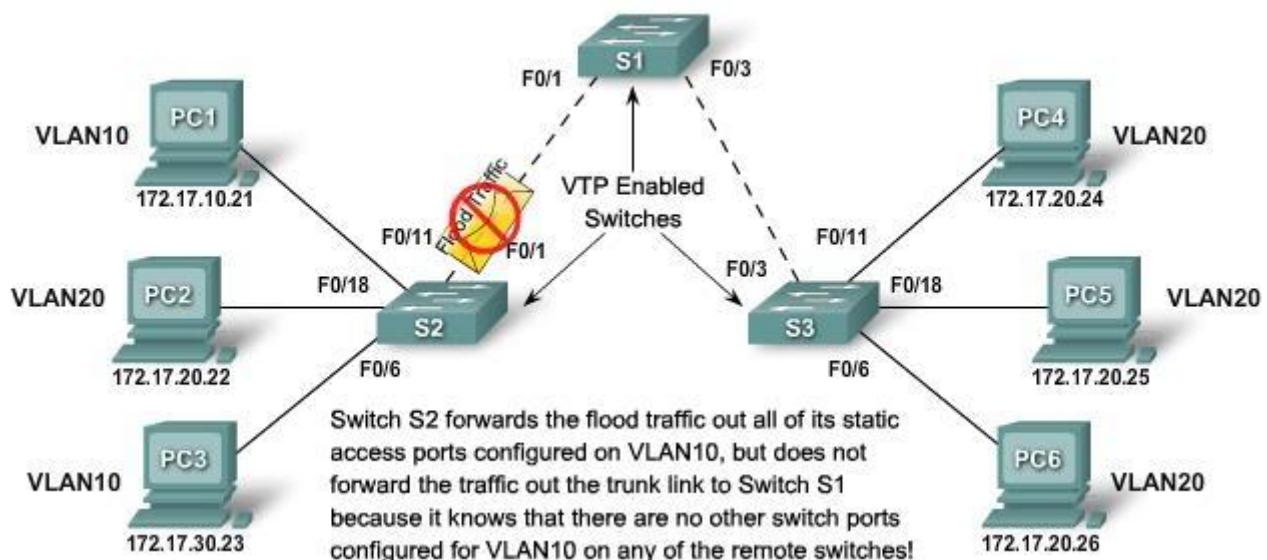
Di seguito sono elencati i comandi VTP:

```
Switch#configure terminal
```

```
Switch(config) #vtp domain nome-dominio
Switch(config) #vtp mode server/client/transparent
Switch(config) #vtp password pwd
Switch(config) #vtp pruning
Switch(config) #exit
Switch#show vtp status
```

È da sottolineare l'importanza della password nel VTP server per evitare che sia possibile modificare/cancellare le VLAN senza autorizzazione su un intero dominio. Se, ad esempio, una persona cancellasse una VLAN su uno switch VTP server, tale modifica avrebbe effetto su tutto il dominio. Di conseguenza tutte le porte associate a tale VLAN verrebbero messe automaticamente down e la macchina risulterebbe scollegata dalla rete.

Infine il concetto **VTP pruning** (potatura) stabilisce quando una connessione trunk evita il traffico di flooding superfluo verso gli switch non interessati ad una particolare revisione. Nella figura seguente, ad esempio, gli switch S1 e S3 non sono interessati al traffico destinato alla VLAN 10, ma ricevono ugualmente messaggi indirizzati a questa VLAN poiché sono configurati in trunk mode. Se il pruning è abilitato S2 saprà che in quell'area della rete non ci sono host appartenenti alla VLAN 10 e non inoltrerà sul trunk i frame della VLAN 10.



Questo è anche possibile configurando direttamente l'interfaccia trunk con il comando:

```
SwitchS2(config)#interface Fa0/11
SwitchS2(config-if)#switchport trunk allowed vlan 20
```

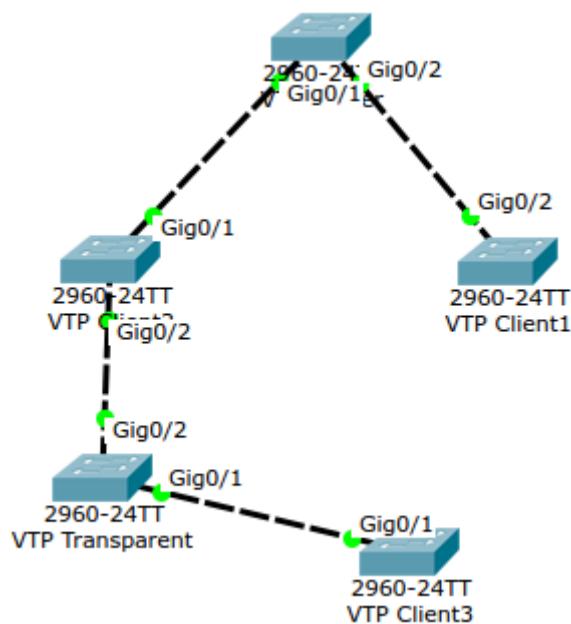
In questo modo lo switch S2 viene abilitato al trunk solo per i messaggi della VLAN 20 e i frame destinati alla 10 non vi transiteranno. Questa configurazione però è statica e quindi poco flessibile. Se, ad esempio PC1 venisse spostato da S2 a S3 bisognerebbe consentire su S2 anche l'inoltro della 10. Se, invece, è stato abilitato il pruning questo avverrà in automatico.

11 - Laboratorio VTP: Switch server, client e transparent

[Video della Prof.ssa Sophia Danesino.](#)

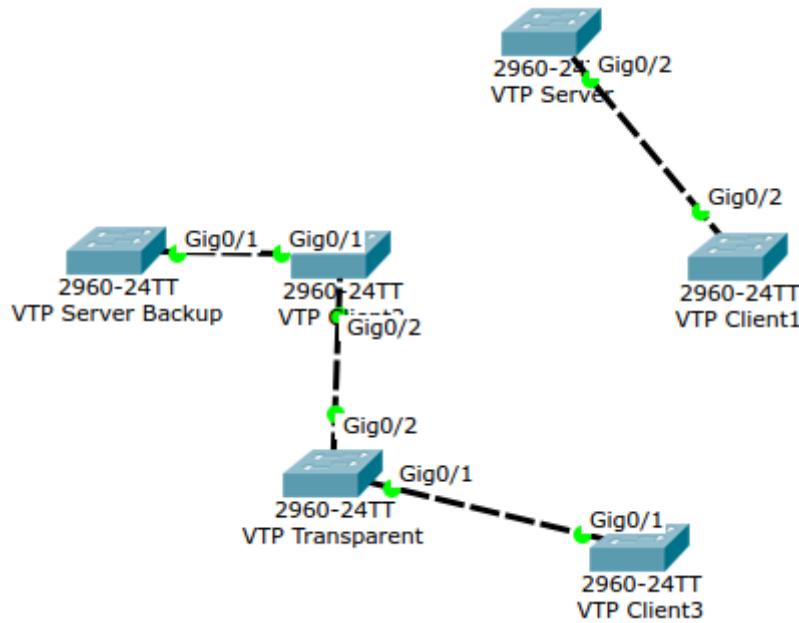
Si costruisca lo scenario mostrato nel video.

Durante la prima parte del video lo scenario sarà quello mostrato nell'immagine seguente.



In questo scenario si dovranno creare le VLAN, come mostrato nei paragrafi precedenti, e si dovrà procedere con la **configurazione del dominio**, la **configurazione dello switch server** che propagherà le informazioni agli switch client e l'**impostazione della modalità client a tutti gli switch del dominio che non rivestono il ruolo di switch server**. Si analizzi con attenzione come avviene la propagazione agli switch client delle informazioni sulle VLAN configurate nello switch server. Inoltre si osservi il cambio del numero di revisione della configurazione quando viene aggiunta una VLAN nello switch server.

Nella seconda parte del video lo scenario viene modificato come mostrato dall'immagine seguente.



In questo scenario è stato aggiunto uno switch server di backup nel quale non sono state configurate le VLAN presenti nello switch server originario. Grazie al numero di revisione della configurazione presente negli switch client, superiore a quella presente nello switch server di backup, sarà quest'ultimo ad aggiornare la sua configurazione, prendendola direttamente da quella dei client. Quindi **la propagazione delle configurazioni può avvenire in due sensi in quanto si rispetta sempre il numero di revisione di quella più recente.**

Per **controllare lo stato del protocollo VTP** in uno switch, si utilizzi il comando:

```
Switch#show vtp status
```

Per **creare un dominio**, assegnandogli un nome, si utilizzi il comando:

```
Switch(config)#vtp domain nome
```

Per **modificare la modalità di funzionamento** in uno switch (server/client/transparent) si utilizzi il comando:

```
Switch(config)#vtp mode server/client/transparent
```

scegliendo solo una delle tre opzioni tra `server`, `client` o `transparent`.

Comprehension Activity

12 - How VTP works

Using Packet Tracer create the scenario shown in the following videos:

1. [How to use VTP -VLAN Trunking Protocol? - Part 1](#)
2. [How to use VTP -VLAN Trunking Protocol? - Part 2](#)
3. [How to use VTP -VLAN Trunking Protocol? - Part 3](#)

Talk with your classmate about new concepts and new commands you will have found in the video, and add all of them in your personal CLI manual.

13 - VLAN - Activity Overview

1. [VLANs and Trunks](#)
2. [Switch & VLAN Packet Tracer Challenge](#)
3. [Routing and Switching Essentials Practice Final](#) (ripasso generale)

14 - Subnetting e VLAN

Progettare con Packet Tracer, la simulazione di una piccola **rete dell'università di POLISTUDIO**.

La rete prevede:

- due laboratori per gli studenti:
 - LABINFO con 5 PC al 1° piano e un PC TOTEM (rilevazione presenze) collegato a UFFDOCENTI;
 - LABTLC con 5 PC al piano terra e un PC TOTEM (rilevazione presenze) collegato a UFFDOCENTI;
- due uffici:
 - UFFDOCENTI con 3 PC al piano terra e 2 PC al 1° piano;
 - UFFSEGRETERIA 2 PC al 1° piano e 3 PC al 2° piano.

Le reti dei laboratori e delle segreterie prevedono i seguenti indirizzi:

NOME	Primo IP disponibile Host	GATEWAY	VLAN
LABINFO	192.168.10.1/24	192.168.10.254/24	10
LABTLC	192.168.20.1/24	192.168.20.254/24	20
UFFDOCENTI	192.168.50.1/24	192.168.50.254/24	50
UFFSEGRETERIA	192.168.60.1/24	192.168.60.254/24	60

Progettare e realizzare con un simulatore di rete le seguenti soluzioni:

1. utilizzare le VLAN usando router-on-a-stick;
2. utilizzare le VLAN usando switch L3;
3. utilizzare solo switch L2 e un router per fare subnetting usando DHCP.

Si realizzi anche una relazione scritta dove si confronteranno le diverse soluzioni spiegandone i vantaggi e gli svantaggi, sia a livello implementativo, sia a livello di costi (cercare su Internet informazioni al riguardo)

Sicurezza informatica e crittografia



"On the Internet, nobody knows you're a dog."

Credit: Peter Steiner. © The New Yorker magazine

By Source, Fair use, <https://en.wikipedia.org/w/index.php?curid=13627120>

Prefazione

I seguenti appunti sono in parte basati sul lavoro svolto dalla Prof.ssa Sophia Danesino, mentore, amica, appassionata di conoscenza e insegnante fuori dal comune. Sul suo [canale YouTube](#) è possibile trovare i video su molti degli argomenti trattati.

Per l'arricchimento e la stesura finale degli appunti sono risultate fondamentali le lezioni del [Prof. Steven Gordon](#) a cui va la mia profonda gratitudine per la pubblicazione delle sue lectures sul suo [canale YouTube](#).

Sicurezza in rete

La sicurezza delle reti

Libro vol.3 - La sicurezza nei sistemi informativi

STUDIARE: *L. Lo Russo, E. Bianchi, Sistemi e Reti - Nuova Edizione OPENSCHOOL - Per l'articolazione INFORMATICA degli Istituti Tecnici settore Tecnologico, vol. 3, ed. Hoepli, 2017.*

- UdA4 - La sicurezza delle reti
 - Lezione 1 - La sicurezza dei sistemi informativi.
 - La sicurezza dei dati pp.166-168
 - Sicurezza di un sistema informatico pp.169-170
 - Valutazione dei rischi pp.170-172
 - Principali tipologie di minacce pp.172-173 - [SYN attack](#)
 - Sicurezza nei sistemi informativi distribuiti pp.173-175

Crittografia simmetrica

Tecniche crittografiche per la protezione dei dati

Libro vol.3 - La crittografia simmetrica

STUDIARE: L. Lo Russo, E. Bianchi, Sistemi e Reti - Nuova Edizione OPENSCHOOL - Per l'articolazione INFORMATICA degli Istituti Tecnici settore Tecnologico, vol. 3, ed. Hoepli, 2017.

- UdA3 - Tecniche crittografiche per la protezione dei dati
 - Lezione 1 - La crittografia simmetrica
 - La sicurezza nelle reti pp.96-97
 - Crittografia pp.97-98
 - Crittoanalisi pp.98-99
 - Cifrari e chiavi pp.99-100
 - Il cifrario DES pp.100-102
- CLIL Listening - [Introduction to Cryptography](#) del Prof. Steven Gordon (*il video è utile per comprendere meglio quanto spiegato dal libro, con l'aggiunta di alcuni approfondimenti, ed è parte integrante del materiale di studio*)
- UdA3 - Tecniche crittografiche per la protezione dei dati
 - Lezione 1 - La crittografia simmetrica
 - 3-DES pp.102-103 (*è necessario comprenderne la ragione dell'uso dell'algoritmo e spiegarlo in termini generali, non serve il dettaglio*)
 - IDEA p.103 (*è necessario comprenderne la ragione dell'uso dell'algoritmo e spiegarlo in termini generali, non serve il dettaglio*)
 - AES pp.103-105 (*è necessario comprenderne la ragione dell'uso dell'algoritmo e spiegarlo in termini generali, non serve il dettaglio*)
 - Limiti degli algoritmi simmetrici p.106
- CLIL Listening - [Symmetric Key Encryption and Brute Force Attacks](#) del Prof. Steven Gordon (*il video è utile per comprendere meglio quanto spiegato dal libro e presenta degli interessanti esempi di brute force attack su DES, 3-DES e AES, ed è parte integrante del materiale di studio*)

Esercizi sulla crittografia simmetrica

Nei seguenti esercizi si farà riferimento alla seguente notazione:

- $C = E(K_{XY}, P)$ con la quale si identifica un messaggio cifrato C scambiato tra due utenti X e Y , ottenuto applicando sul testo in chiaro P un algoritmo crittografico simmetrico $E()$ che usa la chiave simmetrica K_{XY} condivisa tra i due utenti X e Y .
- $P = D(K_{XY}, C)$ con la quale si identifica un messaggio in chiaro P scambiato tra due utenti X e Y , ottenuto applicando al testo cifrato C un algoritmo de-crittografico $D()$ che usa la chiave simmetrica K_{XY} condivisa tra i due utenti X e Y .

Esercizi

1. Due utenti X ed Y decidono di comunicare in modo confidenziale utilizzando la crittografia simmetrica e si scambiano il seguente messaggio cifrato:
 $C = E(K_{XY}, P)$
Supporre che l'utente Z intercetti il testo cifrato C e tenti di eseguire la seguente operazione:
 $D(K_{XZ}, C)$
Si spieghi cosa può assumere l'utente Z da questo tentativo in base a quanto appreso finora sulla crittografia simmetrica.
2. Si supponga che l'utente X invii il messaggio cifrato $C = (K_{XY}, P)$. L'utente Y riceverà il messaggio cifrato C_r . Si spieghi cosa potrà desumere l'utente Y nei due seguenti casi:
 - $D(K_{XY}, C_r) = P$
 - $D(K_{XY}, C_r) \neq P$
3. Con le conoscenze acquisite finora, spiegare se la crittografia simmetrica può essere usata per l'autenticazione.

Crittografia asimmetrica

Tecniche crittografiche per la protezione dei dati

Libro vol.3 - La crittografia asimmetrica

STUDIARE: L. Lo Russo, E. Bianchi, Sistemi e Reti - Nuova Edizione OPENSCHOOL - Per l'articolazione INFORMATICA degli Istituti Tecnici settore Tecnologico, vol. 3, ed. Hoepli, 2017.

- Uda3 - Tecniche crittografiche per la protezione dei dati
 - Lezione 2 - La crittografia asimmetrica
 - Generalità pp.108,110-113 (*saltare l'esempio di pp.108-109*)
- CLIL Listening - [Assumptions of Encryption and Authentication](#) (*dal minuto 48:43*) del [Prof. Steven Gordon](#) (*il video è utile perché fornisce i concetti generali della crittografia asimmetrica, ed è parte integrante del materiale di studio*)
- Uda3 - Tecniche crittografiche per la protezione dei dati
 - Lezione 2 - La crittografia asimmetrica
 - RSA pp.113-117 (*l'esempio a pp.114-115 del calcolo del MCD usando il metodo di Euclide può essere tralasciato*)
 - **FACOLTATIVO:** chi volesse approfondire il funzionamento dell'algoritmo RSA e i principi matematici che lo fanno funzionare potrà farlo vedendo i seguenti video del [Prof. Steven Gordon](#):
 - [Basics of Primes and Modular Arithmetic](#)
 - [Eulers Theorem, Fermats Theorem and Discrete Logarithms](#)
 - [Public Key Crypto with RSA](#)
 - [RSA Summary and Examples](#)
- Studiare sugli [appunti del docente RSA](#) pp.89-96
- Uda3 - Tecniche crittografiche per la protezione dei dati
 - Lezione 2 - La crittografia asimmetrica
 - Crittografia ibrida pp.117-119
- CLIL Listening - [Public Key Crypto and Digital Signatures](#) del [Prof. Steven Gordon](#) (*il video è utile perché fornisce molteplici esempi di uso della crittografia simmetrica e asimmetrica, un esempio di uso della crittografia asimmetrica per lo scambio delle chiavi simmetriche, chiarisce il funzionamento delle funzioni hash e introduce la firma digitale; inoltre il video è parte integrante del materiale di studio*)

RSA

In crittografia la sigla **RSA** indica un algoritmo di crittografia asimmetrica, inventato nel 1977 da Ronald Rivest, Adi Shamir e Leonard Adleman utilizzabile per cifrare o firmare informazioni.



Il sistema di crittografia si basa sull'esistenza di due chiavi distinte, che vengono usate per cifrare e decifrare. Se la prima chiave viene usata per la cifratura, la seconda deve necessariamente essere utilizzata per la decifratura e viceversa. La questione fondamentale è che **nonostante le due chiavi siano fra loro dipendenti, non è computazionalmente possibile risalire dall'una all'altra**, in modo che se anche si è a conoscenza di una delle due chiavi, non si possa risalire all'altra, garantendo in questo modo l'integrità della crittografia.

RSA è basato sulla **elevata complessità computazionale della fattorizzazione in numeri primi**¹³, oltre che del calcolo **del logaritmo discreto**¹⁴ e della determinazione **della funzione di Eulero**¹⁵; quando si lavora con numeri molto grandi, dell'ordine di centinaia di cifre, la risoluzione di uno qualsiasi di questi problemi diventa improponibile.

¹³ Il **teorema fondamentale dell'aritmetica** afferma che: *ogni numero naturale maggiore di 1 o è un numero primo o si può esprimere come prodotto di numeri primi. Tale rappresentazione è unica, se si prescinde dall'ordine in cui compaiono i fattori.*

¹⁴ Il **logaritmo discreto** è l'inverso per la determinazione dell'esponente di una potenza, modulo n.

¹⁵ La **funzione di Eulero $\phi(n)$** , detta **toziente**, è una funzione che, per ogni intero positivo n, è **definita come il numero degli interi compresi tra 1 e n e minori di n che sono coprimi con n**.

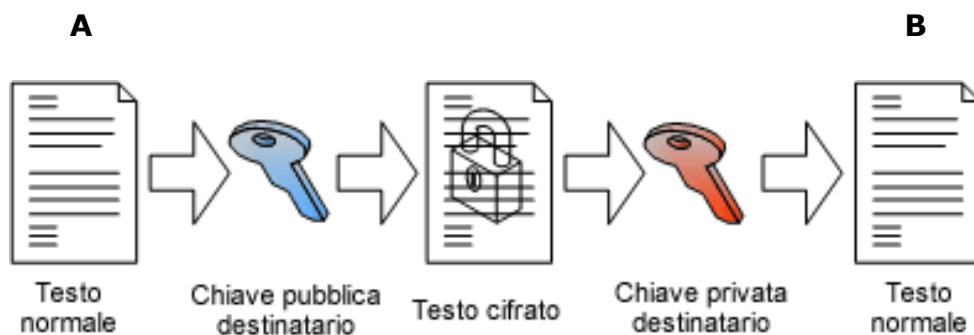
In breve dati due numeri primi **p** e **q** molto grandi, è **facile calcolare**

$$n = p * q$$

ma è **computazionalmente troppo lungo trovare** i due fattori **p** e **q**, noto **n**.

Concetto di base di RSA

Per semplificare il funzionamento immaginiamo che **A** debba spedire un messaggio segreto a **B**.



1. **B** determina due chiavi, una privata e una pubblica. Le **chiavi** sono delle **coppie di numeri**.
2. **B** invia la propria chiave pubblica ad **A**. Chiunque può vedere questa chiave.
3. **A** usa questa chiave per cifrare il messaggio (**garanzia di confidenzialità**).
4. **A** manda il messaggio cifrato a **B**, chiunque può vederlo, ma non decifrarlo.
5. **B** riceve il messaggio e, utilizzando la chiave privata che solo lui conosce, lo decifra.

Immaginiamo che la **chiave pubblica** sia **PU** = (e, n) costituita dalla coppia di numeri **e** e **n**, e la **chiave privata** sia **PR** = (d, n) costituita dalla coppia di numeri **d** e **n**¹⁶. Per **cifrare** un messaggio **m** che deve essere minore di **n** (**m < n**) si utilizzerà la **chiave pubblica** **PU** = (e, n) e l'algoritmo

$$\text{Cifratura: } c = m^e \bmod n$$

mentre per **decifrare** un messaggio cifrato **c** si utilizzerà la **chiave privata** **PR** = (d, n) e l'algoritmo

$$\text{Decifrazione: } m = c^d \bmod n$$

Si noti che l'unico elemento veramente segreto nelle due chiavi **PU** e **PR** è il valore **d**.

¹⁶ Più avanti sarà spiegato come determinare i valori **e** e **d**.

Esempio di criptazione e decrittazione usando RSA

Il messaggio viene rappresentato come un valore intero, e ciò è fattibile perché un messaggio è una sequenza di bit di cui si può trovare l'equivalente decimale.

Messaggio: ' m ' = $(10010001)_2 = (145)_{10}$

quindi crittografare un messaggio equivale a cifrare il suo corrispondente intero decimale.

Vediamo un esempio di crittografia e decrittografia usando **RSA**, e supponiamo che si debba crittografare il messaggio $m = 7$, usando i valori $n = 55$ ed $e = 3$ che costituiranno la chiave pubblica $PU = (e, n) = (3, 55)$

$$c = m^e \bmod n = 7^3 \bmod 55 = 13$$

Per decrittografare il messaggio cifrato **13** usiamo la chiave privata $PR = (d, n) = (27, 55)$

$$m = c^d \bmod n = 13^{27} \bmod 55 = 7$$

Quindi utilizzando la chiave pubblica e quella privata che sono create usando una correlazione matematica, è possibile cifrare e decifrare un messaggio.

La generazione delle chiavi

1. Si scelgono a caso due numeri primi diversi p e q abbastanza **grandi** da garantire la sicurezza dell'algoritmo (sono consigliati almeno 4096 bit); **p e q devono rimanere privati;**
2. si calcola il loro prodotto $n = p * q^{17}$, chiamato modulo (dato che tutta l'aritmetica seguente è modulo n);
3. si calcola il prodotto $z = (p-1) * (q-1)^{18}$
4. si determina la **chiave pubblica PU = (e, n)** scegliendo un numero e tale per cui :
 - a. $1 < e < n$
 - b. e deve essere **coprimo con z**¹⁹
5. si determina la **chiave privata PR = (d, n)** scegliendo un numero d tale per cui $(e*d) \bmod z = 1^{20}$

Da questi valori si determina la **chiave pubblica PU = (e, n)** e la **chiave privata PR = (d, n)**. Dopo la determinazione delle due chiavi i due numeri primi p e q possono anche essere distrutti, ma spesso vengono mantenuti insieme alla chiave privata.

La forza dell'algoritmo sta nel fatto che per calcolare d da e o viceversa, non basta la conoscenza di n , ma serve il numero $z = (p-1) * (q-1)$ e che il suo calcolo richiede tempi molto elevati; infatti fattorizzare in numeri primi, o determinare il toziente di un numero, sono operazioni che richiedono molto tempo se si scelgono dei valori numerici costituiti da molte cifre. Solo conoscendo la fattorizzazione di n è possibile trovare il valore delle chiavi. Infatti conoscendo la fattorizzazione di n è possibile calcolare $z = (p-1) * (q-1)$ e calcolare $d = e^{-1} \bmod z$ usando l'algoritmo esteso di Euclide.

Inoltre la sicurezza di **RSA** si basa sul fatto che la funzione di cifratura $c = m^e \bmod n$ è una funzione "**one-way**" che è computazionalmente difficile da invertire per un nemico che volesse decifrare un messaggio.

¹⁷ La fattorizzazione di n è segreta e solo chi sceglie i due numeri primi, p e q la conosce. Inoltre se i numeri primi p e q vengono scelti molto grandi, diventa molto difficile risalire ad essi conoscendo solo n .

¹⁸ Il valore di $z = (p-1)*(q-1)$ rappresenta la **funzione di Eulero $\phi(n)$** , o **toziente**, che **rappresenta il numero degli interi compresi tra 1 e n e minori di n che sono coprimi con n**.

¹⁹ Due numeri e e z si dicono **coprimi** se non hanno fattori in comune, cioè se $MCD(e, z) = 1$.

²⁰ $(e*d) \bmod z = 1$ implica che $d \equiv e^{-1} \bmod z$. Per il **teorema cinese del resto**, un modo semplice per determinare d è il seguente $d = (z * k + 1)/e$ dove k è un intero positivo.

Esempio di generazione delle chiavi di RSA

Vediamo un esempio di generazione delle due chiavi dell'algoritmo RSA. Per esigenze didattiche verranno presi come numeri primi **p** e **q** due valori piccoli, ma si ricordi che nella realtà i due numeri primi dovranno essere costituiti da un numero molto elevato di cifre per garantire la sicurezza dell'algoritmo.

1. **p = 5, q = 11**
2. **n = p * q = 5 * 11 = 55**
z = (p-1)*(q-1) = 4 * 10 = 40
3. Trovare **1 < e < 55** tale che **MCD(e, 40) = 1**
40 = 5 * 2³
per e = 2 => MCD(2, 40) = 2 => NO
per **e = 3 => MCD(3, 40) = 1 => SI**²¹
4. Trovare **d** tale che **(3 * d) mod 40 = 1**
per d = 2 => (2*3) mod 40 = 6 => NO
per d = 3 => (3*3) mod 40 = 9 => NO
....
per **d = 27 => (3*27) mod 40 = 81 mod 40 = 1 => SI**

Chiave pubblica = (3, 55)

Chiave privata = (27, 55)

La determinazione del valore **d** richiede molto tempo, ma questo valore può essere determinato più velocemente usando il **teorema Cinese del resto** che permette di determinarlo usando la seguente formula

$$d = (z * k + 1)/e$$

con **k intero positivo**

e **d** tale che **(e*d) mod z = 1**

quindi procedendo per passi successivi si ottiene:

- **k = 1 => d = (40 * 1 + 1)/3 => d = 13**, e controllando che il valore vada bene si dovrà avere (e * d) mod z = 1 => (3 * 13) mod 40 = 39 => **NO**
- **k = 2 => d = (40 * 2 + 1)/3 => d = 27**, e controllando che il valore vada bene si dovrà avere (e * d) mod z = 1 => (3 * 27) mod 40 = 1 => **SI**

²¹ Si potrebbe continuare a cercare un qualsiasi altro valore minore di n = 55 che sia coprimo di z = 40.

Esempio sull'uso di RSA

Generazione delle chiavi

- Si scelgano dei due numeri primi diversi: $p = 17$, $q = 11$
- Si moltiplichino i due numeri primi: $n = p * q = 17 * 11 = 187$
- Si calcoli la funzione toziente²² di $n = 187$:

$$z = (p - 1) * (q - 1) = 16 * 10 = 160$$
- Si scelga un valore di e tale che $\text{MCD}(e, z) = 1$ cioè siano coprimi e $1 < e < n$:

$$2^{\text{NO}}, 3^{\text{SÌ}}, 4^{\text{NO}}, 5^{\text{NO}}, 6^{\text{NO}}, 7^{\text{SÌ}}, 8^{\text{NO}}, 9^{\text{SÌ}}, 10^{\text{NO}}, 11^{\text{SÌ}}, \dots$$

Può essere scelto il valore $e = 7$
- Si determini un valore di d tale che $(e * d) \bmod z = 1$ o per il teorema Cinese del resto $d = (z * k + 1)/e$, con k intero positivo:

$$k = 1 \Rightarrow d = (160 * 1 + 1)/7 \Rightarrow d = 23 \Rightarrow (7 * 23) \bmod 160 = 1 \Rightarrow \text{SÌ}$$
- Quindi $\text{PU} = (e, n) \Rightarrow \text{PU} = (7, 187)$, $\text{PR} = (d, n) \Rightarrow \text{PR} = (23, 187)$

I valori che devono assolutamente rimanere **segreti** sono **p**, **q**, **d** e **z**, mentre sono **pubblici** i valori **n**, **e** e gli **algoritmi** usati per criptare e decriptare.

Si supponga di dover effettuare una **comunicazione criptata** tra Alice e Bob:

Alice	Bob
<i>Generazione delle chiavi</i>	
$\text{PU}_A = (7, 187)$ $\text{PR}_A = (23, 187)$ <i>Alice invia a Bob la sua chiave pubblica</i> $\text{PU}_B = (5, 299)$	$\text{PU}_B = (5, 299)$ $\text{PR}_B = (53, 299)$ <i>Bob invia ad Alice la sua chiave pubblica</i> $\text{PU}_A = (7, 187)$
<i>Alice vuole inviare il messaggio m = 15 criptato a Bob. La dimensione del messaggio m deve essere più piccola di n = 299.</i>	
$c = E(\text{PU}_B, m)$ $c = m^e \bmod n$ $c = 15^5 \bmod 299$ $c = 214$	
<i>Bob riceve il messaggio criptato c = 214</i>	
	$c = 214$ $m' = D(\text{PR}_B, c)$ $m' = c^d \bmod n$ $m' = 214^{53} \bmod 299$ $m' = 15$
<i>Bob ha decriptato correttamente il messaggio m = 15</i>	

²² $\phi(n) = \phi(p) * \phi(q)$, dato che p e q sono primi $\Rightarrow \phi(n) = (p - 1) * (q - 1)$. Per brevità nei libri di testo non universitari viene semplicemente indicato $z = (p - 1) * (q - 1)$.

Se Alice volesse inviare a Bob un messaggio, ad esempio *ciao*, si dovrebbero convertire i singoli caratteri in codice ASCII:

c	01100011
i	01101001
a	01100001
o	01101111

Il **messaggio in chiaro** sarà: 01100011011010010110000101101111

La sequenza di bit viene suddivisa in blocchi di g bit in modo tale che g sia il più grande numero tale che $2^g < n$, cioè $2^g < 299$, quindi $g = 8$ ($2^8 = 256$).

Dividiamo il messaggio in blocchi 8 bit: 01100011 01101001 01100001 01101111

Cifratura del messaggio

Blocco	In decimale	Blocco cifrato in decimale $P_U_B=(5,299)$	Blocco cifrato in binario
01100011	99	$99^5 \bmod 299 = 86$	01010110
01101001	105	$105^5 \bmod 299 = 27$	00011011
01100001	97	$97^5 \bmod 299 = 158$	10011110
01101111	111	$111^5 \bmod 299 = 11$	00001011

Il **messaggio cifrato** è: 01010110000110111001111000001011

Decifratura del messaggio

Blocco cifrato in binario	In decimale	Blocco decifrato in decimale $P_R_B=(53,299)$	Blocco decifrato in binario
01010110	86	$86^{53} \bmod 299 = 99$	01100011
00011011	27	$27^{53} \bmod 299 = 105$	01101001
10011110	158	$158^{53} \bmod 299 = 97$	01100001
00001011	11	$11^{53} \bmod 299 = 111$	01101111

Il messaggio decifrato da Bob corrisponde esattamente a quello inviato da Alice.

Esercizi sull'uso di RSA

1. Generare la chiave pubblica e segreta dati i numeri primi $p = 13$ e $q = 23$ e criptare e decriptare un messaggio a vostra scelta.
2. Date le chiavi pubbliche e private $PU = (7, 33)$ e $PR = (3, 33)$, si cifri e si decifri il messaggio $m = 15$
3. Generare la chiave pubblica e segreta dati i numeri primi $p = 7$ e $q = 17$ e criptare e decriptare un messaggio a vostra scelta.
4. Date le chiavi pubbliche e private $PU = (5, 65)$ e $PR = (29, 65)$, si cifri e si decifri il messaggio $m = 7$
5. Generare la chiave pubblica e segreta dati i numeri primi $p = 61$ e $q = 53$ e criptare e decriptare un messaggio a vostra scelta.
6. Date le chiavi pubbliche e private $PU = (17, 3233)$ e $PR = (2753, 3233)$, si cifri e si decifri il messaggio $m = 123$
7. Generare la chiave pubblica e segreta dati i numeri primi $p = 47$ e $q = 71$ e criptare e decriptare un messaggio a vostra scelta.
8. Date le chiavi pubbliche e private $PU = (79, 3337)$ e $PR = (1019, 3337)$, si cifri e si decifri il messaggio $m = 688$

Esercizi sulla crittografia asimmetrica

Nei seguenti esercizi si farà riferimento alla seguente notazione:

- $C = E(PU_x, P)$ con la quale si identifica un messaggio cifrato C ottenuto applicando sul testo in chiaro P un algoritmo crittografico a chiave asimmetrica $E()$ che usa la chiave pubblica PU_x dell'utente X .
- $C = E(PR_x, P)$ con la quale si identifica un messaggio cifrato C ottenuto applicando sul testo in chiaro P un algoritmo crittografico a chiave asimmetrica $E()$ che usa la chiave privata PR_x dell'utente X .

Esercizi

1. Si supponga che l'utente Y riceva il messaggio cifrato $C = E(PU_x, P)$. L'utente Y può decriptare C per ottenere P ? Si motivi la risposta.
2. Si supponga che l'utente X riceva il messaggio cifrato $C = E(PU_x, P)$. L'utente X può decriptare C per ottenere P ? Si motivi la risposta.
3. Si supponga che l'utente Y riceva il messaggio cifrato $C = E(PR_x, P)$. L'utente Y è in grado di conoscere il testo in chiaro P ? E se il messaggio crittografato C fosse ricevuto dall'utente Z , potrebbe ottenere il testo in chiaro P ? Cos'altro può essere dedotto dal ricevimento di C ? Motivare tutte le risposte.
4. Si supponga che l'utente Y riceva il seguente messaggio crittografato:
 $C = E(PU_y, E(PR_x, P))$
Si spieghi se Y potrà decrittografare C per ottenere P e cos'altro potrà dedurre Y dal ricevimento di questo messaggio.
Si supponga inoltre che un utente malevolo Z intercetti il messaggio crittografato C , si spieghi se l'utente Z è in grado di ottenere il messaggio in chiaro P .
5. Con le conoscenze acquisite finora, spiegare se la crittografia asimmetrica può essere usata per l'autenticazione.

Autenticazione e affidabilità

Tecniche crittografiche per la protezione dei dati

Libro vol.3 - Certificati e firma digitale

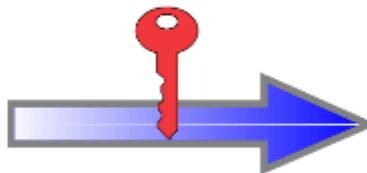
STUDIARE: L. Lo Russo, E. Bianchi, Sistemi e Reti - Nuova Edizione OPENSCHOOL - Per l'articolazione INFORMATICA degli Istituti Tecnici settore Tecnologico, vol. 3, ed. Hoepli, 2017.

- UdA3 - Tecniche crittografiche per la protezione dei dati
 - Lezione 3 - Certificati e firma digitale
 - I sistemi di autenticazione pp.120-122
 - Firme digitali²³ pp.123-125
- CLIL Listening - [Public Key Crypto and Digital Signatures](#) (*dal minuto 48:32*) del [Prof. Steven Gordon](#) (*il video è utile perché chiarisce il funzionamento delle funzioni hash e introduce la firma digitale, ed è parte integrante del materiale di studio*)
- UdA3 - Tecniche crittografiche per la protezione dei dati
 - Lezione 3 - Certificati e firma digitale
 - I certificati digitali pp.125-128
 - Riferimenti normativi pp.128-129

Documento con identità
e chiave pubblica di
Mario Rossi



Chiave Privata
del
soggetto certificatore



Certificato di
Mario Rossi

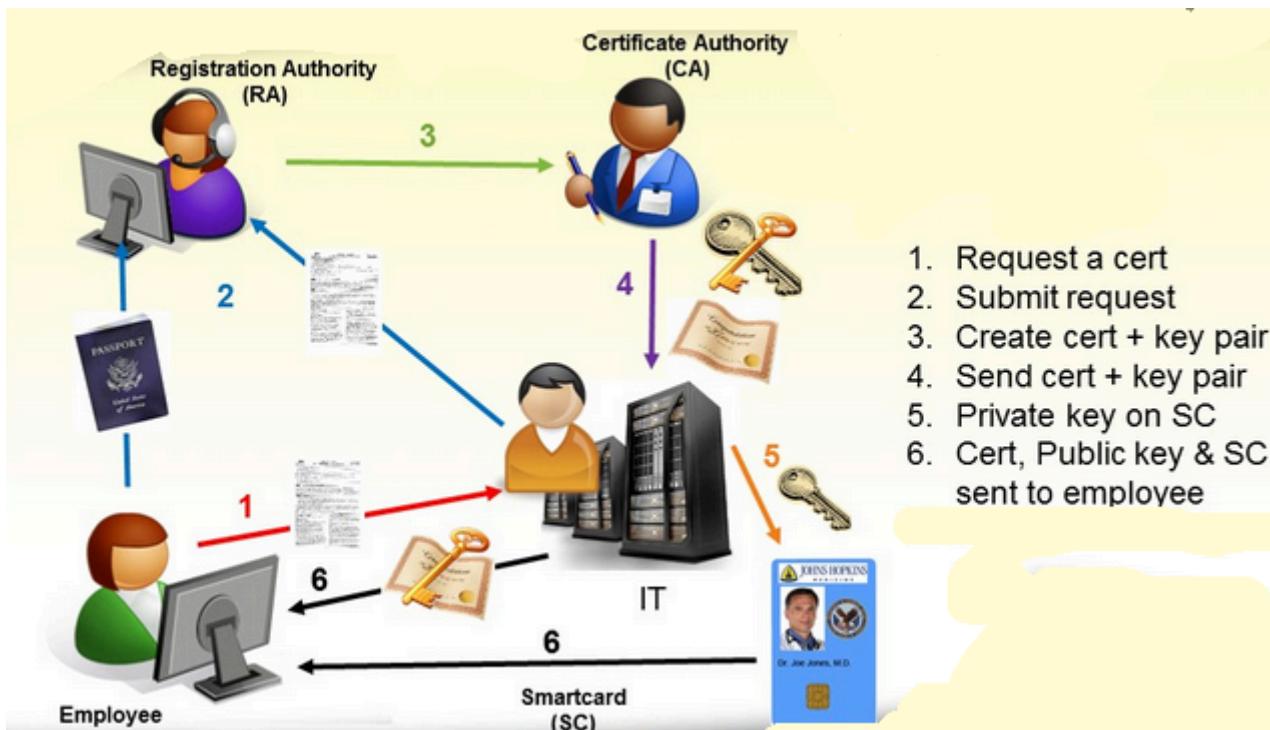
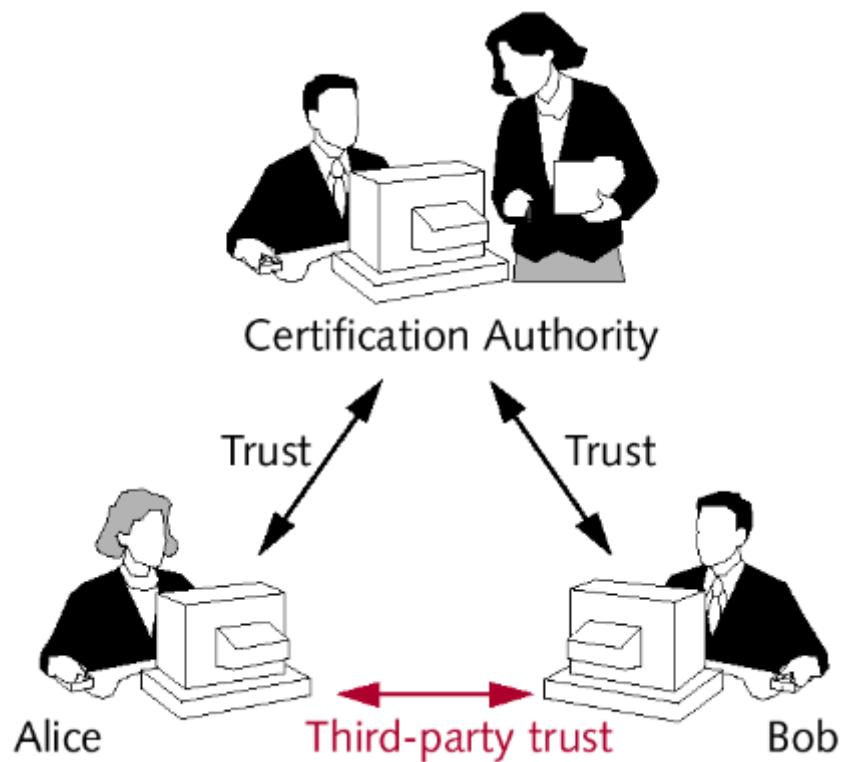
Nome: Mario
Cognome: Rossi
Indirizzo: via

Chiave Pubblica
di
Mario Rossi

Firma del soggetto certificatore

Documento firmato dal
soggetto certificatore

²³ Per approfondimenti sulle funzioni hash, MD5 e SHA si veda [Funzioni di hash](#) - [Wikiversità](#)



1. Request a cert
2. Submit request
3. Create cert + key pair
4. Send cert + key pair
5. Private key on SC
6. Cert, Public key & SC sent to employee

Protocolli per la sicurezza della rete

Prefazione

I seguenti appunti sono in parte basati sulle lavoro svolto dalla Prof.ssa Sophia Danesino, mentore, amica, appassionata di conoscenza e insegnante fuori dal comune. Sul suo [canale YouTube](#) è possibile trovare i video su molti degli argomenti trattati.

Per l'arricchimento e la stesura finale degli appunti sono risultate fondamentali le lezioni del [Prof. Steven Gordon](#) a cui va la mia profonda gratitudine per la pubblicazione delle sue lectures sul suo [canale YouTube](#).

I protocolli sicuri

Le tecniche crittografiche presentate nel capitolo precedente sono applicate ai protocolli di comunicazione al fine di garantire la sicurezza nelle applicazioni di rete, e ciò è indispensabile per attuare le contromisure necessarie a garantire la riservatezza, l'integrità e autenticità come, per esempio, nelle transazioni commerciali o finanziarie.

La sicurezza tramite crittografia è applicata ai protocolli del modello TCP/IP, a partire dal livello di rete fino a quello di applicazione, a seconda delle esigenze.

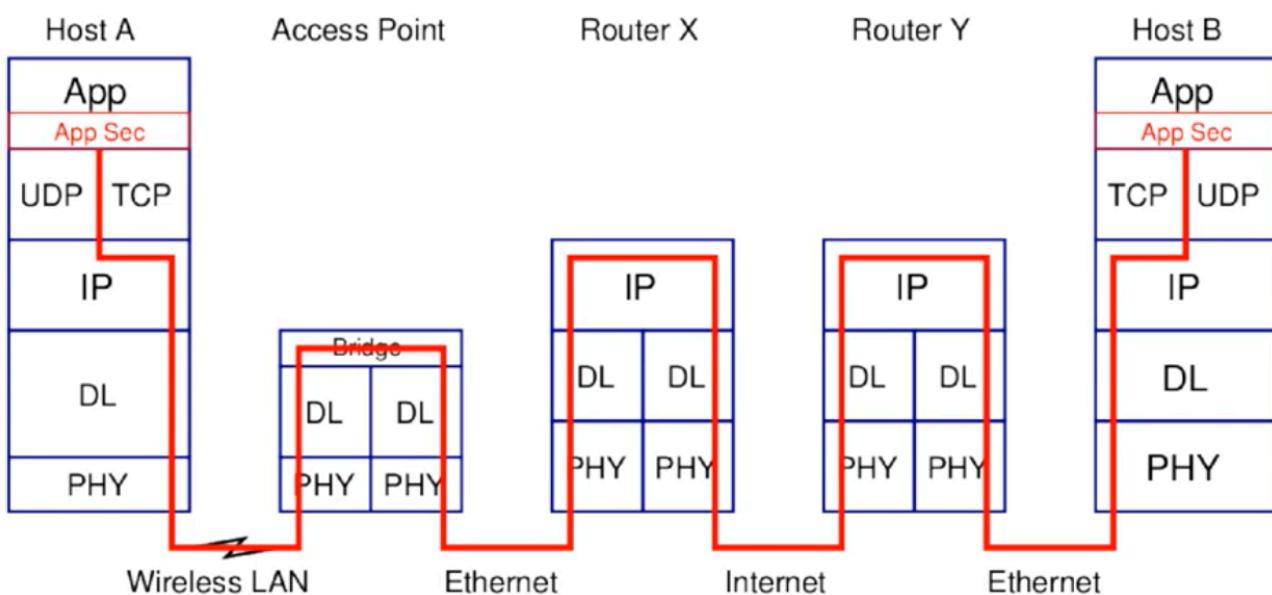
L'applicazione della sicurezza a un protocollo di un determinato livello della pila TCP/IP protegge anche i protocolli di livello superiore, e quindi se viene applicata la crittografia a livello di rete, risulteranno sicuri anche i protocolli di livello trasporto e applicazione. Se invece la sicurezza venisse applicata a livello di trasporto, sarà garantita la sicurezza al livello applicazione, ma non al livello di rete.

La tabella seguente evidenzia alcuni dei protocolli di sicurezza utilizzabili nei diversi livelli della pila protocollare TCP/IP.

Livello applicazione	PGP/GPG, Kerberos, Radius, SSH, S/MIME
Livello trasporto	Transport Layer Security (TLS)
Livello rete	IP Security (IPSec)

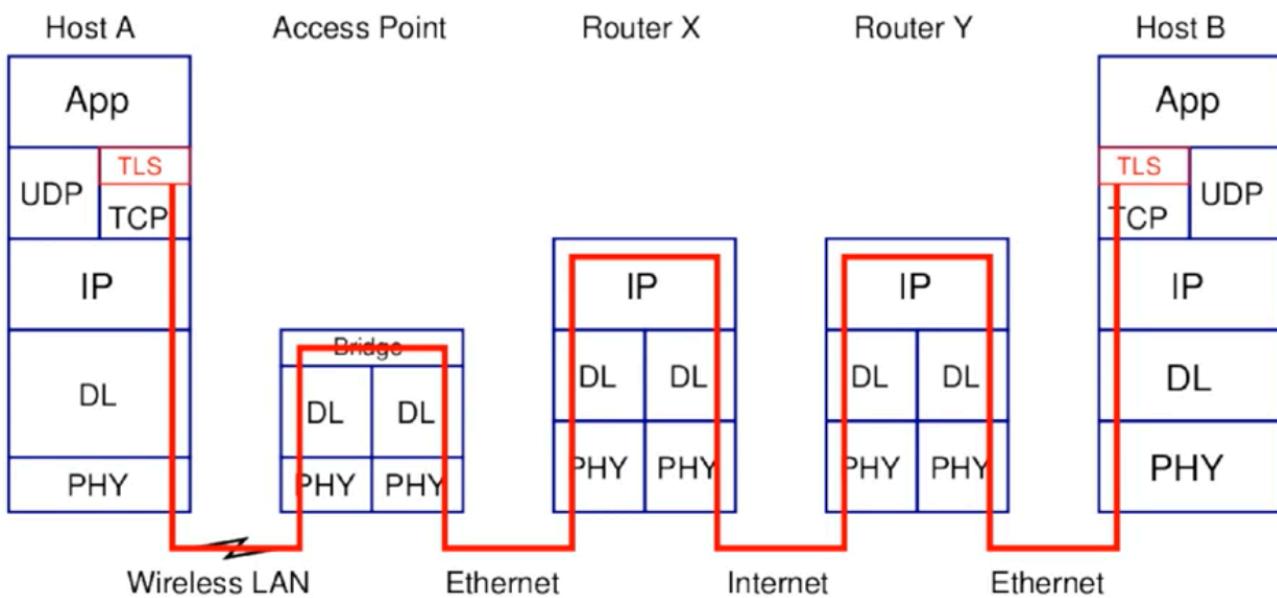
I protocolli per la gestione della sicurezza a livello applicazione, come **PGP/GPG, Kerberos, Radius, SSH, S/MIME**, ecc., sono protocolli che vengono associati ad altre applicazioni per renderle sicure. La sicurezza viene garantita solo al livello applicazione, i livelli sottostanti non sono gestiti in modo sicuro.

Application Level Security: Application-Specific



Il protocollo **TLS** garantisce la sicurezza al livello applicazione e si appoggia al protocollo TCP, non funzionando di solito con UDP. Un'applicazione che usa TLS non dovrà quindi occuparsi della sicurezza che verrà fornita da questo protocollo. Il protocollo TLS è comunemente usato con il protocollo HTTP (HTTPs), ma può essere usato anche con altre applicazioni come la posta elettronica o l'instant messaging.

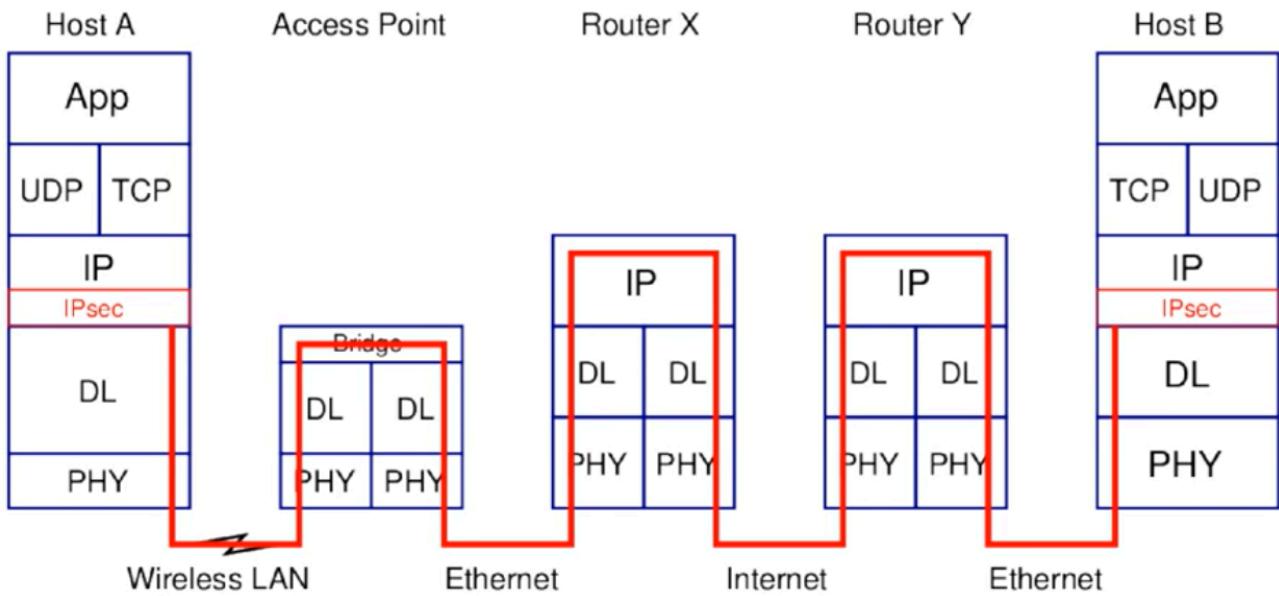
Transport Level Security: TLS/SSL



Il protocollo **IPSec** può offrire una protezione end-to-end, ma può anche essere limitato a porzioni della rete più limitati, e può proteggere gli indirizzi IP del mittente e del destinatario quando usato in modalità **tunnel mode** tra host e router, o tra router e router, oltre a proteggere comunque tutti gli header dei PDU dei livelli superiori a quello di rete e il contenuto del messaggio quando usato in **transport mode** tra end system. Il protocollo IPSec può essere adottato sia per applicazioni basate su TCP, sia per applicazioni basate su UDP.

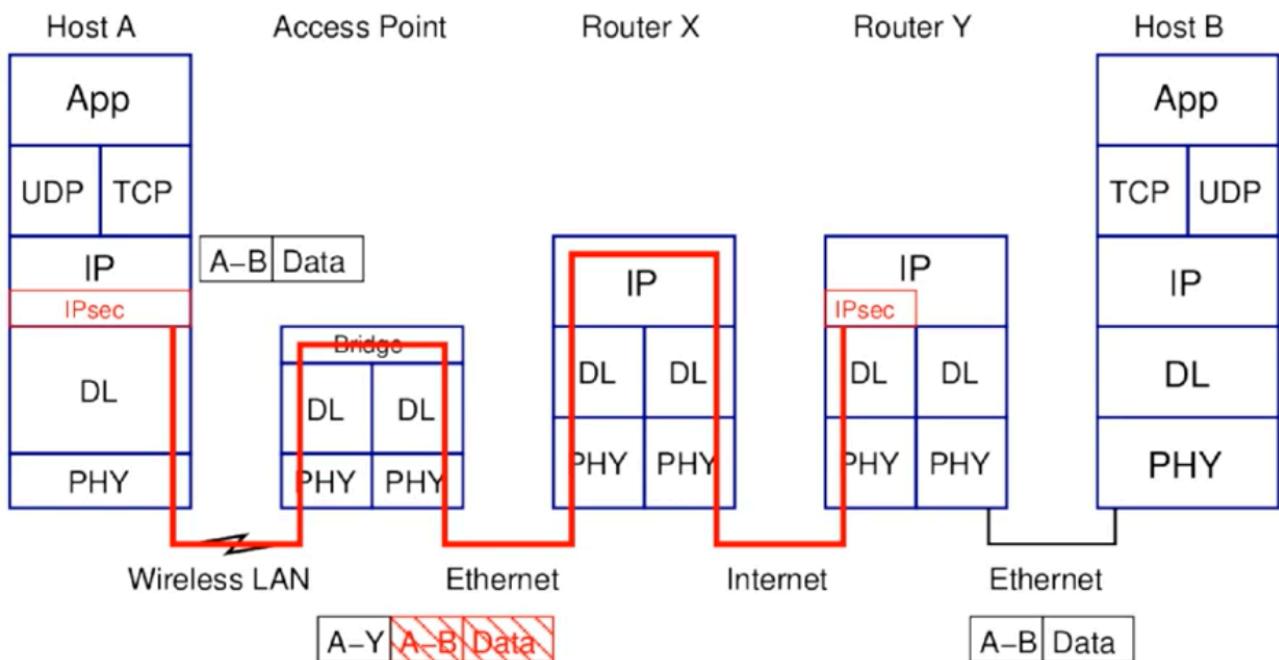
IPSec usato in transport mode.

Network Level Security: IPsec End-to-End

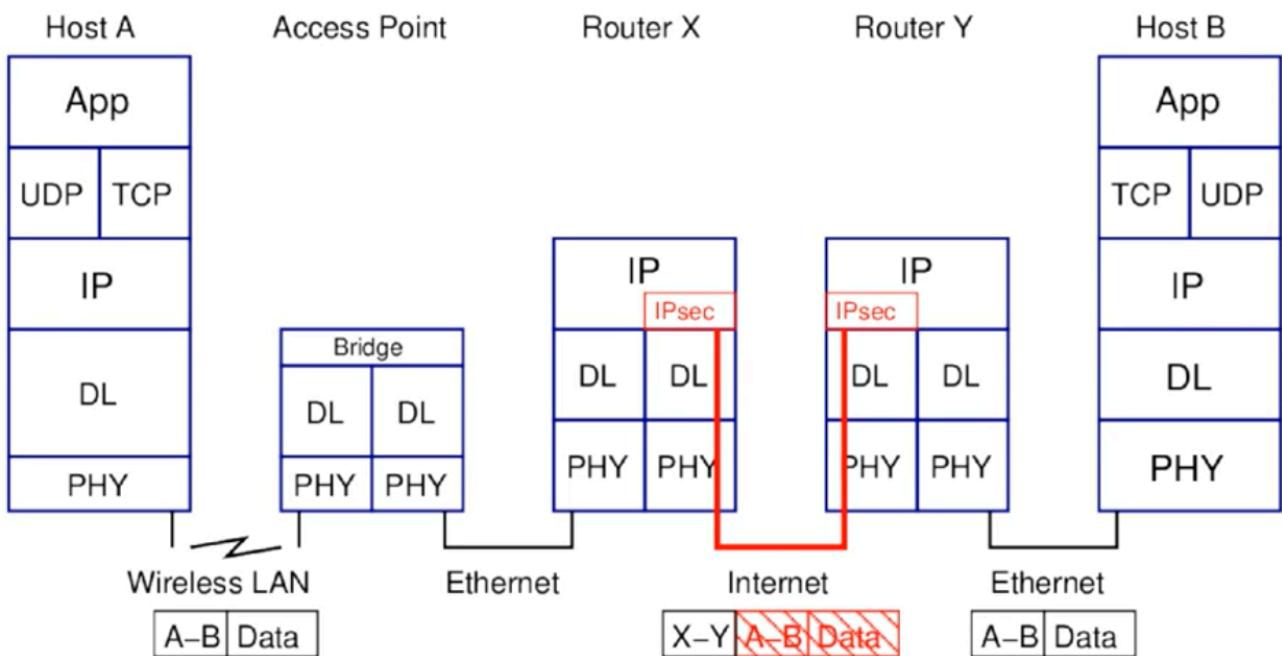


IPSec usato in **tunnel mode**.

Network Level Security: IPsec Host-to-Router



Network Level Security: IPsec Router-to-Router



Saranno analizzati alcuni dei principali protocolli di sicurezza, partendo da quelli più in basso nella pila protocollare TCP/IP.

IP Security (cenni)

IPSec

IPsec (*Internet Protocol Security*) è un sistema che permette di proteggere il traffico IP a livello network. La ragione principale per usare IPsec risiede nel fatto che il protocollo IP non ha alcuna caratteristica di protezione o di autenticazione.

IPsec è in grado di proteggere il traffico raggiungendo i seguenti obiettivi:

- **Confidenzialità**: i dati vengono criptati in modo tale che nessuno possa leggerli all'infuori del mittente e del ricevente.
- **Integrità**: usando una funzione *hash* per ottenere il *digest* dei dati scambiati, il mittente e il ricevente possono essere ragionevolmente certi che una eventuale modifica venga rilevata.
- **Autenticazione**: il mittente e il ricevente si autenticheranno l'un l'altro in modo che siano sicuri di comunicare con l'entità corretta.
- **Anti-replay**: anche se un pacchetto è stato criptato e autenticato, un attaccante potrebbe tentare di catturare questi pacchetti per inviarli nuovamente. IPsec rileva i pacchetti duplicati usando un *sequence number*.

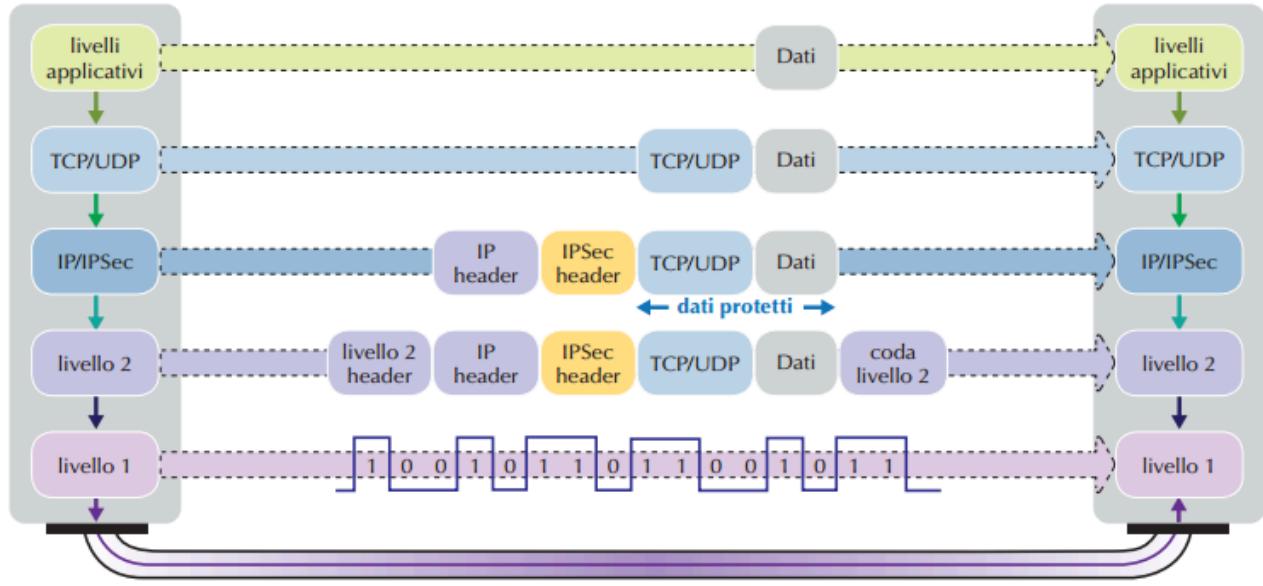
IPsec non è un singolo protocollo ma piuttosto un'architettura di sicurezza a livello Network, composta da vari protocolli e da altri elementi. I protocolli principali che costituiscono IPsec sono tre:

1. **Authentication Header (AH)**: garantisce l'autenticazione e l'integrità del messaggio ma non offre la confidenzialità;
2. **Encapsulating Security Payload (ESP)**: fornisce autenticazione, confidenzialità e controllo di integrità del messaggio;
3. **Internet Key Exchange (IKE)**: è un protocollo di livello applicativo che implementa lo scambio delle chiavi per realizzare il flusso crittografato.

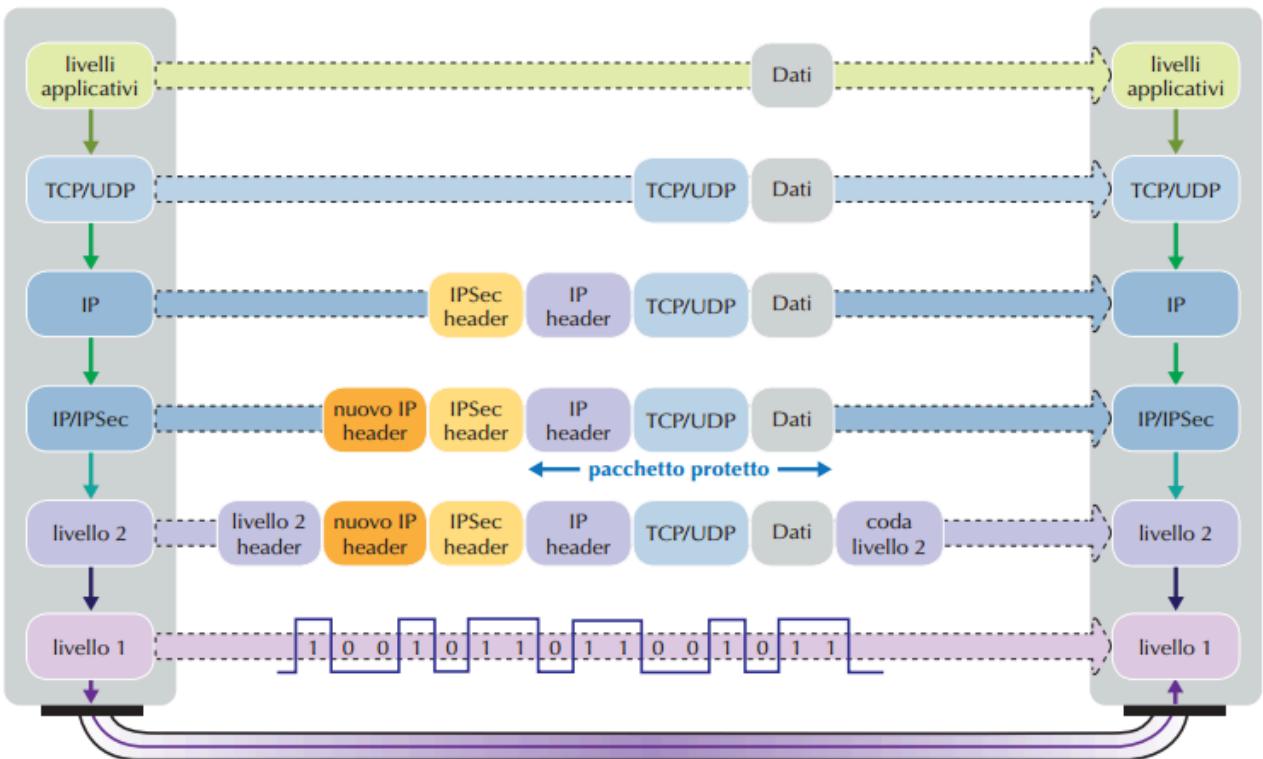
AH (*Authentication Header*) ed **ESP** (*Encapsulating Security Payload*) non si preoccupano dello scambio delle chiavi e presumono che i due interlocutori si siano già accordati usando il protocollo **IKE** (*Internet Key Exchange*) creando tra loro una **Security Association (SA)**, ovvero un "contratto" che specifica quali meccanismi di sicurezza utilizzare e con quali chiavi. È quindi affidato a **IKE** (*Internet Key Exchange*) il compito di negoziare e gestire le diverse **Security Association**, secondo politiche definite localmente. Una **SA** può poi essere associata ad **AH** (*Authentication Header*) o a **ESP** (*Encapsulating Security Payload*). Una **SA** (*Security Association*) ha un tempo di vita, che può essere specificato in termini temporali oppure come quantità di dati trasferiti.

Sia **AH** (*Authentication Header*), sia **ESP** (*Encapsulating Security Payload*) possono essere utilizzati in **modalità trasporto** o in **modalità tunnel** (*tunneling*).

In **modalità trasporto** (*end-to-end*) IPsec agisce tra due end system e aggiunge gli header dei protocolli utilizzati (**AH** o **ESP**) tra header IP e l'header del protocollo di trasporto (TCP o UDP).



In **modalità tunneling** il pacchetto IP originario viene interamente incapsulato in un nuovo pacchetto IP che riporta nel nuovo header IP gli indirizzi di uno o due intermediate system che implementano IPsec.



VPN (Virtual Private Network)

La rete VPN

L'accesso da remoto alle risorse di una LAN pone innanzitutto dei problemi di sicurezza. Per questo motivo, a volte, le aziende preferiscono adottare costose linee dedicate per utilizzarle in proprio. Una soluzione alternativa, ormai molto diffusa, è quella di appoggiarsi alle *Virtual Private Network (VPN)*, reti private che sfruttano Internet come infrastruttura.

Poiché Internet è pubblica e aperta a chiunque, è relativamente semplice intercettare i dati. Per evitare questo pericolo, le VPN si servono di "tunnel virtuali" creati tra le varie sedi: i dati sono cifrati all'entrata del tunnel e decifrati all'uscita, in modo che possano viaggiare in modo sicuro e protetto.

Negli ultimi anni è aumentata l'esigenza di avere sedi aziendali distribuite su un vasto territorio geografico, così come la necessità di lavorare in mobilità. Per questa ragione è diventato strategico poter comunicare in sicurezza, come si lavora nella LAN aziendale.

La **VPN** risponde in pieno a queste esigenze e, in particolare, trova piena realizzazione in alcuni ambiti:

- collegamento tra sedi periferiche aziendali e con la sede centrale (LAN-to-LAN VPN o Site-to-Site VPN);
- collegamento alla LAN aziendale di un lavoratore remoto che opera in mobilità (Client-to-LAN VPN);
- collegamento a risorse aziendali da parte di consulenti, clienti e fornitori.

Le **VPN** offrono molti vantaggi:

- riduzione dei costi: non è necessario affittare delle linee dedicate (molto costose) per collegare tra loro le varie sedi;
- utilizzo di risorse già presenti: è possibile utilizzare le normali linee di collegamento a Internet già presenti nelle aziende;
- flessibilità di accesso alle risorse aziendali: con gli opportuni permessi di accesso, si può accedere a una risorsa della propria azienda indipendentemente dalla sede in cui ci si trova in quel momento;
- trasmissioni sicure: tutte le trasmissioni sono automaticamente cifrate;
- architettura facilmente scalabile: è molto semplice e veloce aggiungere ulteriori sedi alla rete aziendale;
- supporto dei servizi di nuova generazione compresi video e voce con protocollo di cifratura;
- semplicità d'uso.

Modalità di connessioni di una VPN

Un'azienda che ha diverse sedi dislocate anche a grande distanza tra loro, generalmente tenderà ad adottare una VPN come tecnologia per gestire l'insieme delle sedi remote come un'unica rete locale aziendale, cercando di estendere in ambito geografico la propria rete LAN privata e realizzando una WAN privata per il proprio business.

Una simile gestione permette inoltre di considerare anche forme alternative di lavoro come lo homeworking²⁴ e il teleworking²⁵, oltre alla possibilità di collaborare con partner consociati creando LAN estese che vanno oltre i confini della singola azienda.

Esistono due modalità di connessioni per una VPN:

- **remote-access VPN** che permette ad un *teleworker* di emulare, tramite Desktop Remoto, il desktop dell'ufficio principale;
- **site-to-site VPN** che consente alle aziende di ampliare le risorse di rete alle filiali, agli uffici domestici e ai siti di partner.

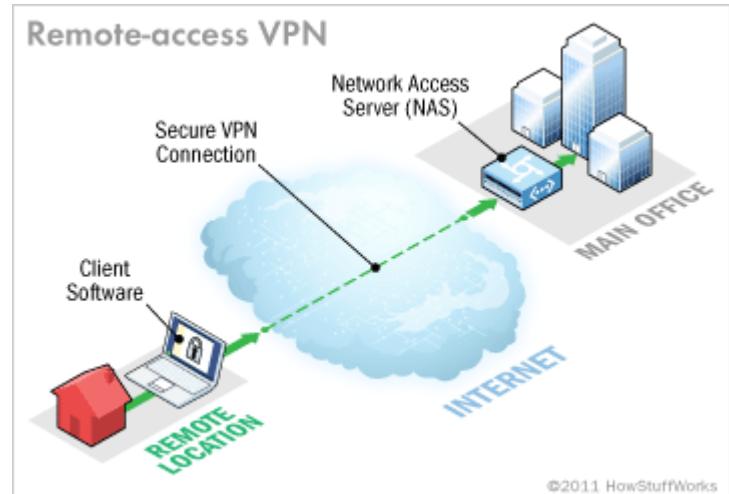
Remote-access VPN

Una **Remote-Access VPN** consente ai singoli utenti di stabilire connessioni sicure con la LAN aziendale remota.

Gli utenti possono accedere alle risorse protette della rete locale lavorativa come se fossero direttamente collegati ai server della rete aziendale.

Ci sono due componenti indispensabili alla realizzazione di un accesso remoto VPN:

1. un **server di accesso** alla rete identificato con l'acronimo **NAS** (*Network Access Server*);
2. un **software VPN Client**.



Un **NAS** (*Network Access Server*) è un software che richiede all'utente di fornire le credenziali valide per accedere alla VPN. Un **NAS** funziona come un gateway per limitare gli accessi ad un dispositivo protetto, in questo caso la VPN, ma potrebbe essere un qualsiasi altro dispositivo o servizio. Quando un client si collega, il NAS effettua un collegamento ad un altro dispositivo, un **server AAA**, per verificare se le credenziali fornite dal client sono valide. Sulla base della risposta ricevuta il **NAS** autorizza o no l'accesso al dispositivo protetto. Il **NAS** non contiene informazioni su

²⁴ Lo *homeworker* svolge il proprio lavoro da casa (ufficio domestico), collegandosi alla rete aziendale.

²⁵ Il *teleworker* svolge il proprio lavoro collegandosi alla rete aziendale da qualsiasi luogo, usando il proprio dispositivo mobile.

quali client possono collegarsi o su quali siano le credenziali valide, infatti per autorizzare il client invia le credenziali ad un **server AAA** (*Authentication, Authorization, Accounting*) che sa come gestirle, tipicamente un server RADIUS. Per ogni connessione VPN, il **server AAA** conferma le credenziali del client che si è autenticato (**Authentication**), identifica ciò a cui il client può accedere tramite la connessione (**Authorization**) e tiene traccia di ciò che il client fa mentre è loggato (**Accounting**).

La componente del **software VPN Client** è necessaria per i dipendenti che desiderano utilizzare la VPN dal proprio computer, in quanto questo software stabilisce e mantiene una connessione alla rete VPN. La maggior parte dei sistemi operativi odierni sono dotati di software in grado di connettersi alle reti **Remote-Access VPN**, anche se alcune **VPN** potrebbero richiedere agli utenti di installare un'applicazione specifica.

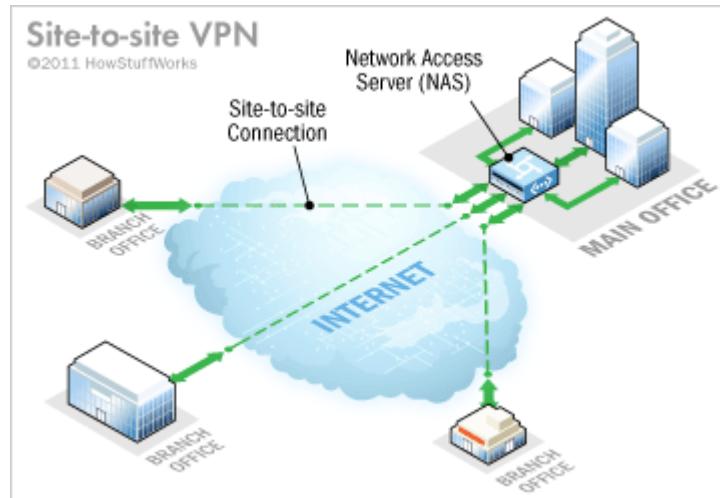
Un **Remote-Access VPN** è generalmente adatto per i singoli dipendenti o utenti, o per aziende con filiali costituite da piccoli uffici.

Site-to-site VPN

In aziende con filiali grandi e con decine o addirittura centinaia di dipendenti occorre affidarsi a un altro tipo di **VPN** che permettono di mantenere le aziende collegate LAN-to-LAN.

Un **Site-to-site VPN** permette di stabilire connessioni sicure attraverso una rete pubblica come Internet.

Il **Site-to-Site VPN** estende la rete aziendale, rendendo disponibili le risorse della sede principale alle sedi secondarie. Un'azienda in crescita con decine di filiali in tutto il mondo rappresenta il tipico esempio di un soggetto che ha bisogno di una **Site-to-site VPN**.



Ci sono due tipi di **Site to-site VPN**:

- **Intranet²⁶ based** utilizzata da società che hanno una o più sedi remote, ognuna con una propria LAN da riunire in un'unica rete WAN privata.
- **Extranet²⁷ based** usata da società che hanno rapporti di forte collaborazione con altre società, come ad esempio un partner fornitore o un'azienda cliente. In questi casi è possibile costruire una VPN extranet che collega le LAN delle diverse imprese, permettendo alle aziende di lavorare insieme in un ambiente

²⁶ Una **rete Intranet** è una rete interna aziendale che impiega le tecnologie e i protocolli di Internet.

²⁷ Una **rete Extranet** è una rete che impiega le tecnologie e i protocolli di Internet per collegare un'azienda ai propri fornitori, clienti o ad aziende consociate.

sicuro, condividendo le risorse e senza dover consentire l'accesso preventivo alla propria intranet.

Anche se lo scopo di un **Site-to-site VPN** è diverso da quello di un **Remote-access VPN**, i due tipi di VPN potrebbero utilizzare parte dello stesso software e gli stessi dispositivi. Idealmente però, un **Site-to-Site VPN** dovrebbe **eliminare in modo trasparente la necessità per ogni host di eseguire il software VPN Client**. Per raggiungere questo scopo vengono utilizzati alcuni dispositivi dedicati:

- **VPN Concentrator** è un dispositivo che sostituisce il **server AAA** installato su un server generico. L'hardware e il software lavorano insieme per stabilire il tunnel VPN e gestire un gran numero di connessioni simultanee.
- **VPN-enabled/VPN-optimized Router** è un tipico router delegato al traffico in rete ma con la caratteristica aggiuntiva di poter instradare i pacchetti utilizzando protocolli specifici per le VPN.
- **VPN-enabled Firewall** che rappresenta un firewall tradizionale adibito a proteggere il traffico tra le reti, ma con la caratteristica aggiunta di poter gestire il traffico utilizzando protocolli specifici per le VPN.
- **VPN Client** che è costituito da un software in esecuzione su un dispositivo dedicato che funge da tunnel-interfaccia per connessioni multiple. Il suo compito è anche quello di evitare che su ogni host sia in esecuzione un software VPN Client.

La sicurezza nelle VPN

Il fatto che le reti VPN abbiano un ambito geografico (WAN) e utilizzino la rete pubblica insicura (Internet), obbliga ad affrontare seri problemi legati alla sicurezza dei dati alla riservatezza delle trasmissioni concentrando l'attenzione su tre fattori cruciali: l'**autenticazione**, la **cifratura** e il **tunneling**.

L'**autenticazione** fornisce la verifica di accesso alla VPN solo ad utenti autorizzati ad usare i servizi della rete.

La **cifratura** permette di mantenere confidenziali le comunicazioni, oltre a garantire l'integrità e l'autenticità dei dati, e nell'ambito delle VPN possono essere usati un'ampia gamma di algoritmi crittografici simmetrici (3DES, CAST, IDEA, ecc.) per cifrare il traffico in rete. Sia l'algoritmo da usare, sia le chiavi segrete che l'algoritmo di cifratura utilizzato, sono concordate e scambiate tra mittente e destinatario attraverso protocolli di sicurezza, come ad esempio il protocollo **IKE** (*Internet Key Exchange*) nell'ambito di **IPSec**.

Il **tunneling** nelle VPN si riferisce a un processo di immissione di un intero pacchetto all'interno di un altro pacchetto prima di essere trasportato su Internet per introdurre funzionalità crittografiche. Il pacchetto esterno protegge il contenuto dalla vista del pubblico e assicura che il pacchetto si muova all'interno di un tunnel virtuale. **IPSec** è uno dei protocolli usati nell'ambito delle VPN per effettuare il **tunneling**.

Authentication, Authorization, Accounting

Le **reti VPN** sono **reti private**, e quindi **per potervi accedere occorre autenticarsi**.

Si definisce **autenticazione dell'identità il processo tramite il quale un sistema informatico, un applicativo o un utente, verifica la corretta**, o almeno presunta tale, **identità di un altro sistema informatico, applicativo o utente che vuole comunicare attraverso una connessione, concedendogli l'autorizzazione a usufruire dei relativi servizi associati**.

Come già detto la porta di accesso di un client alla sua VPN risulta essere un server **NAS** dotato di un processo di autenticazione che utilizza un **server AAA** dedicato a questo scopo. Solo dopo aver superato la fase di **autenticazione** si viene **autorizzati** ad accedere ai servizi della rete.

La VPN dovendo funzionare come la LAN aziendale, **richiederà l'impostazione di opportune autorizzazioni per l'accesso ai servizi della rete, specificando il dominio per ciascun utente creato** (*policy di servizio*). Alcuni di questi servizi, come la condivisione di risorse (dischi, stampanti ecc.), può essere autorizzata solo per il personale dell'azienda, mentre ai client VPN esterni può essere concesso di accedere ai servizi di navigazione Internet (HTTP, HTTPs) o di posta elettronica (SMTP, POP3, IMAP4, SMTPs, POP3s, IMAP4s). Altresì un rappresentante di commercio in visita a un cliente potrà collegarsi, dopo essersi autenticato, alla VPN aziendale attraverso Internet e accedere, autorizzato, alla banca dati aziendale e ai servizi per l'inoltro immediato dell'ordine di acquisto del cliente, contribuendo così a ottimizzare i tempi di consegna e garantendo la privacy.

Per controllare che non siano state effettuate azioni indesiderate e non autorizzate, occorre prevedere **meccanismi di Accounting**. Con **Accounting si intendono tutte le azioni volte a misurare e documentare le risorse concesse a un utente durante un accesso**. Ciò può includere la misura della durata della sessione di lavoro di un utente, oppure la misura del quantitativo di traffico dati inviati e ricevuti da un utente nella sessione di lavoro. Tali informazioni possono essere prodotte e memorizzate attraverso dei file log per poi essere successivamente utilizzate per applicare una tariffazione, per effettuare un controllo delle autorizzazioni, per una pianificazione della capacità o per fini statistici di analisi dei trend. Sostanzialmente dall'analisi dell'**Accounting** in un **server AAA** si possono trarre molte informazioni utili.

Cifratura

In termini di sicurezza spesso risulta fondamentale il rapporto fiduciario tra l'azienda che vuole creare una VPN e il fornitore di servizi Internet che gestisce l'infrastruttura pubblica, in quanto **il fornitore deve garantire la confidenzialità**.

Nella sicurezza informatica per confidenzialità, o riservatezza, si intende la protezione dei dati e delle informazioni scambiate tra un mittente e uno o più

destinatari nei confronti di terze parti. Tale protezione deve essere realizzata a prescindere dalla sicurezza del sistema di comunicazione utilizzata, anche e soprattutto quando il sistema di comunicazione utilizzato è intrinsecamente insicuro, come per esempio la rete Internet. Di conseguenza, per quanto riguarda la **cifratura**, le **VPN utilizzano un'ampia gamma di algoritmi di crittografia** (3DES, CAST, IDEA, ecc.) **per cifrare il traffico in rete.** Sia l'algoritmo da usare, sia le chiavi segrete che l'algoritmo stesso utilizza sono concordate e scambiate tra mittente e destinatario attraverso protocolli di sicurezza. Nello specifico caso delle **reti VPN**, viene soprattutto utilizzato il **protocollo Internet Key Exchange (IKE)** nell'ambito di **IPsec**. Il compito principale di **IKE** è proprio implementare lo "scambio delle chiavi" per cifrare i pacchetti. **IKE** automatizza la gestione delle chiavi e permette all'amministratore di gestire un maggior numero di reti sicure.

La maggior parte dei **protocolli per la sicurezza nelle VPN garantisce anche l'integrità e l'autenticità dei dati** e quindi, in pratica, che i pacchetti ricevuti non siano stati modificati durante la trasmissione e che provengano da fonte certa. L'integrità e l'autenticità vengono garantite mediante meccanismi di firma digitale e certificato digitale.

Tunneling

Lo **scopo dei protocolli di tunneling è aggiungere un livello di sicurezza al fine di proteggere ogni pacchetto nel suo viaggio su Internet.**

In realtà le **VPN possono essere realizzate sia in modalità trasporto, sia in modalità tunnel.**

Nella modalità trasporto hanno un ruolo fondamentale i software impiegati.

Immaginiamo un lavoratore mobile (*teleworker*) che deve collegarsi alla centrale attraverso Internet da qualsiasi punto del Mondo. Il suo dispositivo (notebook, palm, wap, ecc.) dovrà dotarsi di un software per VPN. Il collegamento potrà essere effettuato con qualsiasi ISP in quanto cifratura e decifratura dei dati verranno garantite dal software a bordo del notebook e dagli apparati riceventi presso la sede centrale. Internet lascerà in chiaro solamente le informazioni di instradamento IP.

Nella modalità tunnel hanno un ruolo fondamentale gli apparati, e in particolar modo i router e i firewall.

Questa tecnologia è quella tipica di un collegamento fra una filiale e la sede centrale. Gli apparati sono preposti a trasformare e codificare tutto il traffico fra gli end-point, mentre per gli utenti finali non vi è alcuna percezione delle trasformazioni applicate (trasparenza). In questa modalità i pacchetti di dati vengono inseriti in ulteriori pacchetti richiedendo una maggior performance alla rete.

Il termine **tunneling** si riferisce a un insieme di tecniche per cui **un protocollo viene incapsulato in un protocollo dello stesso livello o di livello superiore**: nel caso delle reti **VPN** viene inserito uno strato che introduce funzionalità crittografiche. Il **tunneling è dunque il processo di immissione di un intero pacchetto all'interno di un altro pacchetto prima di essere trasportato su Internet.** Il

pacchetto esterno protegge il contenuto dalla vista del pubblico e assicura che il pacchetto si muova all'interno di un tunnel virtuale. Tale stratificazione di pacchetti viene chiamata **incapsulamento**.

Gli host o i dispositivi di rete su entrambe le estremità del tunnel (*tunnel interface*), possono incapsulare i pacchetti in uscita e riaprire i pacchetti in entrata. Gli utenti (a un'estremità del tunnel) e il personale IT (a una o entrambe le estremità del tunnel) dovranno configurare le interfacce di cui sono responsabili per utilizzare un determinato protocollo di tunneling.

Un protocollo di tunneling, chiamato anche un protocollo di incapsulamento, è un modo standardizzato per incapsulare i pacchetti, e il pacchetto viaggia con lo stesso protocollo di trasporto che avrebbe utilizzato senza il tunnel. Come noto, il protocollo di trasporto definisce le modalità con cui ogni dispositivo trasferisce pacchetti attraverso Internet mediante il proprio ISP. Ciascun pacchetto interno mantiene però ancora il proprio protocollo (*passenger protocol*), come Internet Protocol (IP) o AppleTalk, che definisce come si viaggia sulle LAN poste a ciascuna estremità del tunnel.

I **protocolli usati per il tunneling** sono diversi e questi sono anche le tecnologie usate nella tipologia **Secure VPN**:

- IPsec (*IP security*):
 - *Encapsulating Security Payload* (ESP): fornisce autenticazione, confidenzialità e controllo di integrità del messaggio.
 - *Authentication Header* (AH): garantisce l'autenticazione e l'integrità del messaggio ma non offre la confidenzialità.
 - *Internet Key Exchange* (IKE): implementa lo scambio delle chiavi per realizzare un flusso cifrato.
- SSL/TLS (*Secure Sockets Layer/Transport Layer Security*):
 - garantisce confidenzialità e affidabilità delle comunicazioni su rete pubblica;
 - protegge da intrusioni, modifiche o falsificazioni;
 - si poggia sul solo protocollo di trasporto TCP.
- BGP/[MPLS](#) (*Border Gateway protocol/Multi-Protocol Label Switching*):
 - utilizzato su reti a commutazione di pacchetto, tipicamente IP;
 - le decisioni di instradamento, alle quali viene associata una etichetta, vengono prese in modo asincrono rispetto al trasporto del traffico (di solito il percorso viene stabilito prima della spedizione dei pacchetti) e si applicano ad una intera classe di destinazioni;
 - MPLS non può essere considerato un protocollo di rete, piuttosto è una tecnologia che all'interno delle reti potenzia il trasporto del traffico;
 - instrada più tipi di traffico (dati, voce, video) sullo stesso canale, consentendo di differenziare la banda di trasmissione in base al tipo di traffico e di aggirare le zone congestionate e i collegamenti interrotti.
- PPTP (*Point-to-Point Tunneling Protocol*):
 - sviluppato da Microsoft™;
 - assicura autenticazione, cifratura e compressione dei dati;
- IEEE 802.1Q (*Ethernet tagged VLAN*):

- aggiunge al frame Ethernet un tag contenente il VLAN Identifier corrispondente alla VLAN interessata al trasferimento;
- permette il trasferimento promiscuo di frame appartenenti a VLAN diverse utilizzando una sola interfaccia e un singolo collegamento di rete;
- SSH (*Secure SHell*) tunneling:
 - trasferisce qualsiasi dato in rete utilizzando una connessione cifrata;
- SOCKS (*SOCKetS*):
 - standard IETF definito nello RFC 1928;
 - proxy trasparente che permette di effettuare connessioni TCP dirette tra computer su due reti IP differenti nei casi in cui un instradamento diretto non sia disponibile.
- GRE (*Generic Routing Encapsulation*):
 - sviluppato dalla CISCO™;
 - crea un collegamento point-to-point virtuale in maniera che nessuno dei due punti si debba preoccupare dell'infrastruttura su cui passa la comunicazione.
- L2TP (*Layer 2 Tunneling Protocol*):
 - è un protocollo di livello 5 (*sessione*) che agisce però come un protocollo di livello 2 (*data-link*) usando pacchetti UDP per incapsulare i pacchetti L2TP e per mantenere una connessione point-to-point;
 - deve essere associato ad un altro protocollo per implementare autenticazione, confidenzialità e integrità dei dati (solitamente IPsec).
- L2TPv3 (*Layer 2 Tunneling Protocol version 3*):
 - evoluzione di L2TP creato come alternativa a MPLS.

Fra questi i **principali protocoli per la sicurezza** sono **IPsec (IP security)**, **SSL/TLS (Secure Sockets Layer/Transport Layer Security)** e **BGP/MPLS (Border Gateway Protocol/Multi-Protocol Label Switching)**.

Le **VPN IPsec-based, SSL/TLS-based e BGP/MPLS-based sono diventate le soluzioni VPN scelte dalla maggior parte delle aziende per collegare uffici remoti, utenti remoti e partner di business**, in quanto, con le loro differenze, forniscono comunicazioni sicure con diritti di accesso su misura per i singoli utenti, che si tratti di dipendenti, consulenti o partner, aumentano la produttività, ampliando la rete e le applicazioni aziendali, riducendo i costi delle comunicazioni e accrescendo la flessibilità.

Tipi di VPN

Il funzionamento delle **VPN**²⁸ è articolato e complesso, ma in generale il loro funzionamento può essere raggruppato in tre categorie distinte: **Trusted VPN**, **Secure VPN** e **Hybrid VPN**.

²⁸ [Virtual Private Network - Wikipedia](#)

Trusted VPN

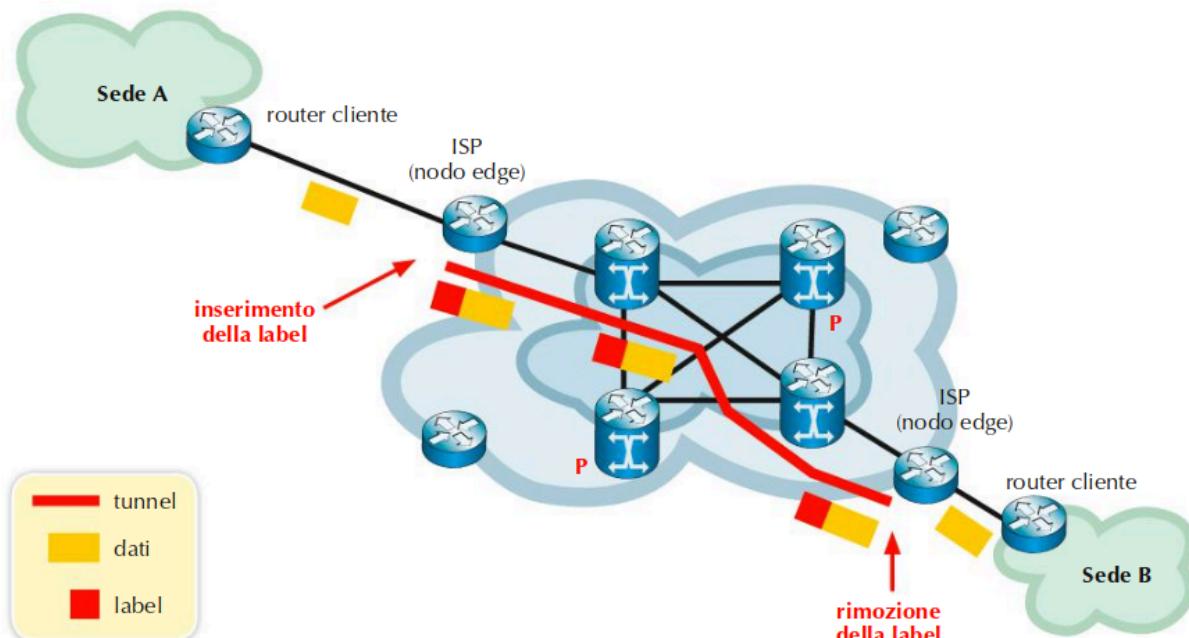
Una **Trusted VPN** garantisce che nessun terzo non autorizzato possa usufruire del circuito virtuale di comunicazione del cliente. Questo implica che il cliente abbia un proprio indirizzo IP e una propria politica di sicurezza.

Il circuito virtuale è instaurato attraverso uno o più "interruttori" di comunicazione che potrebbero essere compromessi da chi vuole disturbare il traffico della rete. Il cliente di una **Trusted VPN** si aspetta quindi che il fornitore del servizio, di solito il provider, mantenga l'integrità del circuito in modo da impedire l'accesso da parte di intrusi.

Le aziende che utilizzano una **Trusted VPN** vogliono avere la sicurezza che i loro dati si muovano attraverso una serie di **percorsi che hanno proprietà specifiche e che sono controllati da un ISP** (*Internet Service Provider*). Il cliente ha quindi fiducia che i percorsi attraverso i quali questi dati si muovono siano mantenuti sicuri secondo i criteri di un precedente accordo, anche se generalmente il cliente non conosce quali siano i percorsi utilizzati dal fornitore della **Trusted VPN**.

Più recentemente i fornitori di servizio hanno cominciato a offrire un nuovo tipo di **Trusted VPN**, questa volta usando Internet invece della rete telefonica come substrato di comunicazione. Queste nuove **Trusted VPN** non offrono sicurezza, ma danno ai clienti un modo di creare facilmente segmenti di rete su vasta scala (WAN). I segmenti **Trusted VPN** possono essere inoltre controllati da un posto unico e spesso con una qualità di servizio garantita (**QoS - Quality of Service**) dal provider.

In una **Trusted VPN** il pacchetto che parte dal router della sede A viene inoltrato a quello del fornitore (*nodo Edge*) su cui è implementato un protocollo che crea un circuito virtuale (MPLS - *MultiProtocol Label Switching*). Al pacchetto viene aggiunta, tra l'header di livello 2 (PPP o Ethernet) e quello di livello 3 (IP), una "label" (una nuova intestazione) che lo identifica e gli permette di viaggiare, passando tra i router di Core, fino ad arrivare al router dell'ISP di destinazione, che rimuove la "label" e consegna il pacchetto al router dell'utente.



Secure VPN

In una **Secure VPN** i dati viaggiano in un tunnel virtuale protetto sfruttando i protocolli IPSec e TLS/SSL, che garantiscono la riservatezza, l'autenticazione e l'integrità. Non viene però garantito il percorso seguito come avviene nella Trusted VPN.

In particolare, le VPN che utilizzano TLS/SSL consentono agli utenti di collegarsi, tramite browser, da qualsiasi postazione Internet, per stabilire connessioni sicure tra gli utenti remoti e le risorse interne alla rete privata. Queste applicazioni "clientless VPN" o "Web VPN", a differenza di quanto accade nei sistemi che sfruttano IPSec, non necessitano di particolari privilegi, come l'apertura di specifiche porte TCP, infatti l'utente, dopo l'autenticazione, può accedere in sicurezza alle risorse della LAN.

Il motivo principale per cui le società usano una **Secure VPN** è che possono trasmettere informazioni delicate su Internet senza temere che vengano intercettate.

Le **Secure VPN** sono particolarmente utili per permettere accessi remoti da parte di utenti connessi a Internet da zone non controllate dall'amministratore di rete. Una **Secure VPN** che usa ad esempio IPSec potrebbe essere la scelta migliore nel caso ci si appoggi alla connessione pubblica di Internet, in quanto fornisce una comunicazione confidenziale relativamente a basso costo. È il caso, ad esempio, di un esercente che può aver bisogno occasionalmente di una connessione Secure VPN con un fornitore, oppure per un teleworker che necessiti di connettersi alla rete aziendale.

Hybrid VP

La **Hybrid VPN**²⁹ combina la garanzia dei percorsi della Trusted VPN con la garanzia dei servizi di sicurezza di Secure VPN.

Una **Hybrid VPN** combina l'uso di **MPLS** (*Multiprotocol Label Switching*) e di **IPsec** (*Internet protocol security*) nella stessa **VPN**. Di solito queste due tipologie di **VPN** sono usate in ambiti diversi, ma è possibile combinarle usando la **IPSec VPN** come supporto alla **MPLS VPN**.

IPSec VPN (*Secure VPN*) richiede l'uso di dispositivi presso l'utente finale, tipicamente un router o un dispositivo di sicurezza multipurpose, e viene usata per criptare i dati all'interno di un **tunnel VPN**. Invece **MPLS VPN** (*Trusted VPN*) è fornita dallo ISP, utilizzando quindi la rete e i dispositivi dello ISP stesso. Per connettere le due tipologie di VPN è necessario l'uso di un gateway che permetta di collegare il tunnel IPSec con la MPLS VPN, mantenendo però sempre la sicurezza della VPN per tutto il percorso.

Una **Hybrid VPN** risulta facilmente gestibile se una singola filiale remota necessita di collegarsi alla sede centrale. Se invece più filiali necessitano di un collegamento vicendevole, oltre che con la sede centrale, la gestione dei vari tunnel IPSec

²⁹ [What is a hybrid VPN? - AT&T Business](#)

risulterebbe complesso, mentre il protocollo MPLS della Trusted VPN gestisce naturalmente queste situazioni di rete mesh.

TLS/SSL

La sicurezza delle reti

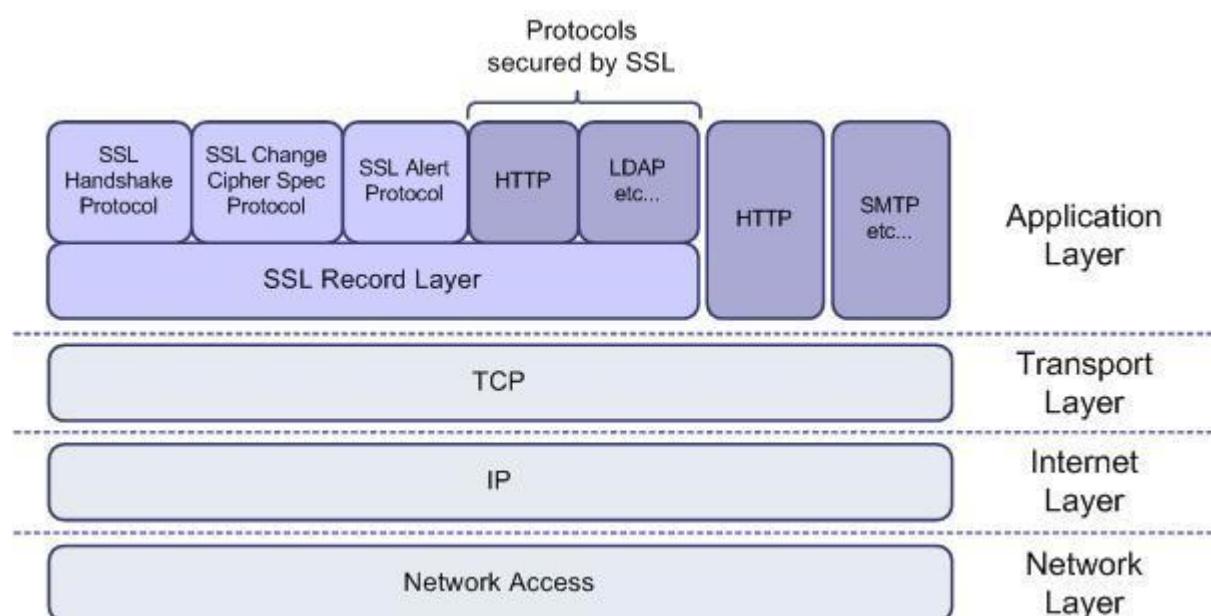
Libro vol.3 - La sicurezza delle connessioni con SSL/TLS

LEGGERE: L. Lo Russo, E. Bianchi, *Sistemi e Reti - Nuova Edizione OPENSCHOOL - Per l'articolazione INFORMATICA degli Istituti Tecnici settore Tecnologico*, vol. 3, ed. Hoepli, 2017.

- UdA4 - La sicurezza delle reti
 - Lezione 2 - La sicurezza delle connessioni con SSL/TLS
 - Generalità pp.183-184
 - Il protocollo SSL/TLS pp.184-186
 - Il funzionamento di TLS pp.186-187
 - Conclusioni pp.188-190

TLS - Transport Layer Security

Il **Transport Layer Security (TLS)** e il suo predecessore Secure Sockets Layer (SSL) ormai deprecato, sono dei protocolli crittografici che permettono una comunicazione sicura dal sorgente al destinatario (end-to-end) su reti TCP/IP fornendo **autenticazione, integrità** dei dati e **cifratura, operando al di sopra del livello di trasporto**. In questo modo viene garantita una comunicazione sicura a tutte le applicazioni che si appoggiano su **TLS/SSL**.

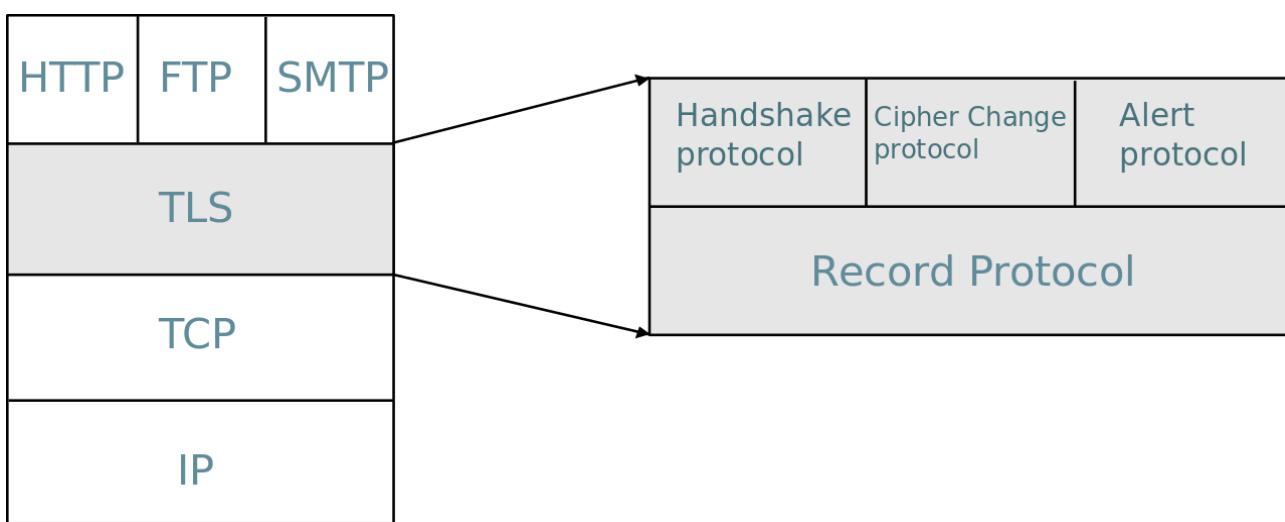


Diverse versioni del protocollo sono ampiamente **utilizzate in applicazioni come i browser, l'e-mail, la messaggistica istantanea e il voice over IP**. Un esempio di applicazione di TLS/SSL è nel protocollo **HTTPS**.

Il protocollo TLS/SSL garantisce:

- **segretezza** tramite l'utilizzo di una **chiave simmetrica**, ad esempio DES, 3DES, AES, IDEA o RC4;
- **autenticazione** tramite lo scambio di **certificati digitali** e l'uso di un **sistema a chiave pubblica**, come ad esempio **RSA**;
- **integrità** utilizzando un **MAC** (*Message Authentication Code*) tramite l'uso di funzioni hash come SHA o MD5.

Lo stack del protocollo **TLS/SSL** è il seguente:



Esamineremo lo **Handshake Protocol** e il **Record Protocol** che sono alla base della suite e che svolgono le seguenti funzioni:

- Fase 1: **handshake protocol per stabilire la sessione**;
- Fase 2: **TLS record protocol per trasferire i dati**.

Una **sessione** è un'associazione tra un client e un server (definisce i parametri crittografici che possono essere condivisi tra più connessioni). Le sessioni servono per evitare costose negoziazioni di nuovi parametri di sicurezza per ogni connessione.

Fase 1: Handshake Protocol

Gli obiettivi principali dello **Handshake Protocol** sono i seguenti:

- autenticazione del server al client e optionalmente autenticazione del client al server;
- negoziazione dei parametri come l'algoritmo di cifratura, la funzione di hashing, le chiavi di sessione.

Lo **Handshake Protocol** si basa su crittografia asimmetrica per lo **scambio dei parametri iniziali ed è responsabile della negoziazione dei parametri di sicurezza di una sessione.**

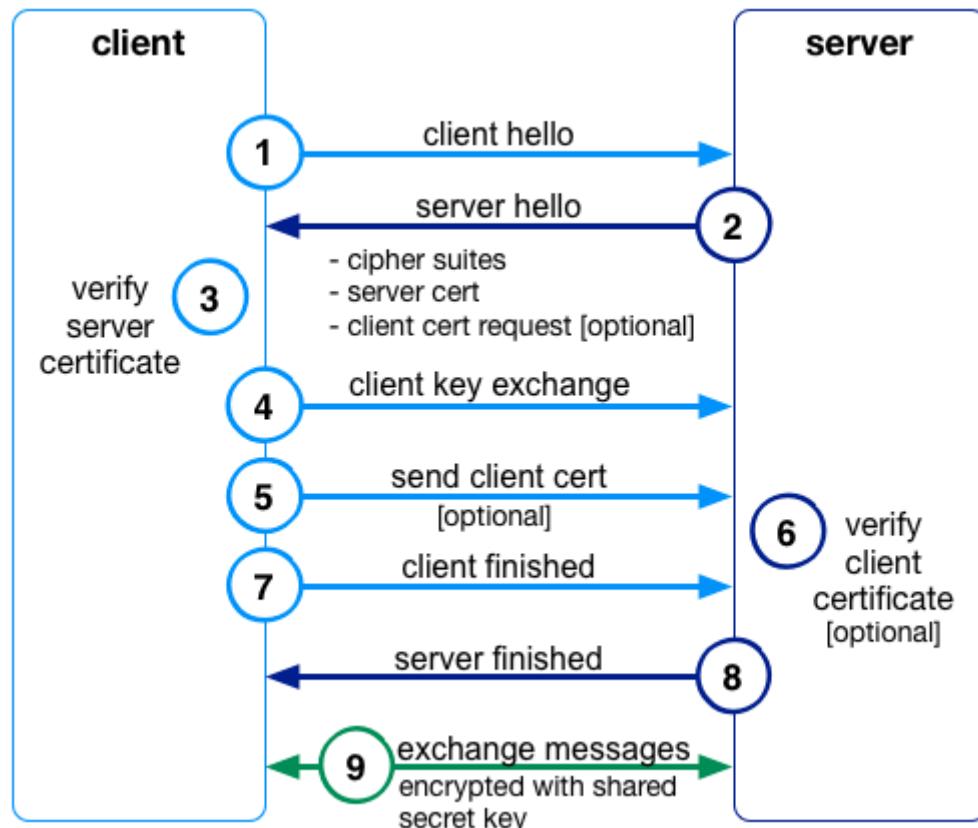
Questa prima fase viene avviata dal client che invia un messaggio "Hello" proponendo la versione del protocollo TLS da usare, le funzioni di cifratura e di hash che è in grado di gestire. Il server risponderà proponendo le funzioni più sicure che implementa, quindi eventualmente con funzioni che non aveva inizialmente proposto il client. Infine il server invierà anche il proprio certificato per autenticarsi verso il client.

L'invio del certificato digitale è importante perché in questo modo il client potrà verificare la validità della chiave pubblica del server tramite la certificazione di una *Certification Authority*, la cui affidabilità dovrà essere validata dal client risalendo la catena delle CA e validandone il relativo certificato digitale.

Se il client verificherà la validità del certificato ricevuto dal server, genererà una chiave di sessione K e la invierà cifrata al server usando la chiave pubblica del server. Nel caso in cui sia necessario autenticare anche il client, quest'ultimo invierà il proprio certificato digitale.

Infine il server decifrerà la chiave simmetrica condivisa di sessione con la quale potrà avviare la comunicazione cifrata.

Lo schema generale è proposto nella figura seguente.

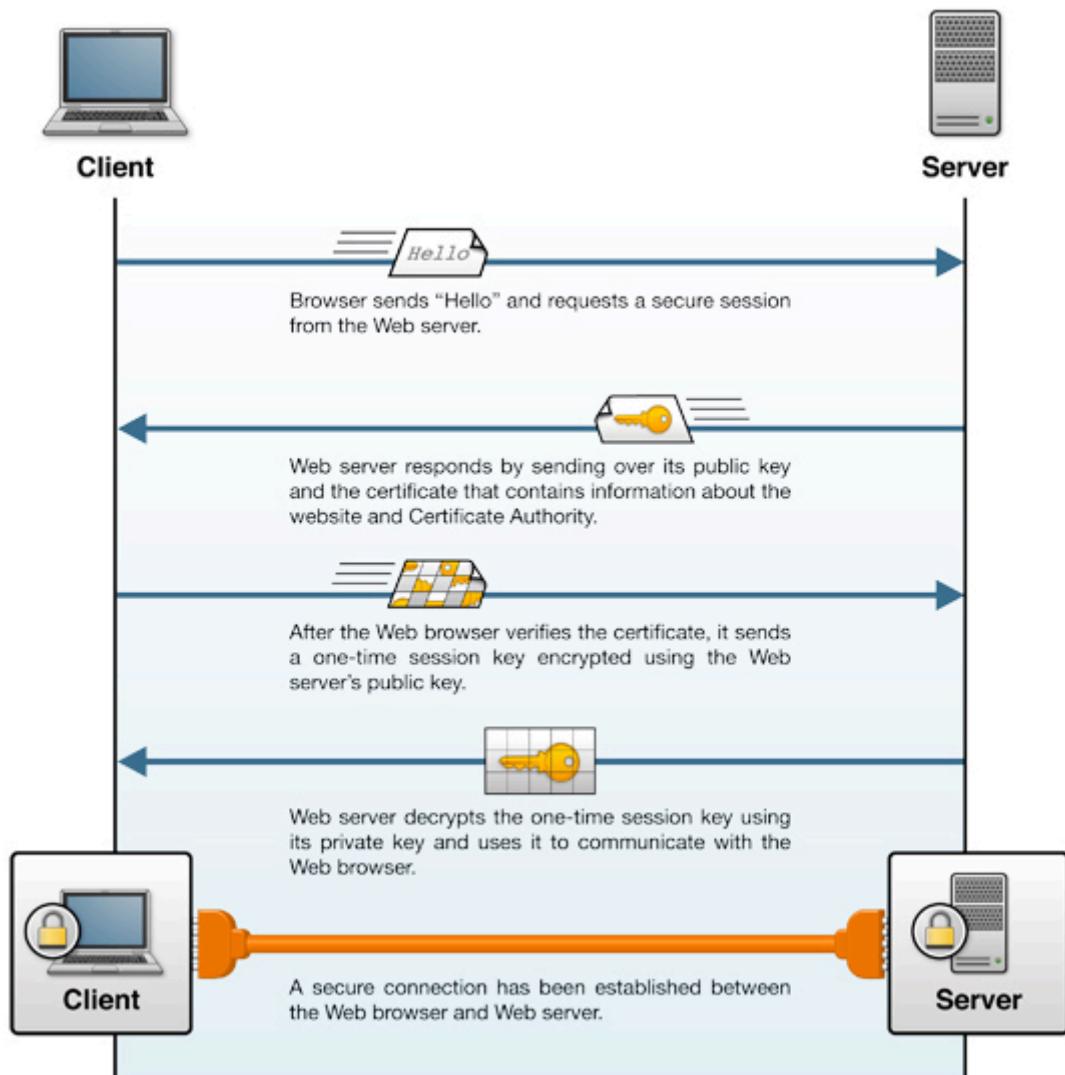


Per iniziare lo handshake il client deve effettuare una richiesta di connessione al server (1). Il server risponderà inviando il suo certificato (2) che il client dovrà verificare per autenticare il server (3).

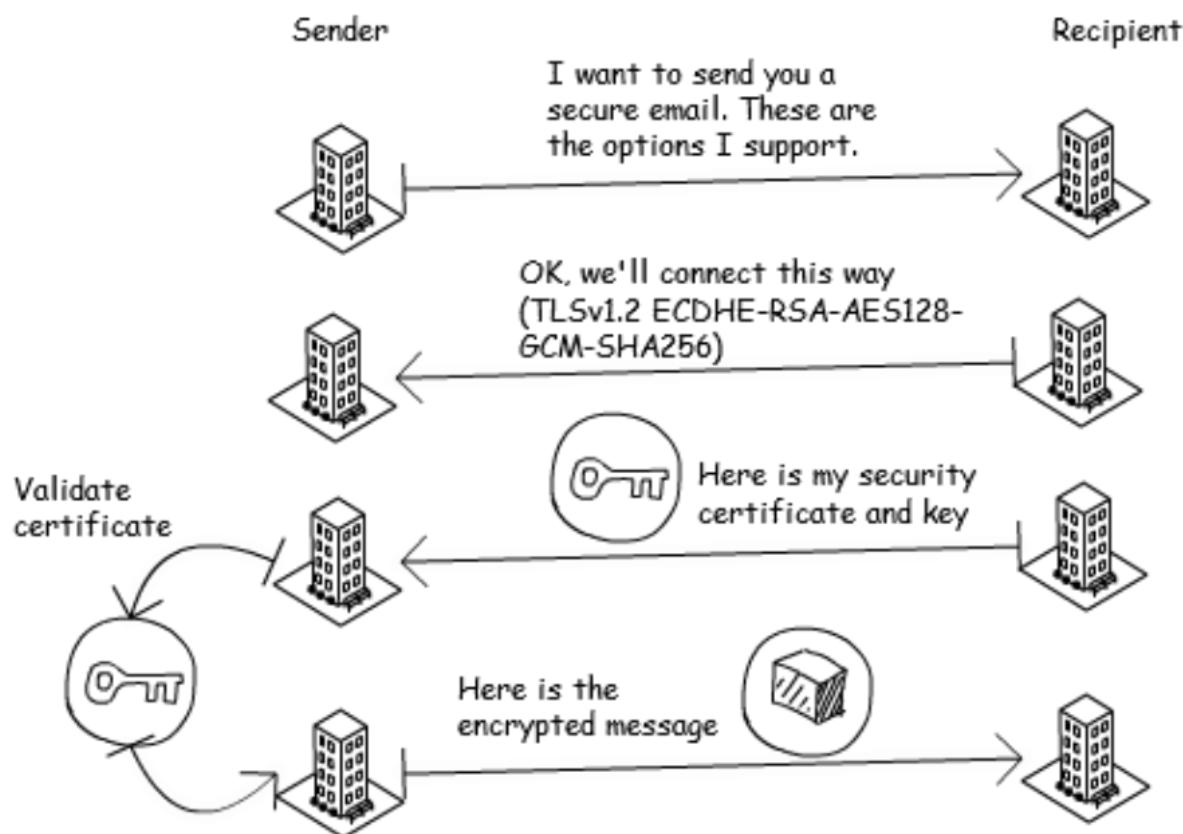
Opzionalmente il server può richiedere al client di autenticarsi inviandogli il suo certificato per raggiungere un livello di sicurezza maggiore (5)(6).

Il client genererà una nuova chiave simmetrica di sessione e, usando la chiave pubblica del server il client la cripterà e la invierà al server (4), avendo la garanzia che solo il server potrà decriptarla usando la sua chiave privata. A questo punto solo il client e il server saranno a conoscenza della chiave simmetrica di sessione e potranno concludere la fase di handshake (7)(8), per iniziare la comunicazione crittata dei dati (9).

La fase di handshake tra un browser e un server Web prevederà i seguenti passi:



Analogamente lo handshake per la posta elettronica prevede le seguenti fasi:



Fase 2: Record Protocol e trasferimento dati

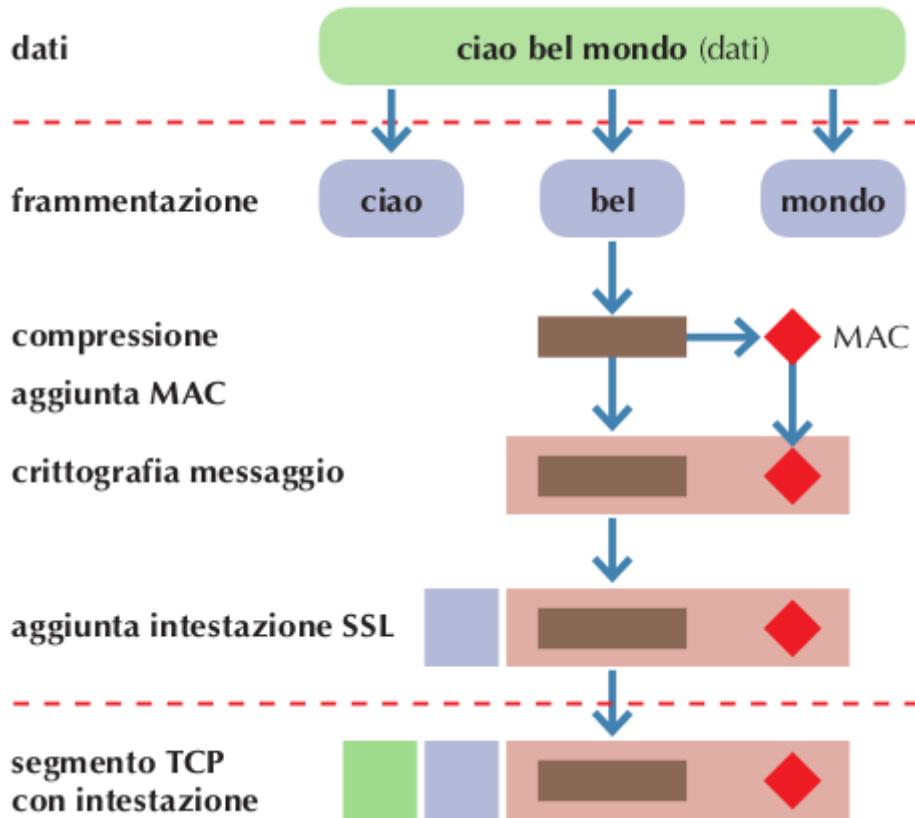
Il **TLS Record Protocol** garantisce sicurezza ed integrità, infatti, una volta concluso lo handshake il canale di comunicazione risulterà sicuro e i dati potranno essere trasferiti in modo cifrato utilizzando la chiave simmetrica di sessione K.

In altre parole TLS si basa sulla chiave di sessione definita nella fase di handshaking per cifrare con crittografia simmetrica.

In trasmissione i dati ricevuti da TLS da parte di un'applicazione vengono suddivisi in blocchi della stessa dimensione, eventualmente vengono compressi, se ne calcola il MAC (*Message Authentication Code*) che garantisce l'integrità, il blocco compresso con il MAC viene criptato con la chiave simmetrica di sessione, si aggiunge lo header TLS e infine il pacchetto viene passato al livello di trasporto per essere inviato.

In ricezione i blocchi in arrivo a TLS dal livello Trasporto vengono decriptati, se ne ricalcola il MAC e lo si confronta con quello ricevuto per verificarne l'integrità, si decomprime il blocco dati e infine quest'ultimo viene passato all'applicazione destinataria con tutti gli altri blocchi che componevano il messaggio originale.

Lo schema generale del funzionamento del **Record Protocol** è il seguente:



S/MIME e posta elettronica

La sicurezza delle reti

Libro vol.3 - La sicurezza nei sistemi informativi

STUDIARE: L. Lo Russo, E. Bianchi, *Sistemi e Reti - Nuova Edizione OPENSCHOOL - Per l'articolazione INFORMATICA degli Istituti Tecnici settore Tecnologico*, vol. 3, ed. Hoepli, 2017.

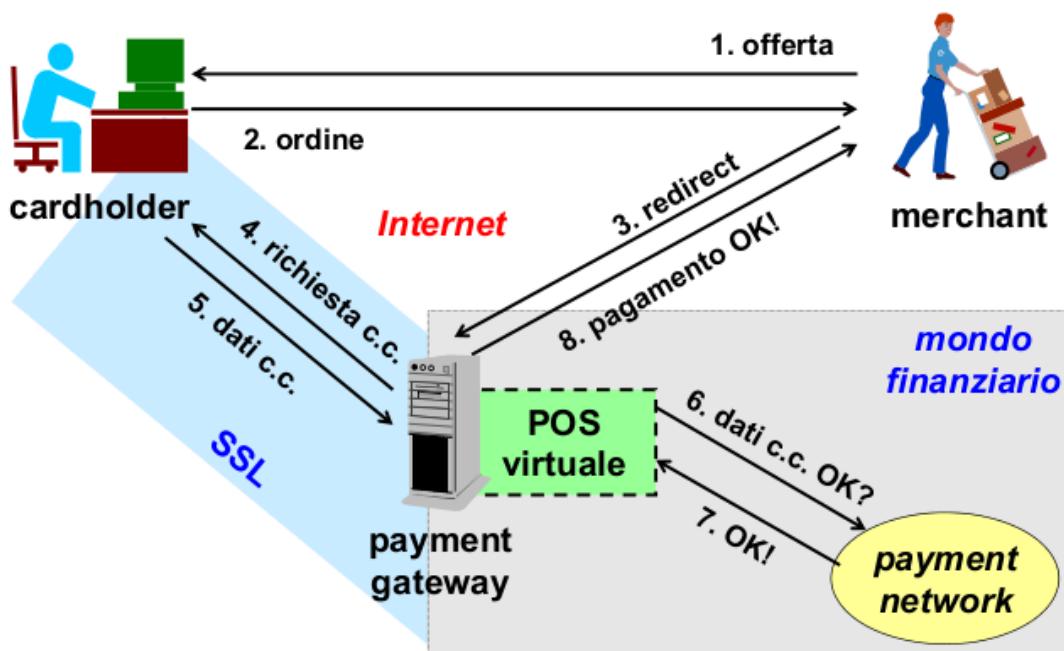
- UdA4 - La sicurezza delle reti
 - Lezione 1 - La sicurezza dei sistemi informativi.
 - La posta elettronica pp.175-176
 - Il protocollo S/MIME per la posta elettronica pp.176-179
 - Un software per la posta sicura: PGP pp.179-182
- [S/MIME](#) - [Wikipedia](#)

Sistemi di pagamento elettronico

L'uso dei **sistemi di pagamento elettronico** tramite carte di credito usando un canale TLS come sistema di protezione sta ormai diventando comune negli acquisti via Web.

L'ente **Payment Card Industry Data Security Standard Council** è l'ente che gestisce il **Payment Card Industry Data Security Standard** (PCI DSS) che è lo standard utilizzato per migliorare i controlli relativi ai dati forniti dal **cardholder**, al fine di ridurre le frodi relative ai pagamenti via Web tramite carte di credito. Ad oggi questo standard viene richiesto per la gestione di tutti i pagamenti via Internet usando la carta di credito, essendo più prescrittivo rispetto ad altre tecniche.

L'immagine seguente mostra l'architettura alla base di queste forme di pagamento.



I pagamenti elettronici nel Web vengono svolti seguendo otto fasi:

1. il **merchant**, ovvero il negoziante online, pubblica un'offerta della merce che vuole vendere;
2. l'offerta attira l'attenzione di un **cardholder**, un possessore di carta di credito, che decide di effettuare un ordine;
3. il **cardholder** viene reindirizzato al **payment gateway** dal sito del **merchant**;
4. tramite **TLS** il **payment gateway** chiede i dati della carta di credito al **cardholder**;
5. il **cardholder** risponde fornendo i dati, sempre usando **TLS**;
6. il **POS virtuale** che si interfaccia tra il mondo di Internet e il mondo finanziario, chiede alla **payment network** se i dati ricevuti dal **cardholder** sono corretti;
7. il **payment network** invia la risposta sulla correttezza o meno dei dati del **cardholder**;

8. se i dati sono corretti si procede con il pagamento, il **merchant** viene informato dell'avvenuto pagamento e può quindi spedire la merce.

Nella configurazione illustrata il **payment gateway** è l'entità che possiede tutte le informazioni sulla transazione, sia quelle relative al pagamento, sia quelle relative alla merce, mentre il **merchant** avrà solo le informazioni relative alla merce, non essendo suo compito validare la bontà delle credenziali fornite dal **cardholder**.

L'ipotesi base affinché un pagamento elettronico vada a buon fine è che l'acquirente possieda una carta di credito e che abbia un browser con TLS. La sicurezza effettiva però dipende dalla configurazione e dalla sicurezza sia del server sia del client, infatti per avere una rete protetta è necessario installare e gestire una configurazione con firewall e non usare password di sistema predefinite o altri parametri di sicurezza impostati dai fornitori dei sistemi di sicurezza. Per proteggere invece i dati dei titolari delle carte di credito è sempre opportuno memorizzarli con attenzione e criptarli quando vengono trasmessi attraverso reti pubbliche aperte. Inoltre bisogna sempre rispettare un programma per la gestione delle vulnerabilità, di conseguenza bisogna usare e aggiornare con regolarità l'antivirus, ma anche sviluppare e mantenere protetti applicazioni e sistemi. In particolar modo è necessario implementare misure forti per il controllo degli accessi (*Authentication*), si deve consentire l'accesso ai dati solo se questo è effettivamente indispensabile per lo svolgimento dell'attività commerciale (*Principle of least privilege* e *Authorization*) e di conseguenza limitare la possibilità di accesso fisico ai dati, chiudendo i server a chiave e introducendo l'uso di un ID univoco per ogni utente del sistema informativo.

Infine bisogna monitorare e testare le reti con regolarità, tenendo traccia di tutti gli accessi effettuati alle risorse della rete e ai dati dei titolari delle carte (*Accounting*), ma anche eseguendo test periodici dei processi e dei sistemi di protezione. Quindi per mantenere un sistema sicuro bisogna stilare l'elenco delle regole che si vuole adottare e soprattutto **osservare rigorosamente le regole impostate dalla politica di sicurezza creata**, senza l'osservanza di quest'ultima importante regola tutte le altre risulterebbero inutili meno.

La sicurezza perimetrale

Prefazione

I seguenti appunti sono basati sulle lezioni del [Prof. Steven Gordon](#) a cui va la mia profonda gratitudine per la pubblicazione delle sue lectures sul suo [canale YouTube](#).

Per l'arricchimento e la stesura finale degli appunti è risultato fondamentale anche il lavoro svolto dalla Prof.ssa Sophia Danesino, mentore, amica, appassionata di conoscenza e insegnante fuori dal comune. Sul suo [canale YouTube](#) è possibile trovare i video su molti degli argomenti trattati.

La sicurezza delle reti: Packet Filtering Firewall

Libro vol.3 - Firewall, Proxy, ACL e DMZ

STUDIARE: L. Lo Russo, E. Bianchi, *Sistemi e Reti - Nuova Edizione OPEN SCHOOL - Per l'articolazione INFORMATICA degli Istituti Tecnici settore Tecnologico*, vol. 3, ed. Hoepli, 2017.

- UdA4 - La sicurezza delle reti
 - Lezione - Firewall, Proxy, ACL e DMZ
 - I firewall pp.191-196

LEGGERE: [Firewall su Wikipedia](#).

Packet Filtering Firewall o stateless firewall

Un [**Packet Filtering Firewall**](#) è un firewall di tipo **stateless** in cui un pacchetto viene analizzato singolarmente solo in base alle regole di filtraggio configurate nel firewall.

Questo sarà il primo tipo di firewall che verrà analizzato in quanto, pur essendo una tecnica ormai superata dallo **Stateful Packet Inspection** perché meno accurato di quest'ultimo, l'analisi delle problematiche che questo tipo di firewall pone permetteranno di comprendere le ragioni che hanno portato allo sviluppo di tecniche più sofisticate, oltre a fornire una maggiore chiarezza del funzionamento della pila protocollare TCP/IP.

Un firewall, per decidere se accettare un PDU o scartarlo, analizzerà gli header dei vari livelli della pila protocollare per capire se le regole di filtraggio possono o meno essere applicate al PDU in esame.

Per stabilire una regola di filtraggio sono necessarie diverse informazioni. Alcune di queste sono recuperabili dagli header dei PDU dei diversi livelli della pila protocollare, mentre le azioni da intraprendere sono invece da impostare.

Relativamente al PDU del **livello IP**, lo header è in grado di fornire le seguenti informazioni utili per operare con delle regole di filtraggio:

- **l'indirizzo IPv4 sorgente;**
- **l'indirizzo IPv4 destinatario;**
- **il protocollo di Trasporto usato durante la comunicazione.**

Gli **indirizzi IPv4 sorgente** e **destinatario** permettono di identificare **chi sta comunicando**, mentre il **campo Protocol** dello header del **livello IP** permette di identificare **quale protocollo di Trasporto viene usato per la comunicazione**. Si ricorda, come esempio, che quest'ultimo campo verrà impostato al valore 6 nel caso venisse usato TCP e al valore 17 nel caso venisse usato UDP, ma a questo [link](#) è possibile analizzare un elenco più esaustivo.

Relativamente al PDU del **livello di Trasporto**, lo header è in grado di fornire le informazioni su **quale applicazione viene usata**, e i campi utili da analizzare per operare con delle regole di filtraggio sono:

- il numero di **porta sorgente**;
- il numero di **porta destinataria**.

Si ricordano, come esempi, alcuni numeri di porte utilizzati dai server: Web - 80, HTTPS - 443, SSH - 22, E-mail - 25, ma a questo [link](#) è possibile analizzare un elenco più esaustivo.

Relativamente all'**azione da intraprendere in una regola di filtraggio** alcuni esempi sono:

- **DROP/DISCARD**: scarta il PDU che rispetta la regola senza comunicare nulla al mittente;
- **ACCEPT/ALLOW**: accetta il PDU che rispetta la regola e, implicitamente scarta qualsiasi altra cosa;
- **DENY**: scarta il PDU che rispetta la regola inviando un messaggio ICMP al mittente che comunichi lo scarto;

ma le azioni sono molteplici e un esaustivo elenco delle possibili azioni varia tra i vari modelli di firewall.

Infine sarà necessario anche stabilire l'**azione di default da intraprendere nel caso il PDU non rispettasse alcuna regola fra quelle configurate nel firewall**.

Sintetizzando, le informazioni necessarie per impostare le regole di filtraggio sono:

1. indirizzo IPv4 sorgente;
2. indirizzo IPv4 destinatario;
3. protocollo di Trasporto;
4. numero di porta sorgente;
5. numero di porta destinataria;
6. azione regola filtraggio;
7. azione di default.

La regola di filtraggio analizzerà il pacchetto usando le prime cinque caratteristiche elencate (punti 1 - 5), se il pacchetto risulterà conforme per ognuno di questi elementi allora il firewall applicherà l'azione di filtraggio stabilita nella regola (punto 6), altrimenti verrà intrapresa l'azione di default (punto 7).

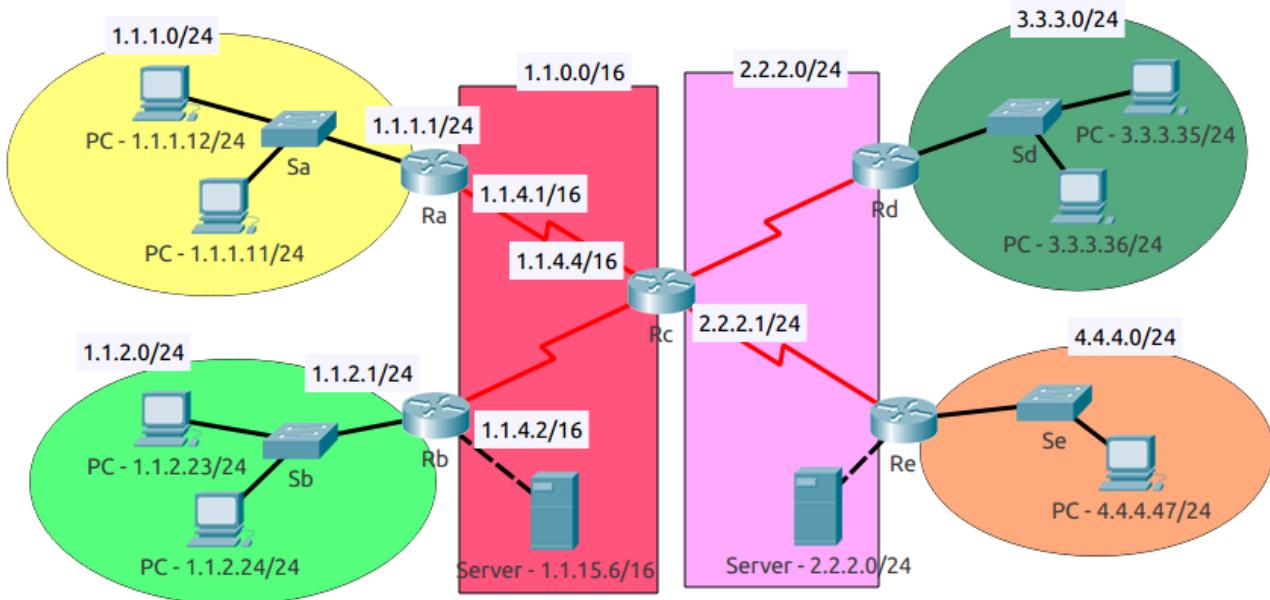
In alcuni di questi elementi sarà **possibile usare dei caratteri jolly**, come lo * che **indica un qualsiasi valore**.

Una regola potrebbe necessitare di ulteriori informazioni, come ad esempio la direzione dei pacchetti (*inside* - il pacchetto arriva dalla rete che si vuole proteggere, *outside* - il pacchetto arriva dall'esterno della rete da proteggere), ma quelle elencate sono per ora sufficienti per analizzare alcuni esempi.

Tutte le regole di filtraggio vengono eseguite seguendo un ordine preciso, diversamente i risultati di filtraggio potrebbero essere imprevedibili.

Esempi di Packet Filtering Firewall

Nei successivi esempi si supporrà di lavorare con la seguente rete e di dover impostare alcune regole di filtraggio dei pacchetti nelle diverse reti o nei singoli PC.



Drop di ICMP Request dall'interno della rete

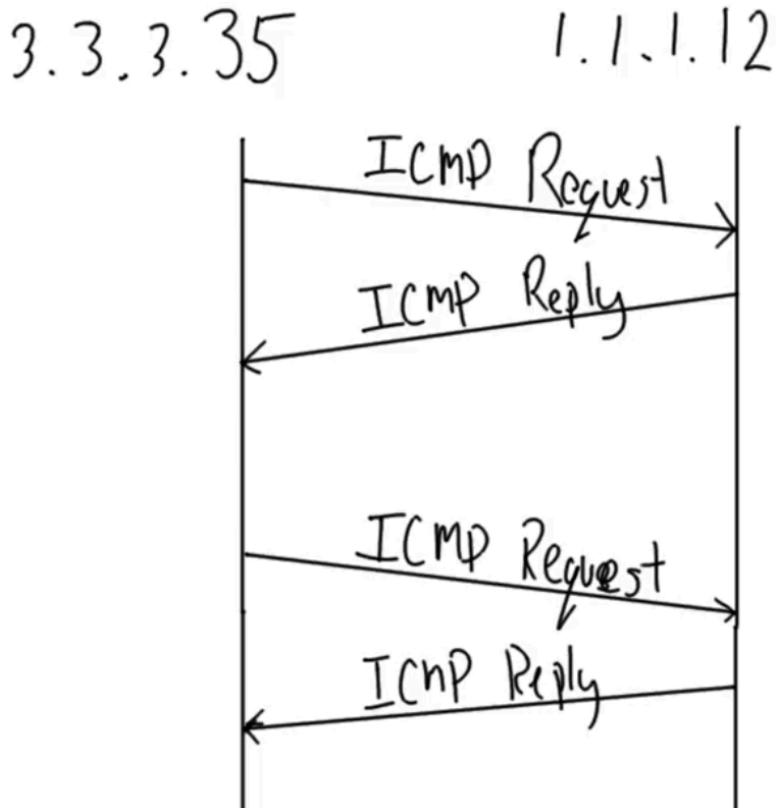
Si supponga di impostare un firewall su ogni singolo computer, e si consideri il **personal firewall del computer 1.1.1.12/24**. Su questo computer si vuole **bloccare il ping request proveniente dal SOLO computer 3.3.3.35/24**. Si imposta la regola di filtraggio appropriata.

1. **indirizzo IPv4 sorgente:** **3.3.3.35/24** in quanto è il computer che invia il ping;
2. **indirizzo IPv4 destinatario:** **1.1.1.12/24** in quanto è il computer destinatario del ping;
3. **protocollo di Trasporto:** ICMP - 1 ([link](#));
4. **numero di porta sorgente:** * che indica qualsiasi valore, dato che per il protocollo ICMP non vengono usati numeri di porta;
5. **numero di porta destinataria:** * che indica qualsiasi valore, dato che per il protocollo ICMP non vengono usati numeri di porta;
6. **azione regola filtraggio:** **DROP**, in questo modo il software che gestisce il ping nel computer 1.1.1.12/24 non riceverà mai la ping request del computer 3.3.3.35/24, e quest'ultimo non verrà avvisato dello scarto del suo ping request;
7. **azione di default:** **ACCEPT**, in questo modo qualsiasi altro tipo pacchetto inviato dal computer 3.3.3.35/24 sarà autorizzato a passare.

Quindi questa regola indicherà che se il computer 1.1.1.12/24 riceverà un ping, il pacchetto verrà scartato senza inviare alcuna notifica, SOLO se proveniente dal computer 3.3.3.35/24. Se il ping provenisse da un diverso computer il pacchetto seguirebbe la regola di default, e quindi sarebbe accettato. Analogamente, sarebbe

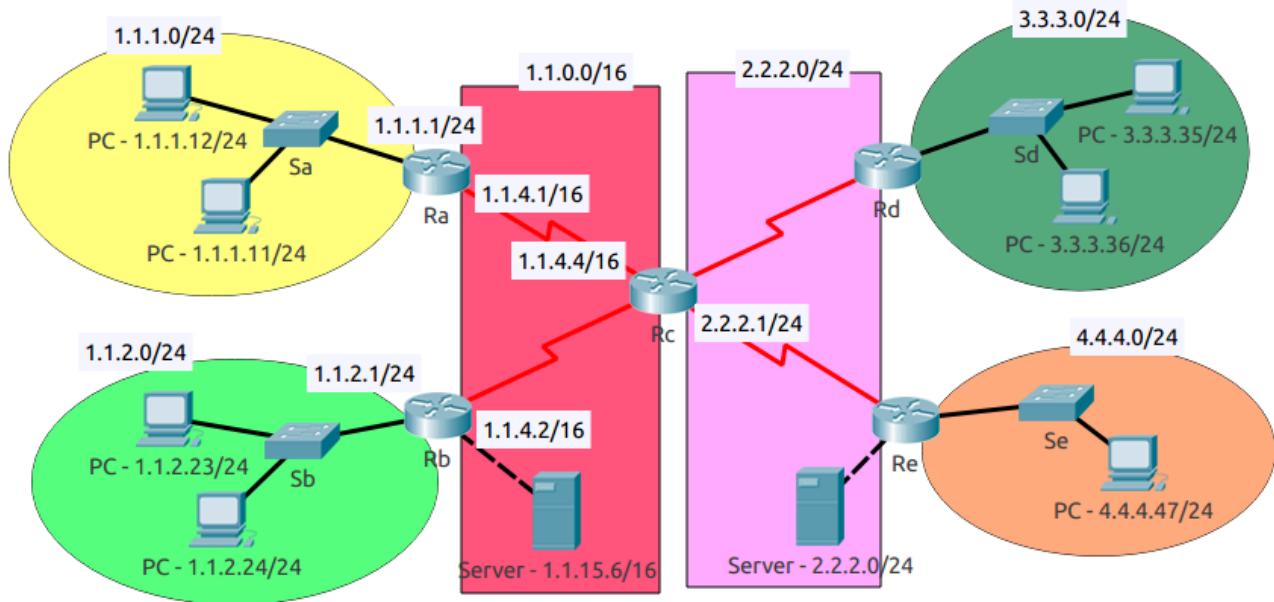
accettata una qualsiasi altra richiesta che usi un protocollo diverso da ICMP proveniente dal computer 3.3.3.35/24.

Si ricordi che lo scambio di pacchetti ping avviene come mostrato nell'immagine seguente. La regola impostata non potrà impedire al computer 3.3.3.35/24 di inviare un pacchetto di ping request (ICMP Request), ma eviterà che il computer 1.1.1.12/24 invii il ping reply (ICMP Reply), in quanto il firewall intercetterà e scarterà il pacchetto prima che il software che gestisce il ping lo riceva.



Drop di connessioni SSH dall'esterno della rete

Nel presente esempio si supporrà di impostare il **firewall nel router Ra** che risulta connesso alle due reti **1.1.0.0/16** e **1.1.1.0/24**, quest'ultima sottorete della precedente.



Il router **Ra** avrà due interfacce, una per ogni rete, con indirizzi IPv4 diversi, la prima con indirizzo **1.1.4.1/16** collegata alla rete **1.1.0.0/16** e la seconda con indirizzo **1.1.1.1/24** collegata alla sottorete **1.1.1.0/24**.

Si supponga che la **sottorete 1.1.1.0/24** sia **la rete che il firewall deve proteggere** e che il **computer 1.1.1.11/24** abbia un **server SSH** al quale **potranno collegarsi SOLO i computer interni alla sottorete**, mentre tutti gli **altri computer esterni alla sottorete non dovranno avere la possibilità di collegarsi**. Si imposta la regola di filtraggio appropriata.

1. **indirizzo IPv4 sorgente**: * perché qualsiasi computer esterno alla sottorete non deve avere la possibilità di connettersi al server SSH, considerando anche che qualsiasi richiesta interna alla sottorete non sarà inviata al router Ra, ma verrà inviata direttamente al server SSH, quindi il firewall non riceverà questo tipo di richiesta³⁰;
2. **indirizzo IPv4 destinatario**: **1.1.1.11/24** perché è il computer che contiene il server SSH;
3. **protocollo di Trasporto**: TCP - 6 ([link](#)) in quanto SSH utilizza a livello di trasporto il protocollo TCP³¹;
4. **numero di porta sorgente**: * in quanto il client SSH riceverà un numero di porta dinamico dal proprio sistema operativo e quindi questo valore non è predicibile;

³⁰ Questo elemento potrebbe essere più preciso indicando come indirizzo IPv4 sorgente qualsiasi indirizzo non appartenente alla sottorete 1.1.1.0/24, ma per questo esempio la scelta dello * è accettabile.

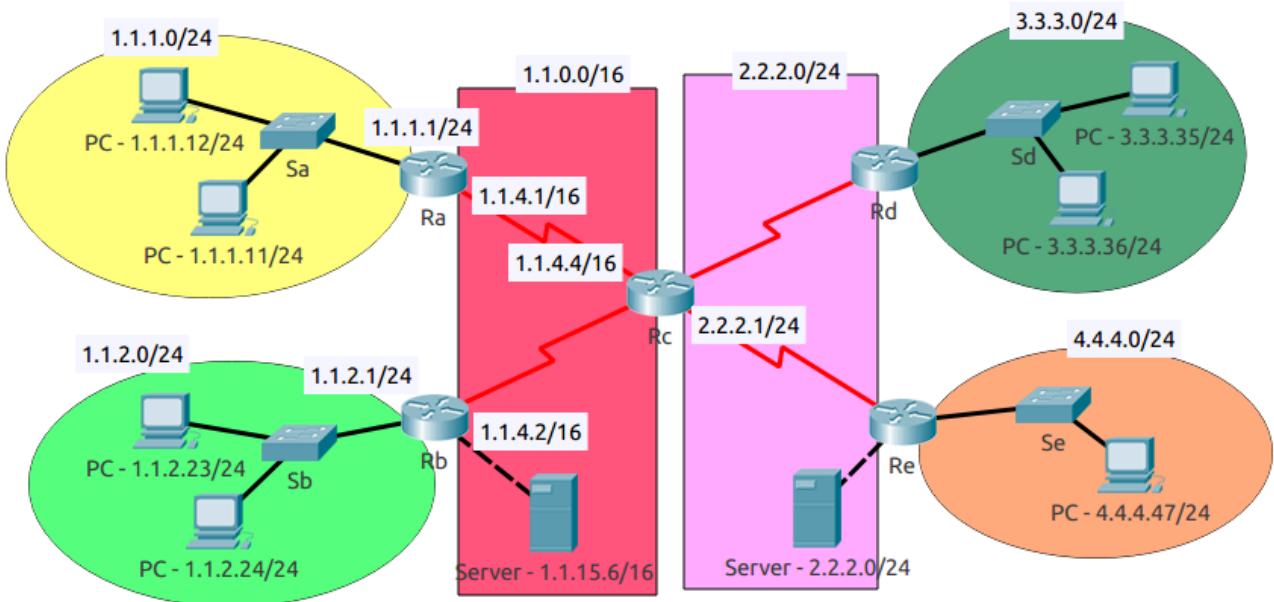
³¹ In realtà il protocollo SSH ammette anche l'uso di UDP, ma è particolarmente raro come caso. Se si desidera essere più precisi nella regola si potrebbe indicare *.

5. **numero di porta destinataria:** server SSH - **22** ([link](#)) che corrisponde al numero di porta del server SSH;
6. **azione regola filtraggio: DROP**, in questo modo il server SSH sul computer 1.1.1.11/24 non riceverà mai la richiesta di connessione da parte di un qualsiasi computer esterno alla sottorete, e il firewall non notificherà l'avvenuto scarto;
7. **azione di default: ACCEPT**, in questo modo qualsiasi altro tipo di pacchetto proveniente dall'esterno che non rispetti almeno uno dei primi cinque elementi indicati sarà ammesso.

Quindi questa regola bloccherà qualsiasi tentativo di connessione al server SSH proveniente dall'esterno della sottorete 1.1.1.0/24, permettendo però l'accesso a qualsiasi altro pacchetto che non voglia connettersi al server SSH. I computer interni alla sottorete 1.1.1.0/24 non saranno analizzati dal firewall in quanto indirizzati direttamente al server SSH, non venendo quindi influenzati dalla regola impostata.

Drop di connessioni a Web esterno alla rete

Nel presente esempio si supporrà di impostare il **firewall nel router Ra** e in questo caso **il firewall dovrà impedire al computer 1.1.1.12/24 di connettersi, utilizzando un browser, ad un qualsiasi Web server presente nella rete 3.3.3.0/24.**



Si imposta la regola di filtraggio appropriata.

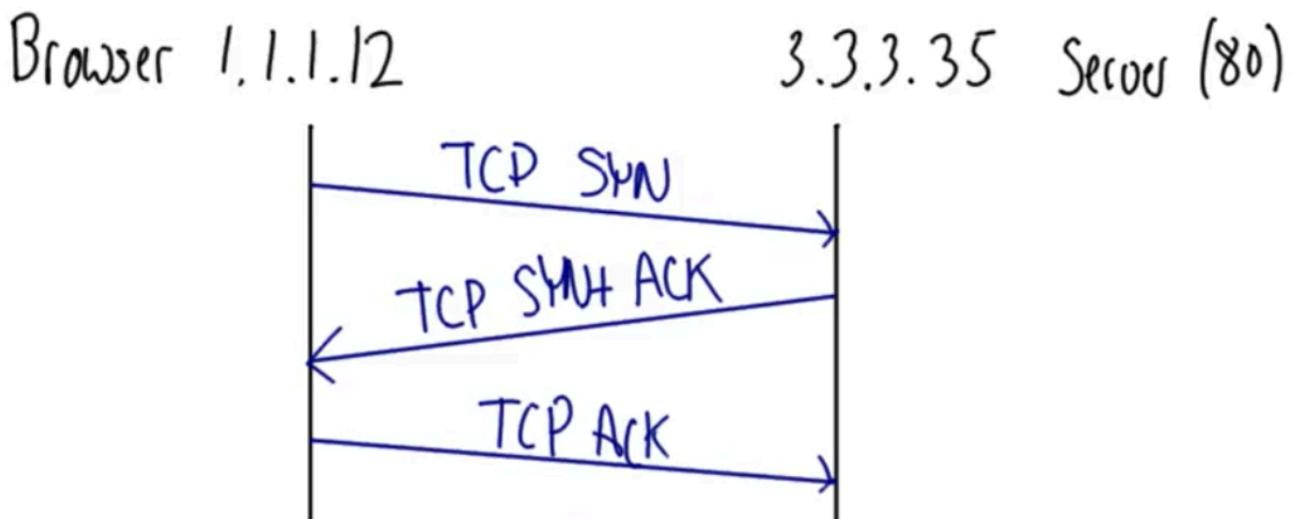
1. **indirizzo IPv4 sorgente:** **1.1.1.12/24** in quanto si vuole impedire solo a questo computer di accedere ad un qualsiasi Web server della rete 3.3.3.0/24;
2. **indirizzo IPv4 destinatario:** **3.3.3.0/24** in quanto si vuole impedire l'accesso a qualsiasi Web server di questa rete e l'uso dell'indirizzo di rete è sufficientemente generale da permetterci di individuarli con un'unica regola³²;
3. **protocollo di Trasporto:** TCP - **6** ([link](#)) in quanto HTTP utilizza a livello di Trasporto il protocollo TCP;
4. **numero di porta sorgente:** * in quanto il client HTTP riceverà un numero di porta dinamico dal proprio sistema operativo e quindi questo valore non è predicibile;
5. **numero di porta destinataria:** HTTP - **80** ([link](#)) in quanto un Web server usa di solito questo numero di porta;
6. **azione regola filtraggio:** **DROP**, in questo modo qualsiasi richiesta HTTP inviata dal computer 1.1.1.12/24 e diretta alla rete 3.3.3.0/24 verrà direttamente scartata dal firewall, e quindi non raggiungerà mai la rete 3.3.3.0/24;
7. **azione di default:** **ACCEPT**, in questo modo qualsiasi altro tipo di pacchetto proveniente dal computer 1.1.1.12/24 che non rispetti almeno uno dei primi cinque elementi indicati sarà ammesso.

³² Sarebbe stato possibile scrivere più regole di filtraggio, una per ogni Web server. La scelta dipenderà da cosa si vorrà regolare.

Quindi questa regola bloccherà qualsiasi richiesta HTTP inviata dal computer 1.1.1.12/24 verso la rete 3.3.3.0/24, richiesta che a livello di Trasporto usa il protocollo TCP. Dato che il firewall impedirà alla richiesta HTTP di fuoriuscire dalla rete 1.1.1.0/24, un Web server della rete 3.3.3.0/24 non la riceverà mai e, di conseguenza, il computer 1.1.1.12/24 non riceverà neanche la relativa HTTP response. La regola di filtraggio permette comunque di raggiungere un Web server presente nella rete 3.3.3.0/24 tramite un altro protocollo, come ad esempio SSH.

Questo esempio merita un'analisi accurata per comprendere effettivamente quale tipo di messaggio viene bloccato nello scambio HTTP.

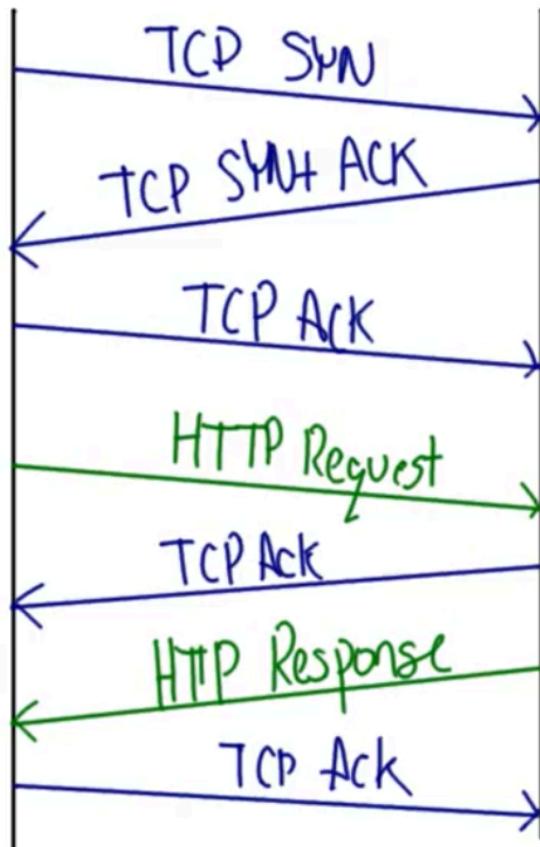
Se la regola di filtraggio non fosse impostata, la connessione tra un browser e un Web server prevederebbe inizialmente la creazione di una connessione TCP tra il client ed il server, quindi il primo PDU inviato dal client al server sarebbe un segmento di richiesta di sincronizzazione da parte del client, cioè un segmento di tipo SYN. Il Web server confermerebbe il segmento SYN inviato dal client, e al contempo manifesterebbe la sua volontà di sincronizzarsi, il tutto utilizzando un segmento di tipo SYN-ACK. Infine il client confermerebbe la ricezione del segmento inviato dal server usando un segmento ACK. Questa procedura permette di stabilire una connessione TCP ed è conosciuta come *three-way handshake*.



Una volta stabilita la connessione verrebbero inviati i dati, che nel nostro caso sarebbe una HTTP Request dal client al server. Il server confermerebbe la HTTP Request e invierebbe la HTTP Response, solitamente una pagina Web, a sua volta confermata dal client.

Browser 1.1.1.12

3.3.3.35 Server (80)



Abilitando il firewall e analizzando la regola impostata in questo esempio, si deduca quale pacchetto sarebbe scartato nella sequenza di scambi di segmenti appena illustrata³³.

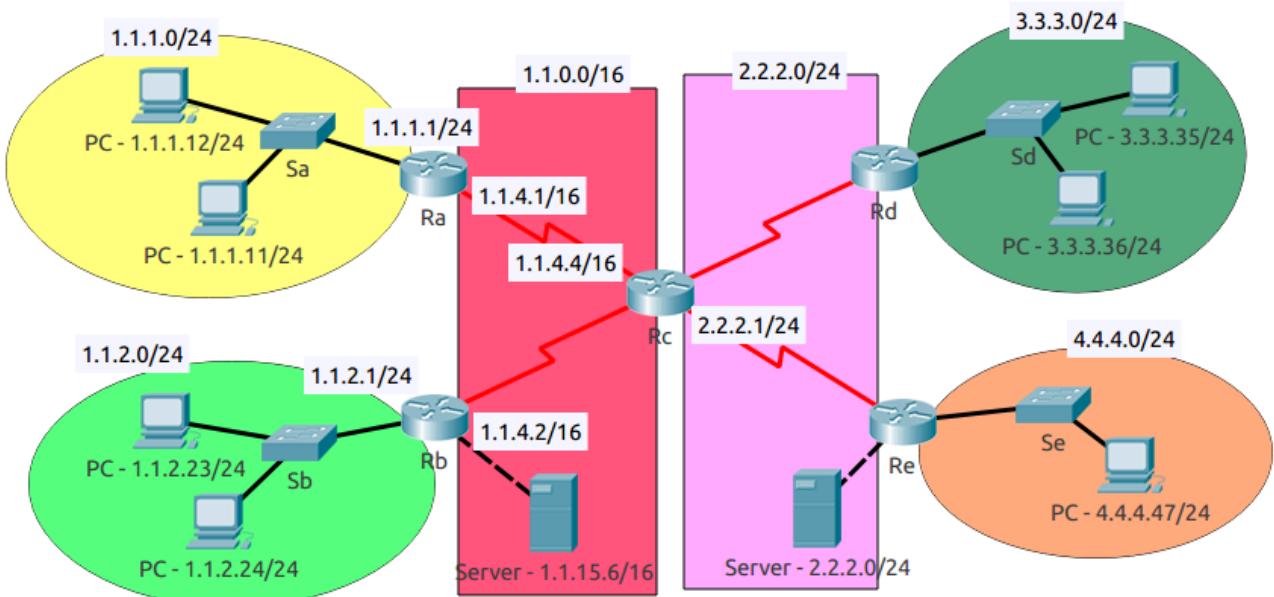
La regola impostata in questo esempio è aggirabile in molteplici modi. Si individuino le diverse modalità per aggirare la regola e le possibili contromisure da attuare sul firewall per prevenire che ciò avvenga³⁴.

³³ Verrebbe scartato il primo segmento SYN inviato dal client al server del *three-way handshake*. Questo avverrebbe anche per la connessione SSH illustrata nell'esempio precedente, dato che questo protocollo usa TCP a livello di Trasporto.

³⁴ **1.** Utilizzando una VPN, o un proxy server o un altro tipo di server a cui collegarsi in qualche modo, Questo metodo richiede però la presenza di un server VPN disponibile all'altra estremità o la possibilità di far agire un server remoto diverso come si desidera. **2.** Cambiando l'indirizzo IPv4 sorgente usandone un altro della rete 1.1.1.0/24 dato che, in questo caso, è stato bloccato solo il computer 1.1.1.12/24, ma chiaramente la regola potrebbe essere modificata bloccando tutti gli indirizzi IPv4 della rete. **3.** Utilizzando un indirizzo IPv4 sorgente fake, ad esempio l'indirizzo 7.8.3.4/24, ma anche in questo caso è possibile introdurre una regola di filtraggio che indichi di bloccare tutti gli indirizzi IP fake, e cioè tutti gli indirizzi che non appartengono alla rete 1.1.1.0/24. **4.** Utilizzando un protocollo di Trasporto diverso, come UDP, ma questo presuppone che il server gestisca il protocollo HTTP utilizzando anche UDP cosa molto improbabile, e comunque si potrebbe aggiungere una regola di filtraggio che blocchi HTTP su UDP. **5.** Utilizzando un numero di porta destinataria diverso, come ad esempio HTTPS - 443. Se il Web server fosse configurato per gestire HTTPS i pacchetti potrebbero aggirare la regola di filtraggio. Nel firewall sarebbe necessario introdurre un'ulteriore regola di filtraggio che impedisca l'accesso anche tramite HTTPS.

Accept delle sole connessioni a Web esterno alla rete

Nel presente esempio si supporrà di impostare il **firewall nel router Ra** e in questo caso **il firewall dovrà permettere al computer 1.1.1.12/24 di connettersi, utilizzando un browser, ad un qualsiasi Web server presente nella rete 3.3.3.0/24, ma di default il firewall dovrà impedire qualsiasi altra forma di comunicazione con l'esterno.**

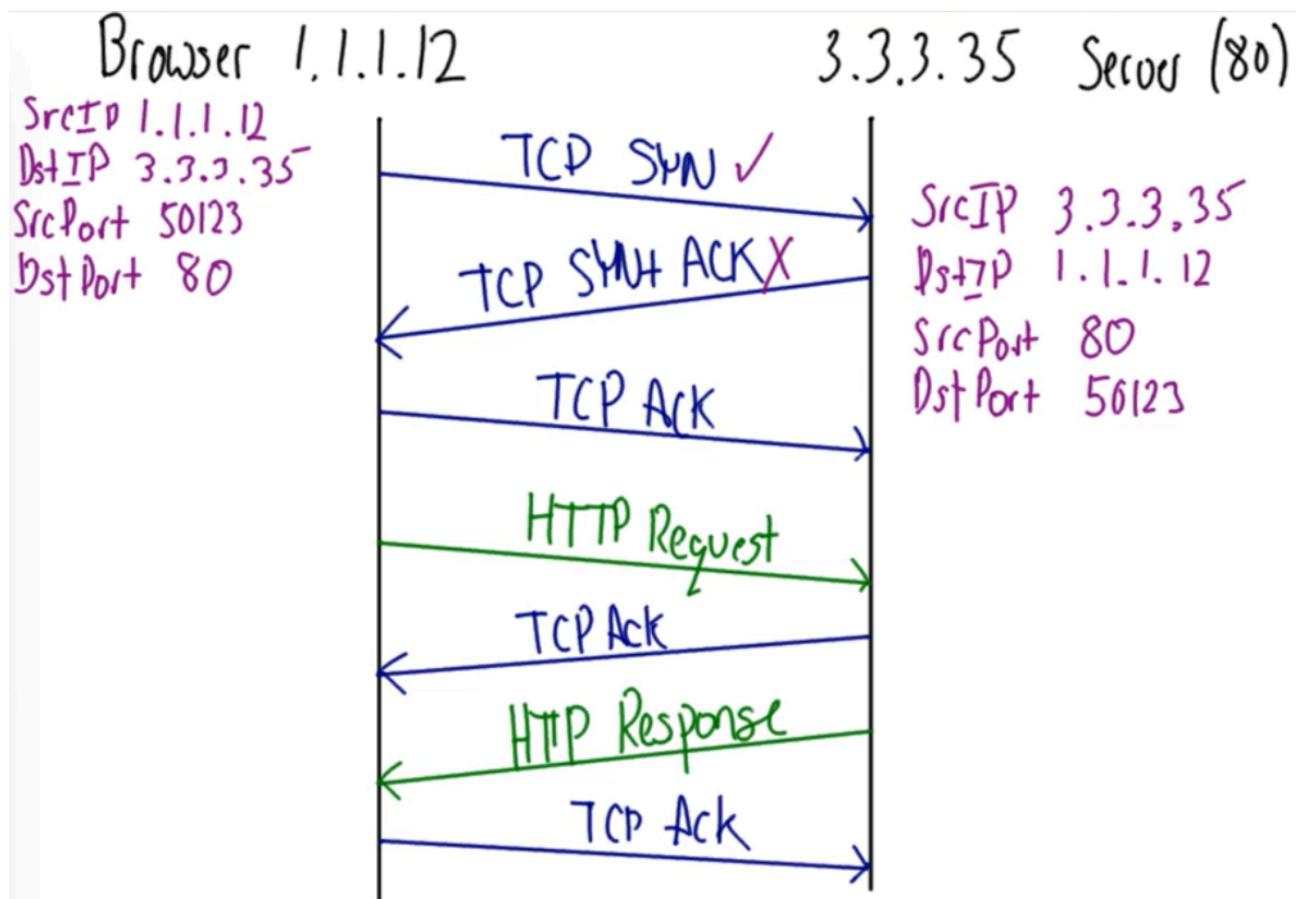


Questo esempio risulta più complicato dei precedenti e richiederà l'impostazione di più regole. Si porvi inizialmente a risolverlo senza guardare la soluzione, poi si proceda confrontando la propria soluzione con quanto proposto di seguito.

Probabilmente la prima soluzione trovata risulterà simile al precedente esempio, con qualche ovvia variazione, come illustrato di seguito:

SrcIP	DstIP	Prot.	SrcPort	DstPort	Action
1.1.1.12/24	3.3.3.0/24	6	*	80	ACCEPT
Default action:		DROP			

Questa soluzione però **non risulterà funzionante** perché, mentre il segmento SYN inviato dal client per avviare il *three-way handshake* risponderà esattamente alla regola di filtraggio, riuscendo quindi a raggiungere il Web server, il segmento SYN-ACK inviato da server verrà scartato in quanto non rispetterà la regola di filtraggio, e quindi la connessione non verrà stabilita.



Per permettere la riuscita della comunicazione **sarà quindi necessario aggiungere una seconda regola di filtraggio**, per consentire lo scambio sia dal client al server, sia dal server al client, come mostrato di seguito:

SrcIP	DstIP	Prot.	SrcPort	DstPort	Action
1.1.1.12/24	3.3.3.0/24	6	*	80	ACCEPT
3.3.3.0/24	1.1.1.12/24	6	80	*	ACCEPT
Default action:		DROP			

L'ordine di esecuzione delle regole di filtraggio segue esattamente quello indicato, e quindi il firewall tenterà di applicare la prima regola di filtraggio, ma se un pacchetto non la rispettasse, il firewall cercherà di applicare la seconda regola. Se nessuna delle regole di filtraggio elencate fosse rispettata allora verrebbe applicata la regola di default.

Si verifichi individualmente l'effettivo funzionamento delle regole indicate nella tabella del firewall per l'esempio proposto.

Considerazioni sull'azione di default

La corretta impostazione di un firewall in una situazione reale richiede l'impostazione di molte regole ed è opportuno che venga effettuata con attenzione dall'amministratore per non ritrovarsi con un firewall non funzionante correttamente, o con delle violazioni di sicurezza, o altri problemi con gli utenti finali.

Spesso **l'impatto degli errori di configurazione** di un firewall **dipende dall'azione di default impostata**. È opportuno soffermarsi quindi a considerare le differenze nell'impostare come azione di default una **ACCEPT** o un **DROP**.

Impostando come **azione di default ACCEPT** il firewall permette qualsiasi cosa tranne ciò che è esplicitamente vietato dalle regole di filtraggio. Questa azione di default **ha senso in una situazione in cui le operazioni da vietare sono poche**, come ad esempio nel caso di un'utenza domestica. In ogni caso **questa impostazione risulta pericolosa in termini di sicurezza** in quanto, se venisse fatto un errore nella configurazione delle regole del firewall, un attaccante potrebbe accedere alla rete interna proprio perché di default viene accettato tutto. Inoltre non è possibile predire quali azioni potrebbe fare un attaccante, e quindi potrebbe essere **difficile impostare le regole necessarie per proteggere una rete da intrusioni esterne in modo puntuale**. Sicuramente **questa regola di default permette di impostare regole di filtraggio più semplici**.

Impostando come **azione di default DROP** il firewall vieterà qualsiasi cosa tranne ciò che è esplicitamente permesso dalle regole di filtraggio. Questa azione di default **risulta più sicura rispetto al concetto di sicurezza** in quanto vietare tutto e permettere solo azioni mirate consente di fatto di blindare la rete che si vuole proteggere. **Anche in caso di errori** di configurazione delle regole di filtraggio, **l'azione di default permetterà di proteggere la rete**, eventualmente scartando anche dei pacchetti che invece dovevano passare, ma almeno la sicurezza verrebbe mantenuta, a meno di colossali errori nelle regole di filtraggio per permettere il passaggio dei messaggi. Risulterà comunque **più complessa l'impostazione delle regole di filtraggio**, che dovranno prevedere un'attenta analisi delle diverse fasi di comunicazione di ciò che si vuole permettere, al fine di prevedere esattamente e solo gli scambi pianificati.

La sicurezza delle reti: Stateful Firewall

Libro vol.3 - Firewall, Proxy, ACL e DMZ

STUDIARE: L. Lo Russo, E. Bianchi, *Sistemi e Reti - Nuova Edizione OPENSCHOOL - Per l'articolazione INFORMATICA degli Istituti Tecnici settore Tecnologico*, vol. 3, ed. Hoepli, 2017.

- UdA4 - La sicurezza delle reti
 - Lezione - Firewall, Proxy, ACL e DMZ
 - Stateful inspection pp.196-197

LEGGERE: Firewall su [Wikipedia](#).

Stateful Firewall

Le regole di filtraggio impostate nell'ultimo esempio risultano aggirabili da un attaccante in quanto la seconda regola di fatto permette a qualsiasi applicazione che si trovi nella rete 3.3.3.0/24, e che usi la porta 80, di contattare il computer 1.1.1.12/24 su qualsiasi numero di porta, e questo è decisamente un problema di sicurezza.

Usando il **Packet Filtering Firewall** la regola incriminata è però necessaria per permettere l'instaurazione di una connessione TCP; senza di essa infatti, come si è visto, verrebbe bloccato il segmento SYN+ACK inviato dal server per instaurare una connessione TCP. Questa regola permette a qualsiasi applicazione esterna di penetrare la rete che si sta tentando di proteggere, semplicemente usando come numero di porta sorgente di una applicazione la porta 80.

Per ovviare a questo problema la successiva generazione di firewall ha implementato lo **Stateful Packet Inspection**. Questa tecnica di filtraggio si basa sul concetto di **instaurazione di una sessione** che è costituita da **due unità comunicanti individuate** in modo preciso da **indirizzo IP sorgente e destinatario, numero di porta sorgente e destinatario**, senza alcuna ambiguità sulla porta del client, e **protocollo del livello di Trasporto**. Per le connessioni TCP vengono anche gestiti i numeri di sequenza dei pacchetti e i flag che individuano lo stato della connessione.

Uno **Stateful Firewall** permette di creare la comunicazione tra client e server, se ammisible dalle regole impostate, e qualsiasi messaggio successivo appartenente alla stessa comunicazione viene automaticamente accettato, non necessitando quindi di una regola diversa per gestire le diverse direzioni della comunicazione.

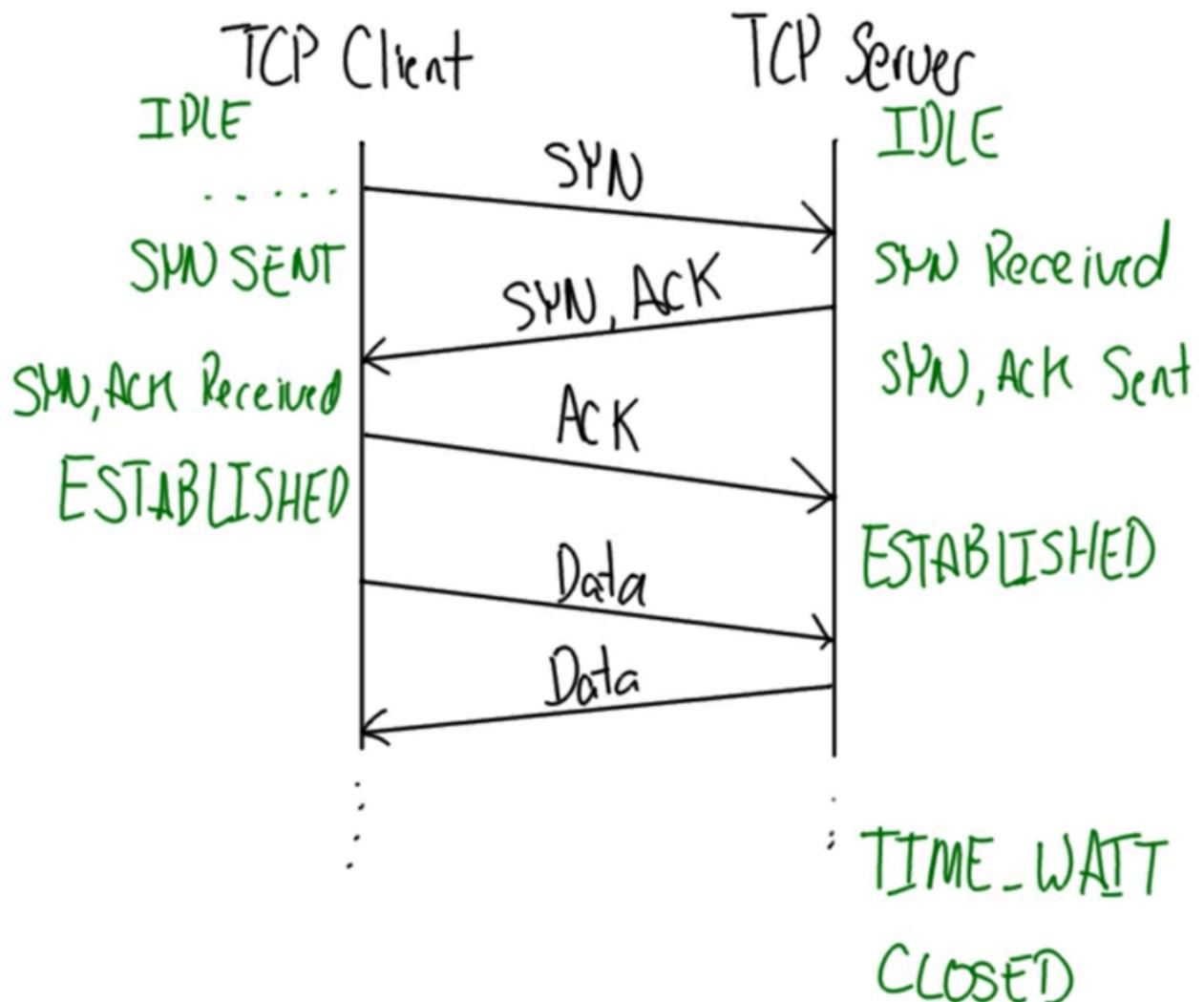
In pratica in uno **stateful firewall** viene impostata una sola regola per risolvere il problema posto nell'esempio precedente:

SrcIP	DstIP	Prot.	SrcPort	DstPort	Action
1.1.1.12/24	3.3.3.0/24	6	*	80	ACCEPT
Default action:		DROP			

senza dover impostare anche la regola per la direzione opposta, in quanto **il firewall memorizza in una tabella separata la sessione in corso** per qualsiasi client verso qualsiasi server della rete 3.3.3.0/24, ammettendo solo i pacchetti esterni che riportano esattamente tutti e solo i parametri della sessione, inclusa la porta precisa del client. Con questa regola inoltre sono ammessi solo i pacchetti che viaggiano dalla rete da proteggere verso l'esterno, e nessun messaggio che giunga dall'esterno verrà ammesso nella rete, a meno che non faccia parte di una sessione di comunicazione registrata dal firewall.

Nello **Stateful Packet Inspection** lo stato di una comunicazione viene memorizzato all'interno della **SPI Table** (*Stateful Packet Inspection Table*).

Gli stati utili per una connessione TCP sono sintetizzati nella seguente immagine:



Facendo riferimento all'esempio precedente, dopo l'invio del pacchetto contenente il segmento SYN del protocollo TCP, il firewall crea la seguente riga nella sua **SPI Table**:

SrcIP:SrcPort	DstIP:DstPort	Prot.	State
1.1.1.12:57640	3.3.3.35:80	6	SYN Received

in quanto dal punto di vista del client 1.1.1.12:57640 lo stato della connessione sarà *SYN Sent*, ma dal punto di vista del router Ra, se il pacchetto è passato, lo stato dovrà essere *SYN Received* in quanto il server lo avrà ricevuto.

Lo stato della connessione cambierà nel corso del tempo e, al termine del *three-way handshake* lo stato cambierà in *ESTABLISHED*:

SrcIP:SrcPort	DstIP:DstPort	Prot.	State
1.1.1.12:57640	3.3.3.35:80	6	ESTABLISHED

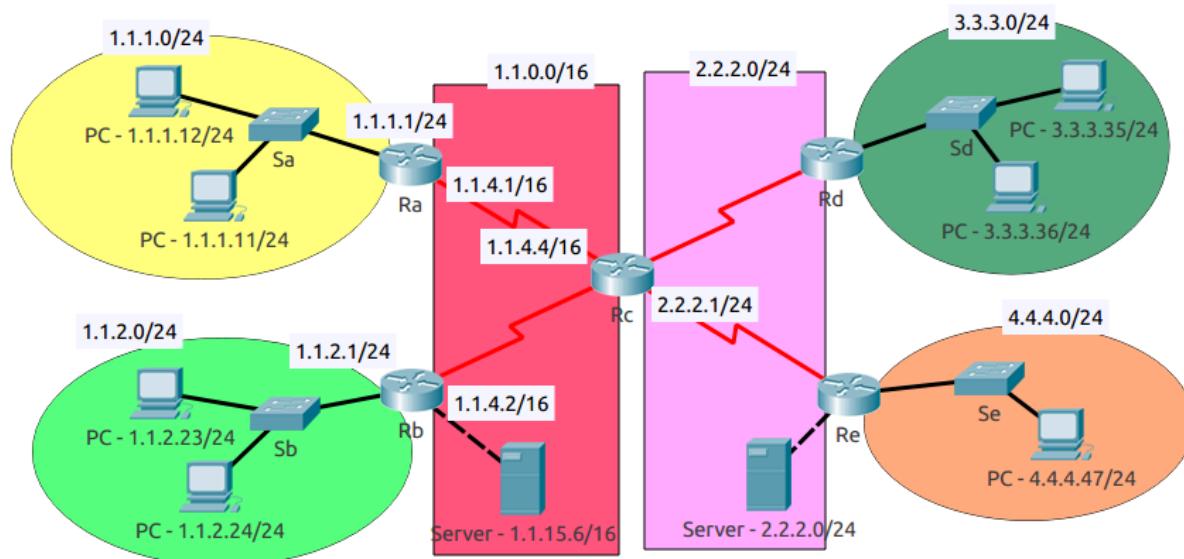
Lo **stateful firewall** per decidere se un pacchetto può o meno passare controllerà prima di tutto se appartiene ad una **sessione** presente nella **SPI Table**, e solo se non troverà alcuna corrispondenza in questa tabella andrà ad esaminare le regole di filtraggio.

Alla chiusura della connessione lo stato diventerà *CLOSED* e i pacchetti non saranno più automaticamente autorizzati usando la **SPI Table**, ma dovranno invece essere analizzati attraverso le regole di filtraggio del firewall.

Le regole di filtraggio vengono definite dall'amministratore della rete, mentre la **SPI Table** viene automaticamente gestita dal software del firewall.

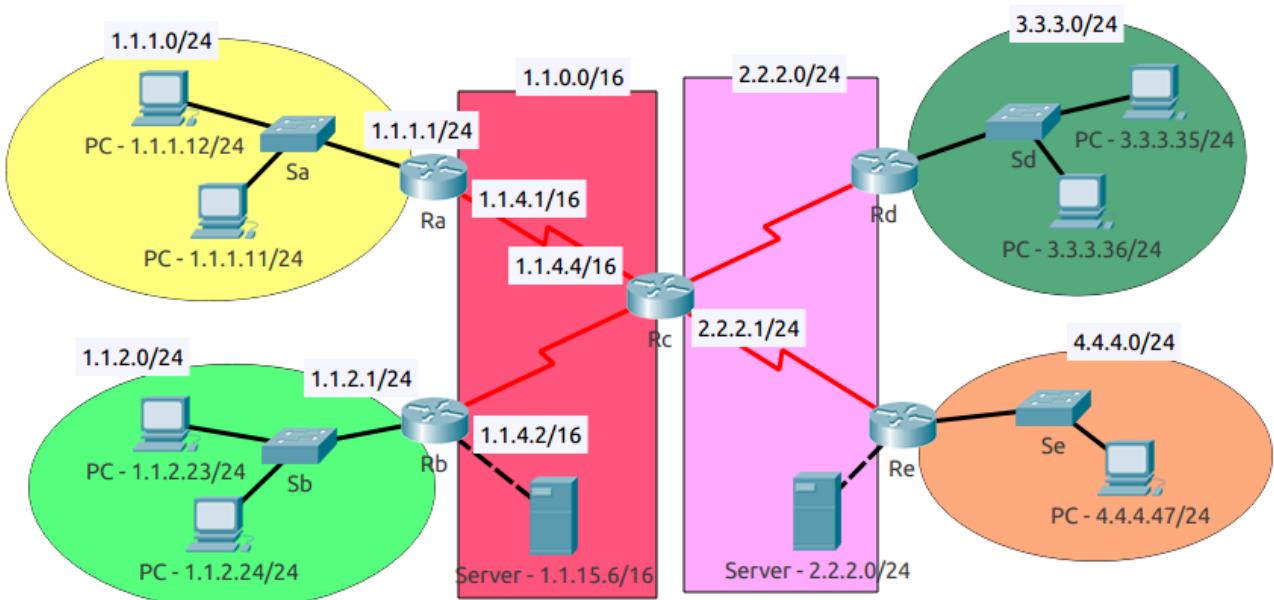
Esempi di Stateful Packet Filtering

Nei successivi esempi si supporrà di lavorare con la seguente rete e di dover impostare alcune regole di filtraggio dei pacchetti nelle diverse reti o nei singoli PC.



Accept delle sole connessioni a Web esterno alla rete

Nel presente esempio si supporrà di impostare il **firewall nel router Ra** e questo **dovrà permettere al computer 1.1.1.12/24 di connettersi, utilizzando un browser, ad un qualsiasi Web server presente nella rete 3.3.3.0/24**, ma di default il firewall dovrà impedire qualsiasi altra forma di comunicazione con l'esterno.



Questo esempio è esattamente quello visto usando il Packet Filtering Firewall, ma in **questo caso lo analizzeremo utilizzando uno Stateful Packet Filtering**, ricordando che verrà utilizzata la **SPI Table** (Stateful Packet Inspection Table) per mantenere traccia delle comunicazioni attive.

La **regola** da impostare **nello stateful firewall** risulterà la seguente:

SrcIP:SrcPort	DstIP:DstPort	Protocol	Action
1.1.1.12:*	3.3.3.0/24:80	6	ACCEPT
Default action:		DROP	

Supponiamo che il computer 1.1.1.12 cerchi di comunicare con un Web server nella rete 3.3.3.0/24, ad esempio quello con indirizzo 3.3.3.36.

L'invio del pacchetto contenente il segmento SYN del protocollo TCP porterà il firewall a controllare prima di tutto la SPI Table per vedere se il pacchetto appartiene ad una connessione già esistente. Dato che non vi è alcuna informazione relativamente a questa connessione nella SPI Table, verranno ispezionate le regole del firewall e, di conseguenza, il pacchetto sarà accettato.

Subito dopo il firewall che si trova nel router Ra memorizzerà nella **SPI Table** lo stato attuale della connessione:

SrcIP:SrcPort	DstIP:DstPort	Prot.	State
1.1.1.12:48037	3.3.3.36:80	6	SYN Received

Quando il server 3.3.3.36:80 riceverà il segmento SYN, invierà come risposta un pacchetto contenente il segmento SYN+ACK del protocollo TCP. Il firewall controllerà se nella SPI Table esista una riga riferita alla connessione, ed essendoci, anche se gli indirizzi IPv4 risulteranno scambiati, riconoscerà l'appartenenza del pacchetto alla comunicazione registrata e cambierà lo stato nella **SPI Table** in *SYN+ACK Received*:

SrcIP:SrcPort	DstIP:DstPort	Prot.	State
1.1.1.12:48037	3.3.3.36:80	6	SYN+ACK Received

A questo punto il client 1.1.1.12:48037 invierà un pacchetto contenente il segmento ACK del protocollo TCP. Il firewall nel router Ra riconoscerà l'appartenenza del pacchetto alla comunicazione memorizzata nella **SPI Table** e lo farà passare, modificando lo stato della connessione in *ESTABLISHED*:

SrcIP:SrcPort	DstIP:DstPort	Prot.	State
1.1.1.12:48037	3.3.3.36:80	6	ESTABLISHED

Qualsiasi pacchetto successivo appartenente alla comunicazione stabilita e che abbia una combinazione dei parametri che la identificano, passeranno attraverso il firewall senza che vengano controllate le regole di filtraggio.

Nel momento in cui lo stato della comunicazione passerà definitivamente a *CLOSED*, la riga della comunicazione verrà cancellata dalla SPI Table e qualsiasi altro pacchetto successivo contenente una combinazione dei parametri identificativi della comunicazione sarà analizzato dal firewall attraverso le regole di filtraggio.

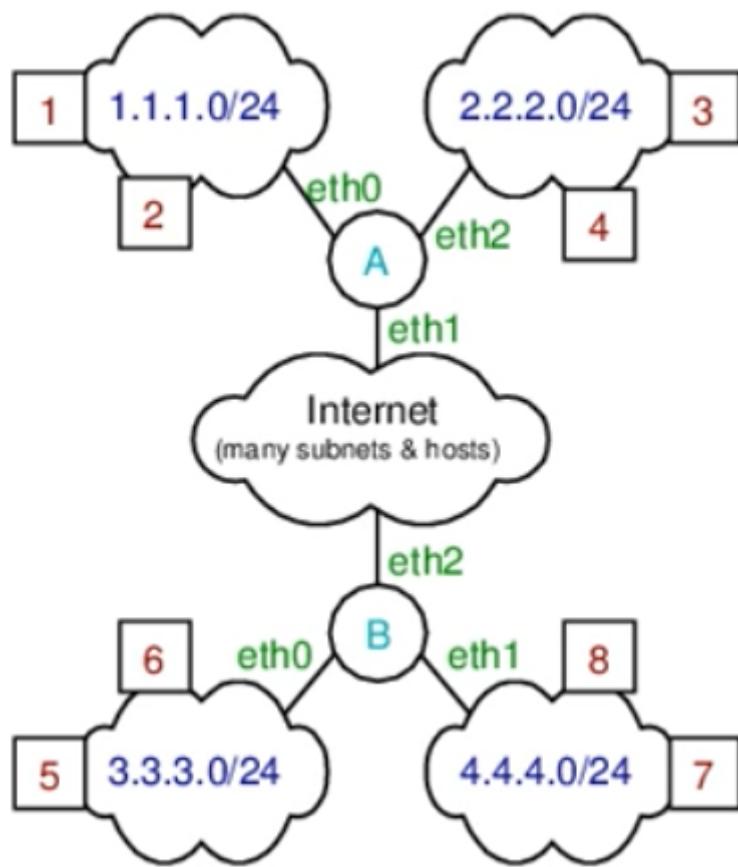
L'uso del comando netstat, in ambiente Linux, permette di vedere lo stato delle connessioni:

```
mgm@umgm:~$ netstat -t -n
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address          Foreign Address        State
tcp      444      0 192.168.64.111:42520  192.168.64.4:139    ESTABLISHED
tcp      1        0 192.168.64.111:40360  91.189.94.25:80     CLOSE_WAIT
tcp      0        0 192.168.64.111:50034  216.58.198.37:443   TIME_WAIT
tcp      32       0 192.168.64.111:46290  162.125.66.3:443   CLOSE_WAIT
tcp      1        0 192.168.64.111:48140  34.197.172.44:443  CLOSE_WAIT
tcp      0        0 192.168.64.111:41852  216.58.198.33:443   ESTABLISHED
tcp      0        0 192.168.64.111:58058  216.58.198.35:443   ESTABLISHED
tcp      0        0 192.168.64.111:57588  162.125.18.133:443  ESTABLISHED
tcp      44       0 192.168.64.111:42760  192.168.64.4:139    ESTABLISHED
tcp      0        0 192.168.64.111:35514  35.161.168.59:443   ESTABLISHED
tcp      32       0 192.168.64.111:42538  162.125.34.6:443   CLOSE_WAIT
```

tcp	0	0	192.168.64.111:50608	66.102.1.189:443	ESTABLISHED
tcp	32	0	192.168.64.111:43128	162.125.66.4:443	CLOSE_WAIT
tcp	0	0	192.168.64.111:50606	66.102.1.189:443	ESTABLISHED
tcp	0	0	192.168.64.111:60342	216.58.198.46:443	ESTABLISHED
tcp	0	0	192.168.64.111:57596	162.125.18.133:443	ESTABLISHED
tcp	32	0	192.168.64.111:57168	162.125.66.3:443	CLOSE_WAIT

Analisi di pacchetti

Si supponga di avere la seguente topologia nella quale il firewall Stateful Packet Filtering è posto nel router A. Le reti che si vogliono proteggere sono la 1.1.1.0/24 e la 2.2.2.0/24.



Le regole di filtraggio impostate nel firewall sono le seguenti:

Rule	SrcIP:SrcPort	DstIP:DstPort	Protocol	Action
1	3.3.3.6:*	1.1.1.2:22	TCP	ACCEPT
2	4.4.4.8:*	1.1.1.1:25	TCP	ACCEPT
3	2.2.2.0/24:*	2.2.2.0/24:*	TCP	ACCEPT
4	2.2.2.0/24:*	4.4.4.7:80	TCP	ACCEPT
5	4.4.4.0/24:*	2.2.2.3:80	TCP	ACCEPT
Default action:		DROP		

e sono state avviate una serie di comunicazioni che hanno portato la SPI Table ad avere le seguenti righe registrate (per semplicità viene omesso lo stato della sessione):

SPI	SrcIP:SrcPort	DstIP:DstPort	State
1	3.3.3.6:44981	1.1.1.2:22	...
2	3.3.3.6:47231	1.1.1.1:22	...
3	3.3.3.6:47231	1.1.1.2:22	...
4	4.4.4.7:40327	2.2.2.3:80	...
5	4.4.4.7:47231	2.2.2.3:80	...
6	2.2.2.3:44981	4.4.4.7:80	...
7	2.2.2.4:22	2.2.2.4:40327	...
8	4.4.4.8:44981	1.1.1.1:25	...

Primo caso

Si spieghi quale sarà l'ordine di controllo e la relativa sorte per un pacchetto ricevuto dal firewall che abbia le seguenti caratteristiche:

- SrcIP:SrcPort: **1.1.1.2:23**
- DstIP:DstPort: **3.3.3.5:44981**.

Per prima cosa verrà analizzata la SPI Table per capire se il pacchetto appartiene ad una comunicazione già in atto. Dato che nella SPI Table non è presente nessuna riga che riporti queste caratteristiche, il pacchetto sarà analizzato tramite le regole di filtraggio. Nell'elenco delle regole impostate nel firewall non risulta presente alcuna riga che contenga una qualsiasi combinazione dei dati del pacchetto, e quindi verrà applicata la regola di default e il pacchetto sarà scartato.

Secondo caso

Si spieghi quale sarà l'ordine di controllo e la relativa sorte per un pacchetto ricevuto dal firewall che abbia le seguenti caratteristiche:

- SrcIP:SrcPort: **4.4.4.7:80**
- DstIP:DstPort: **2.2.2.3:44981**.

Per prima cosa verrà analizzata la SPI Table per capire se il pacchetto appartiene ad una comunicazione già in atto. Dato che nella SPI Table è presente la riga 6 che riporta i dati del pacchetto, indipendentemente dall'ordine dei vari indirizzi, il pacchetto ricevuto appartiene ad una comunicazione già in atto e quindi viene automaticamente accettato senza che vengano analizzate le regole di filtraggio.

La sicurezza delle reti: Server Proxy Firewall

Libro vol.3 - Firewall, Proxy, ACL e DMZ

STUDIARE: L. Lo Russo, E. Bianchi, *Sistemi e Reti - Nuova Edizione OPENSCHOOL - Per l'articolazione INFORMATICA degli Istituti Tecnici settore Tecnologico*, vol. 3, ed. Hoepli, 2017.

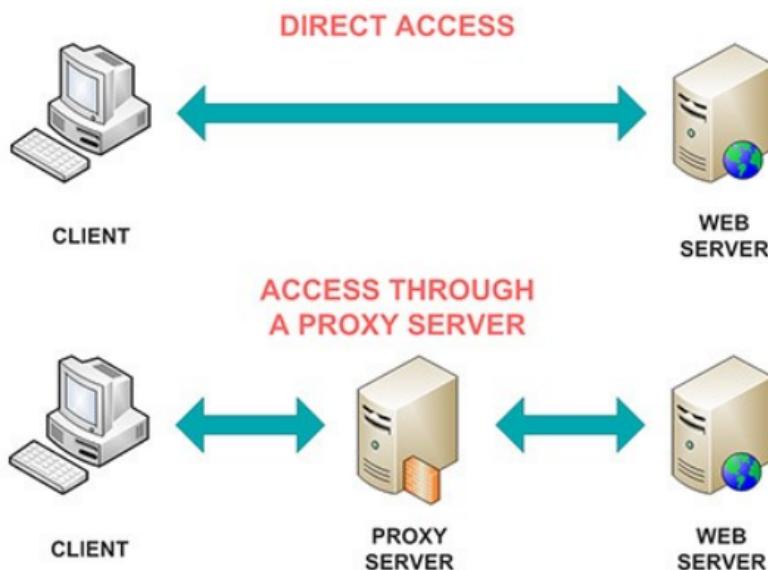
- Uda4 - La sicurezza delle reti
 - Lezione - Firewall, Proxy, ACL e DMZ
 - Application proxy pp.197-199
 - DMZ pp.199-202

LEGGERE: Firewall su [Wikipedia](#), soprattutto [Next Generation Firewall](#)

Server Proxy Firewall

Un **Server Proxy Firewall** lavora a livello Applicazione o a livello Sessione/Trasporto della pila protocolare ISO/OSI, ed è in grado di selezionare i pacchetti effettuando un'ispezione dei dati contenuti in essi, oppure comprendere quali connessioni TCP risultano ammesse, effettuando l'inoltro per conto del client dei pacchetti o bloccandoli a seconda della configurazione delle regole impostate.

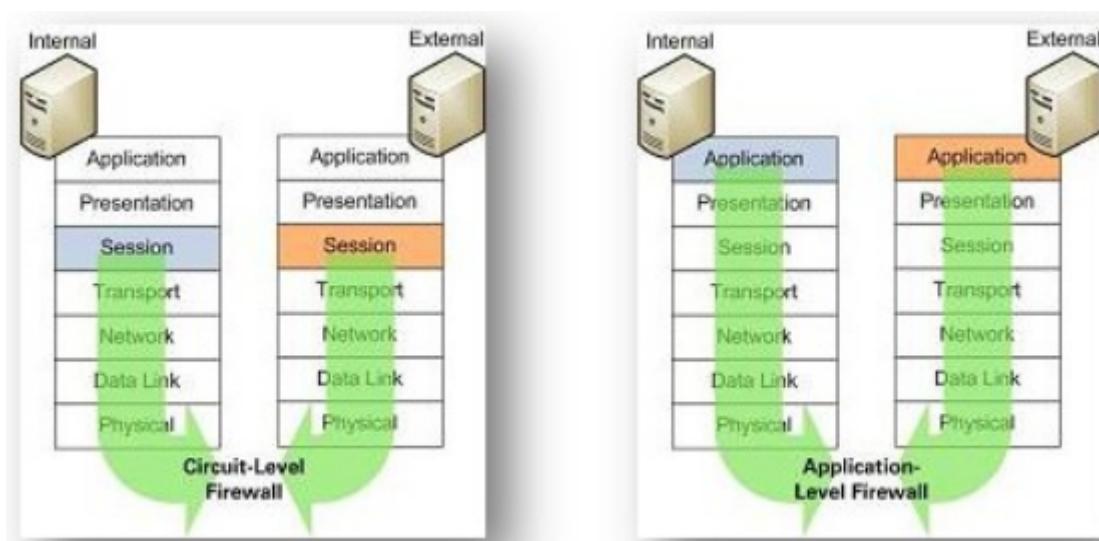
Tutti i client sono forzati ad inviare il loro traffico verso il **Server Proxy**, risultando bloccato l'accesso diretto a server esterni in modo da operare il filtraggio dei pacchetti in uscita e in ingresso. Di fatto **il Server Proxy spezza la comunicazione tra client e server** frapponendosi fra loro, **inoltrando i pacchetti alle due entità** solo dopo averli controllati.



Un **Server Proxy** viene usato per effettuare:

- **caching**, memorizzando i risultati delle richieste al fine di migliorare le prestazioni; in questo modo offre l'accesso rapido alle risorse già accumulate nella cache e riduce il traffico nella rete che protegge;
- **controllo**, applicando delle regole per determinare quali richieste inoltrare o rifiutare e limitare l'ampiezza di banda usata dai client;
- **monitoraggio** permettendo di tenere traccia delle operazioni effettuate da una stazione, identificando il suo indirizzo IP, oppure individuando l'utente tramite l'account con cui si è autenticato.

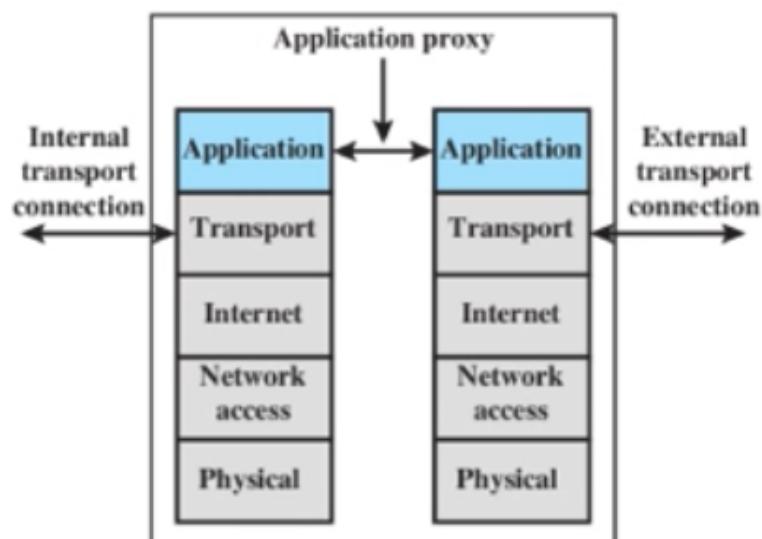
I **Server Proxy** si suddividono in due categorie, gli **Application Proxy** che lavorano al livello Applicativo della pila protocollare ISO/OSI, e i **Circuit-Level Proxy firewall** che invece lavorano a livello Sessione/Trasporto.



Application Proxy o Application-level Gateway

Un **Application Proxy** è un firewall con funzionalità più avanzate rispetto a quelle viste finora in quanto **permette di filtrare i pacchetti in base al contenuto dei dati trasmessi**. Questo vuol dire che, ad esempio, non viene proibito l'accesso ad un Social network, ma viene impedito l'invio di messaggi allo stesso Social network nel caso questi veicolino contenuti non desiderati.

Un **Application Proxy** effettua un inoltro, per conto del client, del traffico del livello applicativo diretto verso un host remoto. Dato che un **Application Proxy** **lavora a livello applicativo**, **potrà filtrare i pacchetti in modo più sofisticato**, ma

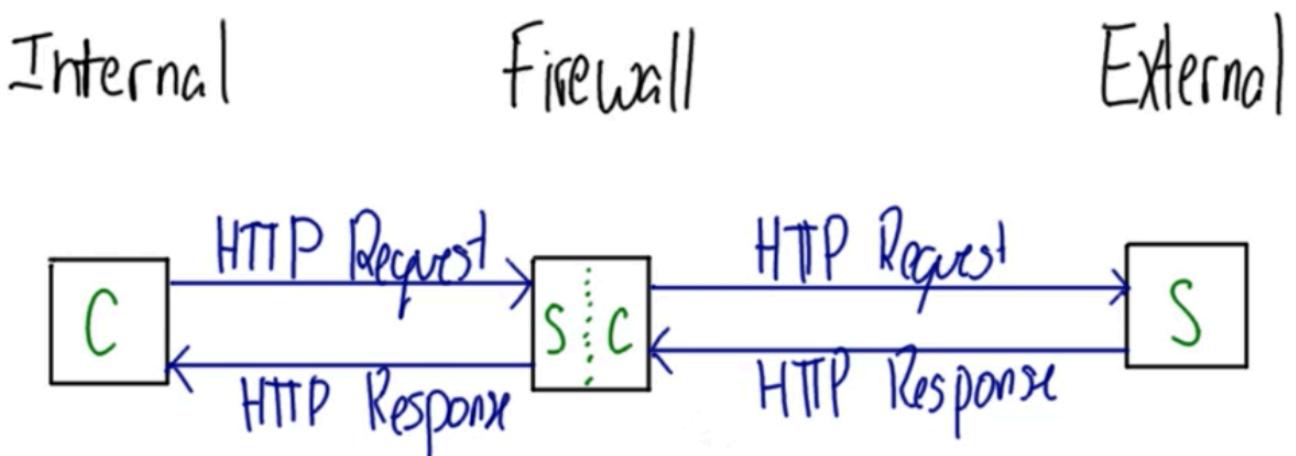


soltamente **sarà in grado di farlo solo per una specifica applicazione**, ad esempio FTP, HTTP, SMTP, ecc. Questa tecnica di filtraggio aggiunge un **maggior overhead ad ogni comunicazione**, ma per la particolare applicazione gestita risulterà **più sicuro rispetto ai firewall visti in precedenza**.

Usando un **Application Proxy**, se un client desidera connettersi ad un server remoto, dovrà prima di tutto inviare la propria richiesta al proxy, ad esempio richiedendo un file, una pagina web o qualsiasi altra risorsa disponibile sul server di interesse. Il proxy valuterà se inoltrare o meno la richiesta in base alle regole configurate e, nel caso sia ritenuto possibile soddisfarla, questa verrà inoltrata al server di interesse. Il server consegnerà la risposta al proxy e quest'ultimo, dopo averne controllato il contenuto, la inoltrerà al client.

Usando un **Application Proxy non si ha più una connessione diretta tra macchina interna ed esterna alla rete, bensì vengono gestite due connessioni separate**, una tra il client ed il proxy, l'altra tra il proxy e il server. Dal punto di vista del client il proxy risulterà essere il server del servizio richiesto, mentre dal punto di vista del server reale, il proxy figurerà come il client del servizio. Questa modalità permetterà di creare una netta distinzione tra rete interna e rete esterna.

Ad esempio, utilizzando un Web proxy, il client invierà la HTTP Request al proxy, invece di inviarla direttamente al server Web. Se, in base alle regole del proxy, la richiesta potrà essere inoltrata, sarà il proxy ad inviare una HTTP Request verso il Web server. Se invece la richiesta non potrà essere inoltrata, sarà scartata dal proxy. Se il Web server ricevesse la HTTP Request dal proxy, quest'ultimo figurerà come client per il Web server che, di fatto, non saprà mai l'indirizzo IP del client che aveva originariamente inviato la richiesta. Il Web server invierà quindi la HTTP Response al proxy e, se verrà ritenuta affidabile dal proxy, verrà inoltrata al client originario.

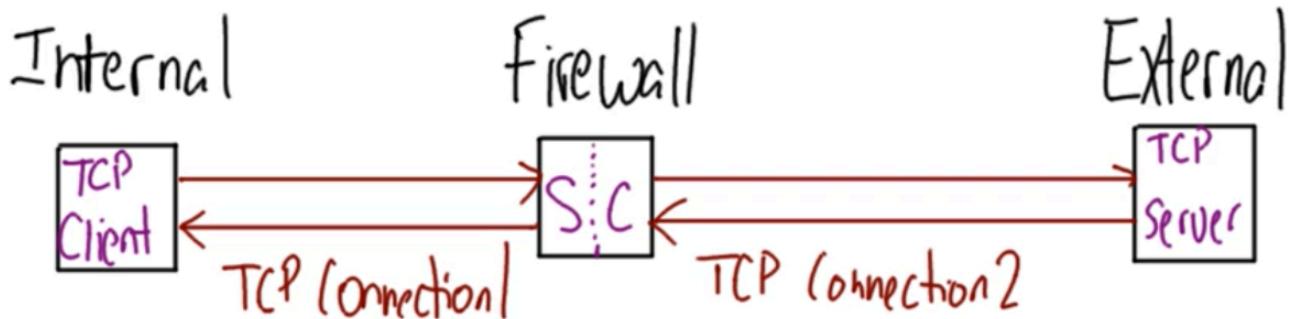
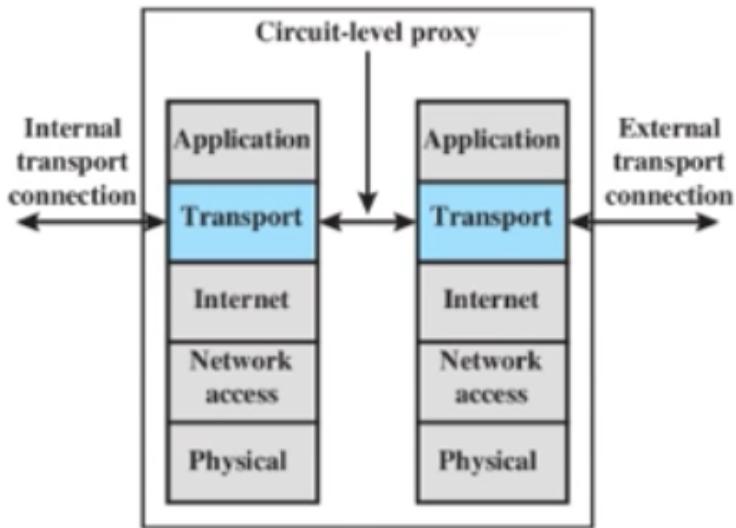


La presenza del firewall in mezzo alla comunicazione di fatto modifica la richiesta originale, e questo potrebbe avere un qualche impatto con certe applicazioni.

Circuit-Level Proxy Firewall o Circuit-Level Gateway

Un **Circuit-Level Proxy Firewall** si frappone tra la comunicazione client-server come un Application Proxy Firewall, ma a differenza di questo **lavora a livello TCP effettuando un inoltro dei segmenti TCP fra il client e il server.**

Questo tipo di firewall **non è vincolato ad una particolare applicazione** perché non effettua un controllo sul dato contenuto nel segmento, ma **effettua un controllo dei segmenti, instaurando due distinte connessioni TCP**, una con il client e una con il server originario.



La funzione di filtraggio di questo proxy si basa nell'individuare quali connessioni TCP risultano permesse e viene usato quando gli utenti della rete da proteggere risultano affidabili. Da questo punto di vista un **Circuit-Level Proxy** opera uno **Stateful Packet Inspection**, ma a differenza di uno Stateful firewall ha la capacità di spezzare la connessione tra un host della rete che protegge e un host della rete esterna. Infatti nelle comunicazioni svolge il ruolo di intermediario ed è quindi l'unico punto della rete che comunica con l'esterno, nascondendo gli altri host che vi appartengono.

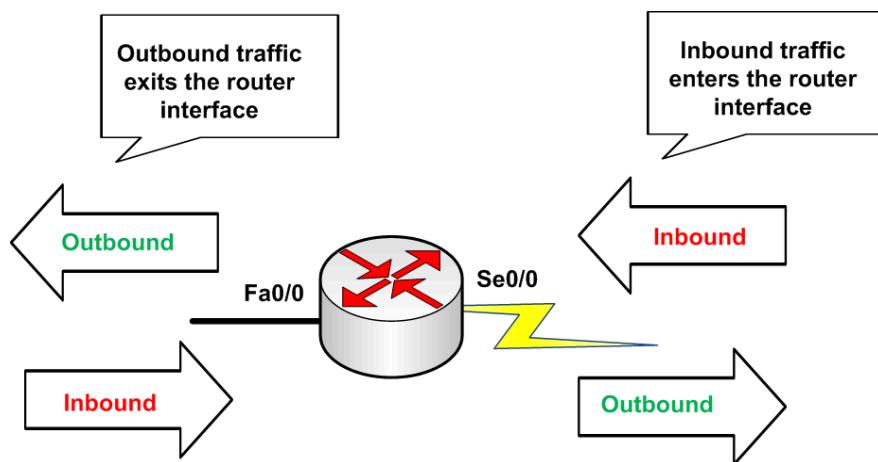
Per garantire una maggiore sicurezza associata ad un minor overhead è possibile applicare un **Application-Level Gateway** sulle connessioni TCP entranti dall'esterno verso la rete da proteggere, e un **Circuit-Level Gateway** sulle connessioni TCP uscenti dalla rete da proteggere.

Access Control List [leggere]

Introduzione

Le **ACL** (Access Control List) permettono di effettuare il filtraggio dei pacchetti che raggiungono un router. Le **ACL** sono configurate dal sistemista di rete sul router che riveste anche il ruolo di firewall, e indicano quali pacchetti debbano essere scartati e quali invece debbano attraversare il router.

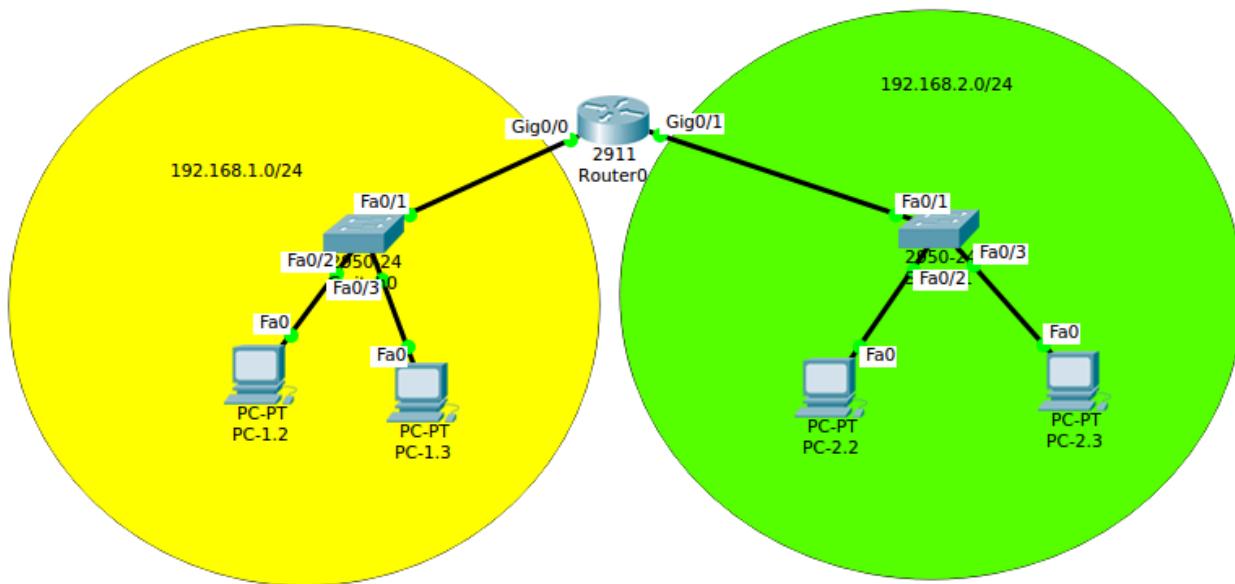
Ogni **ACL** va definita su una porta del router, specificando se il traffico filtrato da tale regola sia in ingresso (**inbound**) o in uscita (**outbound**):



Una spiegazione introduttiva sulle **ACL** fornita dalla Prof.ssa Sophia Danesino è raggiungibile al seguente link: [ACL: Introduzione \(IT\)](#)

Laboratorio: ACL standard outbound

Si svolga il laboratorio illustrato dalla Prof.ssa Sophia Danesino a questo [link](#) e si legga la spiegazione seguente.



Obiettivo: **PC-2.3 non deve accedere alla rete Gialla.**

1. Creare l'ACL

Per creare una ACL standard si utilizzerà il seguente comando:

```
Router(config)# access-list 1 deny 192.168.2.3 0.0.0.035
```

mentre per inserire la **policy** (regola) **di default** si dovrà utilizzare il seguente comando:

```
Router(config)# access-list 1 permit any
```

Wildcard mask: è una maschera di bit che indica quale parte di un indirizzo IP deve essere analizzata. Uno 0 binario nella *wildcard mask* indica una posizione dell'indirizzo IP che deve essere confrontata con una regola presente nel router, mentre un 1 binario nella *wildcard mask* indica una posizione dell'indirizzo IP che non deve essere confrontata con una regola presente nel router. Esempio: /24 diventa 0.0.0.255.

Il numero dell'ACL dipende dal tipo di ACL:

ACL standard	1-99	Filtrà in base all'IP sorgente
ACL estesa	100-199	Filtrà in base all'IP sorgente e destinatario, porta sorgente e destinataria, numero di porta (servizio)

³⁵ Anche: access-list 1 deny **host** 192.168.2.3 (**host** corrisponde alla wildcard 0.0.0.0, **any** alla wildcard 255.255.255.255).

Le regole verranno esaminate in ordine di inserimento dalla prima all'ultima, fino a trovare un pattern corrispondente o fino alla policy di default, che sarà sempre l'ultima regola analizzata.

Per inserire correttamente le regole è sempre meglio definire prima le regole più specifiche e aggiungere successivamente quelle più generali.

Per **elencare le ACL** definite in un router potrà essere usato il comando:

```
Router# show access-lists
```

2. Applicare la ACL ad una interfaccia

Una volta definita la **ACL** sarà necessario indicare al router su quale interfaccia dovrà essere applicata:

```
Router(config)# interface Gig0/0
Router(config-if)# ip access-group 1 out
```

Il parametro **out** indica che la **ACL** deve essere applicata sui pacchetti in uscita (**outbound**) dall'interfaccia del router.

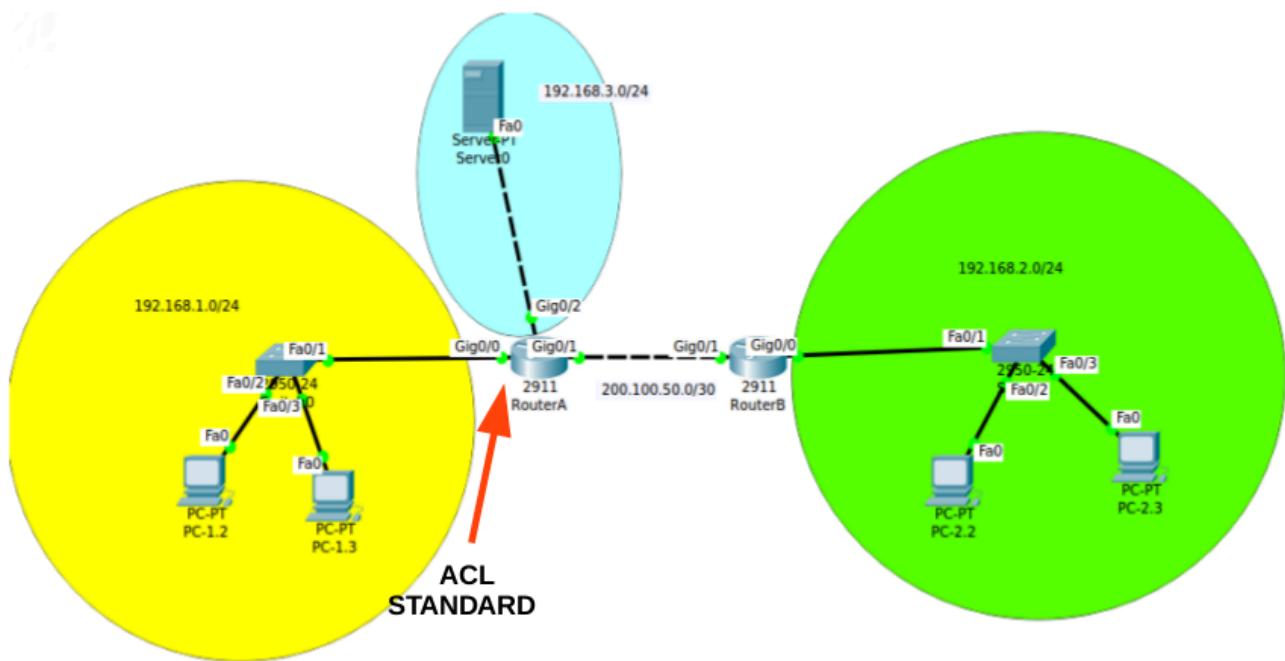
In generale si seguono le seguenti regole per la definizione delle **ACL**:

- le **ACL standard** devono essere definite in una **posizione più vicina possibile all'host di destinazione**;
- le **ACL estese** devono essere definite in una **posizione più vicina possibile all'host sorgente**.

Laboratorio: ACL standard inbound

Si svolga il laboratorio illustrato dalla Prof.ssa Sophia Danesino in questo [link](#) e si legga la spiegazione seguente.

Per capire meglio dove impostare le ACL esaminiamo questo scenario:



Le reti sono tre, e si vuole **impedire al PC-2.3 di accedere alla rete 192.168.1.0/24**.

Uno dei problemi da risolvere con le ACL è quello di filtrare solo il traffico necessario, senza influenzare quello che non necessita di essere controllato. Nell'esempio mostrato, se si impostasse la ACL 1 sul router RouterB, la ACL filtrerebbe anche il traffico rivolto verso la rete 192.168.3.0/24, cosa che non deve accadere.

Affinché la **ACL standard definita** nella topologia mostrata in figura lavori correttamente, deve essere definita sul router RouterA e deve essere specificata sulla porta a cui è collegata la rete Gialla, cioè **il più vicino possibile alla destinazione**.

```
RouterA(config)# access-list 1 deny 192.168.2.3 0.0.0.0
RouterA(config)# access-list 1 permit any
RouterA(config)# interface Gig0/0
RouterA(config-if)# ip access-group 1 out
```

Il comando per eliminare una ACL:

```
RouterB(config)# no access-list 1
```

Laboratorio: ACL estese

Si svolga il laboratorio illustrato dalla Prof.ssa Sophia Danesino in questo [link](#) e si legga la spiegazione seguente.

La configurazione di una ACL standard prossima alla destinazione può creare dei problemi in quanto non è detto che si abbia l'accesso al router in questione. L'uso delle ACL estese permette invece di lavorare su un router direttamente accessibile in quanto **le ACL estese devono essere definite nella porta del router più vicina alla sorgente della comunicazione.**

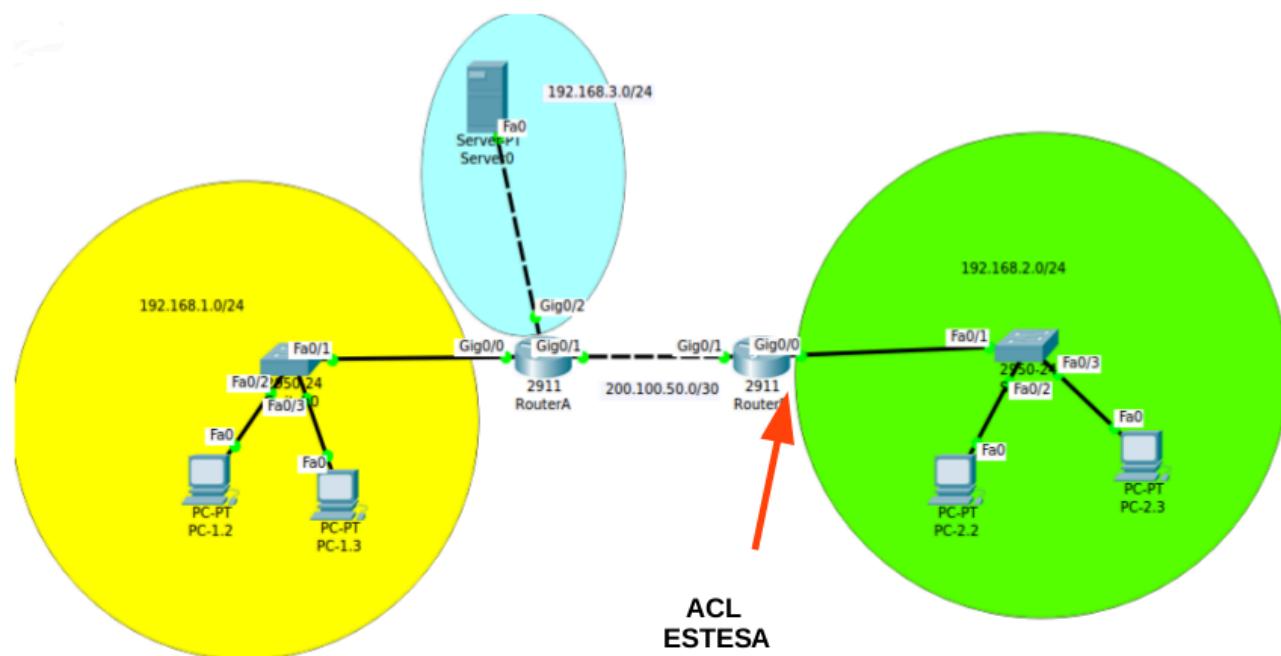
Considerando lo scenario mostrato in figura, **si vuole impedire al PC-2.3 di accedere alla rete 192.168.1.0/24**, e per farlo **si utilizzerà una ACL estesa**. Si osservi che nei comandi seguenti l'uso di un numero superiore a 99 assegnato alla ACL, implica direttamente l'uso di una ACL estesa.

```
RouterB(config)# access-list 100 deny ip 192.168.2.3 0.0.0.0
192.168.1.0 0.0.0.255
```

```
RouterB(config)# access-list 100 permit ip any any
```

La tabella seguente illustra il significato di tutti i parametri della prima istruzione:

access-list	100-199	permit/deny	protocoll o	indirizzo IP origine	wildcard origine	indirizzo IP destinazione	wildcard destinazione
-------------	---------	-------------	-------------	----------------------	------------------	---------------------------	-----------------------



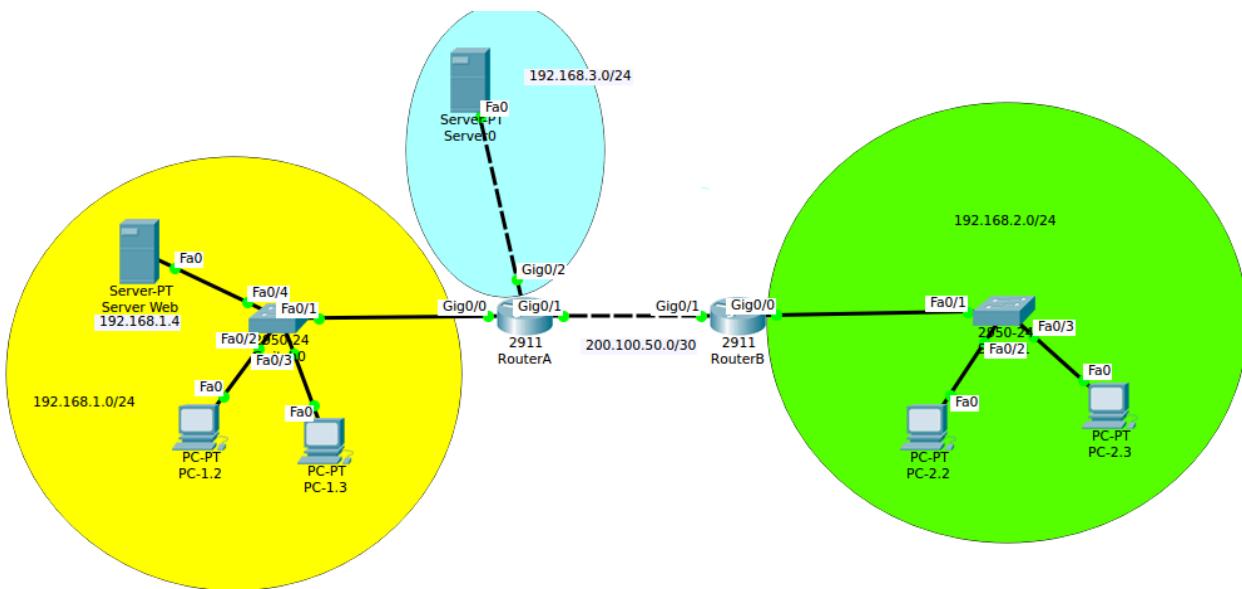
L'ACL estesa va assegnata alla porta più vicina all'host sorgente quindi su RouterB porta Gig0/0.

```
RouterB(config)# interface Gig0/0
RouterB(config-if)# ip access-group 100 in
```

Laboratorio: ACL riferite a servizi

Si svolga il laboratorio illustrato dalla Prof.ssa Sophia Danesino in questo [link](#) e si legga la spiegazione seguente.

Si aggiunga un server HTTP e HTTPS alla rete Gialla. **La rete Verde non deve accedere alla rete Gialla tranne che per accedere al server Web, mentre l'accesso alla rete Azzurra deve essere consentito.**



```
RouterB(config)# access-list 100 permit tcp 192.168.2.0 0.0.0.255 host  
192.168.1.4 eq 80  
RouterB(config)# access-list 100 permit ip 192.168.2.0 0.0.0.255  
192.168.3.0 0.0.0.255
```

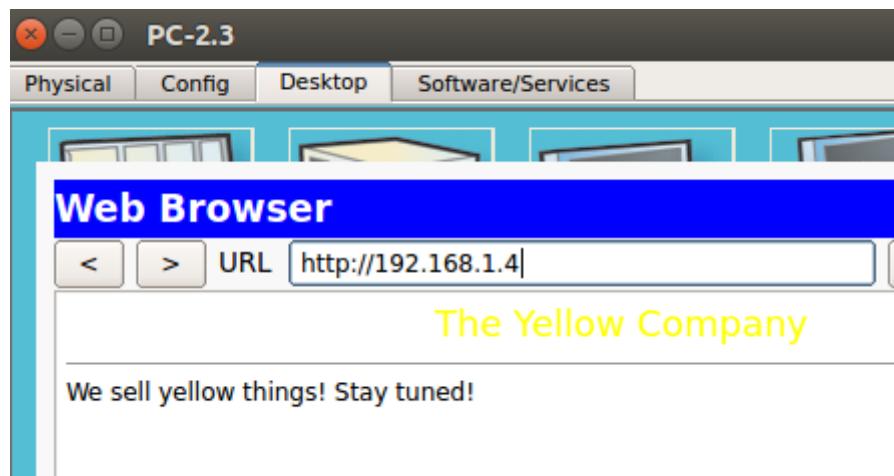
Dato che **nelle ACL estese la policy di default è deny any, non è necessario inserire questa regola.**

Si osservi che il parametro **eq 80** specifica il **numero di porta del servizio filtrato**.

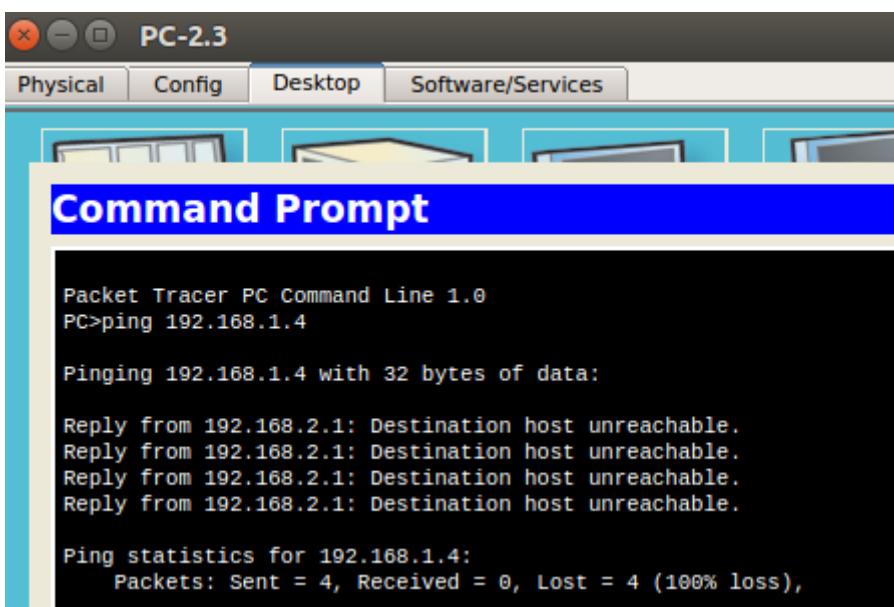
Infine la ACL estesa dovrà essere applicata alla porta Gig0/0 inbound, cioè alla porta più vicina alla sorgente:

```
RouterB(config)# interface Gig0/0  
RouterB(config-if)#ip access-group 100 in
```

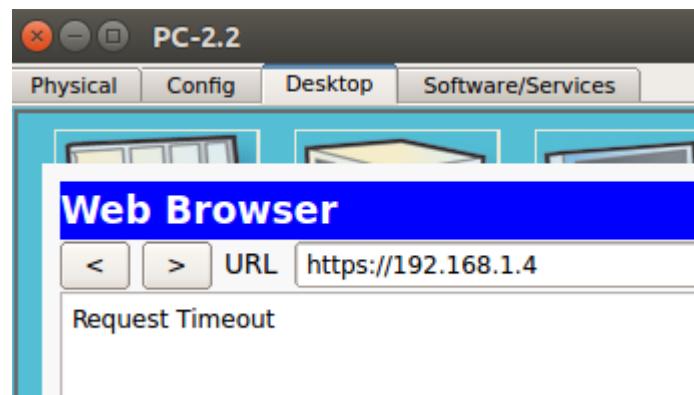
Grazie all'impostazione della ACL estesa la rete Verde riuscirà solo ad accedere alla rete Gialla sulla porta 80.



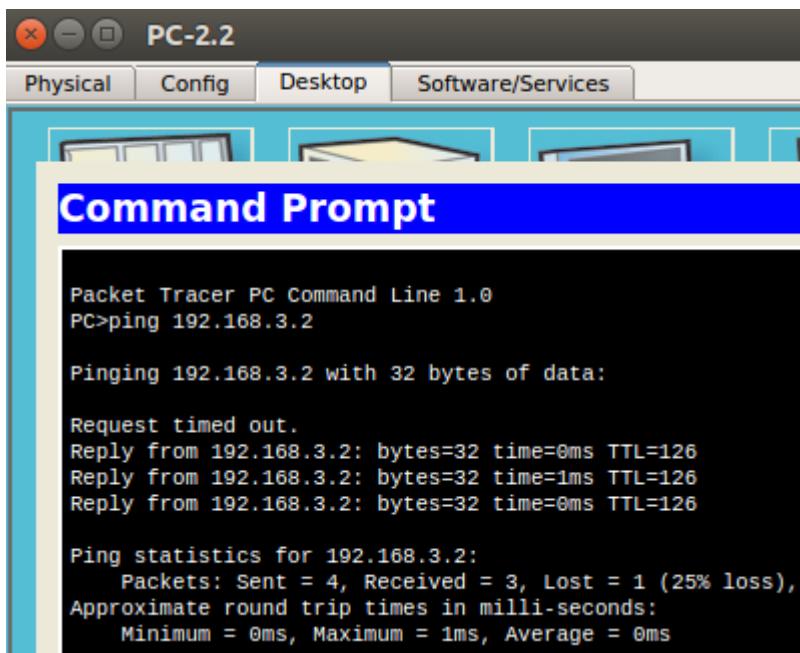
Nell'esempio seguente la ACL estesa impostata sulla porta Gig0/0 del router RouterB filtra il ping verso il Web server 192.168.1.4, e sarà direttamente l'interfaccia Gig0/0 del router RouterB a rispondere con un Reply from 192.168.2.1: Destination host unreachable, indicando che è stata direttamente bloccata la ping request del PC-2.3 in quanto la richiesta è stata fatta verso una porta diversa dalla 80.



La stessa cosa avverrà nel caso si tentasse di contattare il server 192.168.1.4 utilizzando HTTPS che utilizza la porta 443.



Sarà invece garantito l'accesso alla rete 192.168.3.0/24.



The screenshot shows a Cisco Packet Tracer interface titled "PC-2.2". The "Software/Services" tab is selected. A window titled "Command Prompt" displays the following output:

```
Packet Tracer PC Command Line 1.0
PC>ping 192.168.3.2

Pinging 192.168.3.2 with 32 bytes of data:

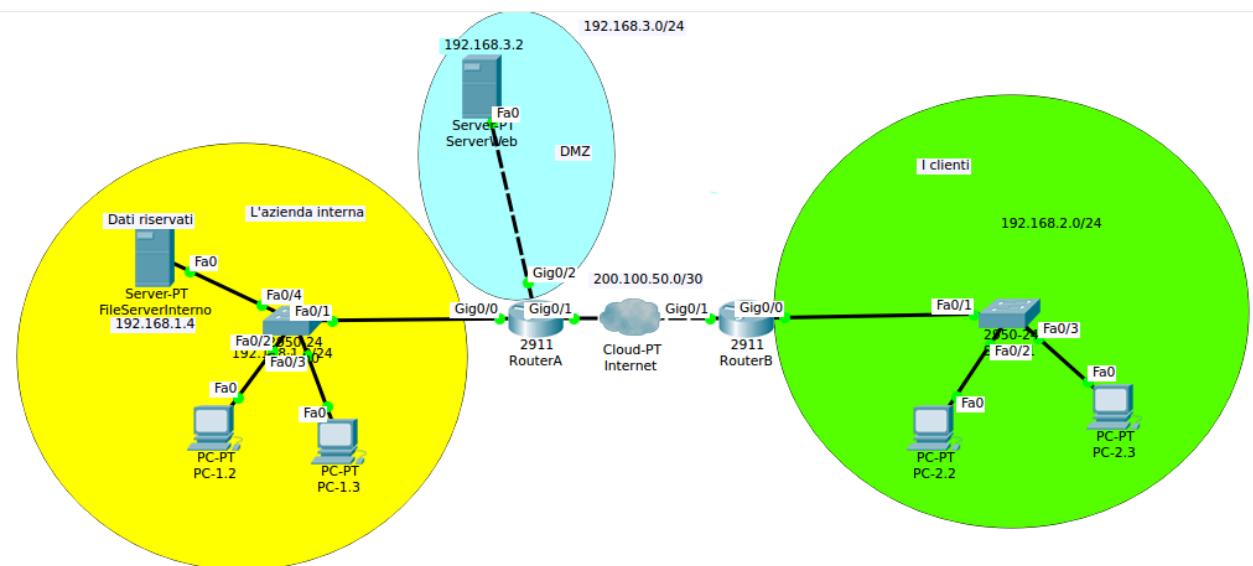
Request timed out.
Reply from 192.168.3.2: bytes=32 time=0ms TTL=126
Reply from 192.168.3.2: bytes=32 time=1ms TTL=126
Reply from 192.168.3.2: bytes=32 time=0ms TTL=126

Ping statistics for 192.168.3.2:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 1ms, Average = 0ms
```

Laboratorio: DMZ

Si svolga il laboratorio illustrato dalla Prof.ssa Sophia Danesino in questo [link](#) e si legga la spiegazione seguente.

Nel laboratorio seguente si distinguerà fra la modalità di accesso di una LAN protetta dall'esterno e la gestione di un server pubblico posto in una DMZ.



L'azienda interna deve essere protetta dall'accesso proveniente dal mondo esterno, mentre l'accesso alla zona DMZ deve essere consentito. Per consentire queste modalità di accesso si lavorerà solo sul router RouterA.

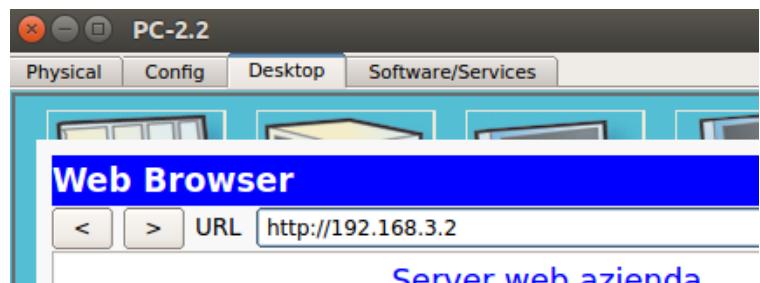
Dovrà essere consentito l'accesso dall'esterno al server Web nella DMZ:

```
Router(config)# access-list 100 permit tcp any host 192.168.3.2 eq 80
```

La ACL deve essere assegnata alla porta di ingresso del RouterA:

```
RouterA(config)# interface gigabitEthernet 0/1
RouterA(config-if)# ip access-group 100 in
```

Impostando queste configurazioni sarà consentito l'accesso dall'esterno al server web in DMZ, e dall'esterno non sarà possibile raggiungere la rete interna dell'azienda.



```
PC>ping 192.168.1.4
Pinging 192.168.1.4 with 32 bytes of data:
Reply from 192.168.2.1: Destination host unreachable.
```

Se fosse necessario consentire ai dipendenti della rete interna all'azienda di utilizzare un accesso Internet l'impostazione delle sole regole precedenti non consentirebbero la navigazione. Infatti il firewall non bloccherebbe la request degli host della rete interna, bensì bloccherebbe la replay, cioè la risposta dall'esterno.

Le configurazioni impostate sul firewall non permettono di distinguere tra un flusso di dati generato dalla WAN e un flusso di risposta ad una esplicita richiesta proveniente dalla LAN. Di conseguenza, se si volesse permettere la navigazione Web agli host nella rete interna dell'azienda, sarebbe necessario aggiungere la seguente ACL:

```
RouterA(config)# access-list 100 permit tcp any eq 80 192.168.1.0  
0.0.0.255
```

In questo modo verrebbe consentita la reply ad una richiesta proveniente dalla LAN interna all'azienda. Ad esempio, se si suppone che una applicazione client, identificabile dal socket 192.168.1.2:40000, effettui una richiesta verso un server Web collocato nella rete Verde, e identificato dal socket 192.168.2.2:80, la regola impostata permetterà di effettuare la request dalla LAN interna all'azienda verso il server Web e, soprattutto, permetterà alla reply da parte del server Web.

```
access-list 100 permit tcp any eq 80 192.168.1.0 0.0.0.255  
reply da 192.168.2.2:80 a 192.168.1.2:40000
```

Si ricordi che l'avvio di una request HTTP presuppone inizialmente l'instaurazione di una connessione TCP, quindi se non fosse impostata la regola appena illustrata, dopo l'invio da parte del client del primo segmento SYN per l'instaurazione della connessione, sarebbe bloccato dal firewall il segmento SYN-ACK inviato dal protocollo TCP del server Web.

Purtroppo l'impostazione di questa regola non garantisce però che la reply sia effettivamente quella inviata dal server Web, infatti potrebbe essere un nuovo flusso di dati creato ad arte con un generatore di pacchetti. Per evitare questa eventualità sarà necessario rendere la ACL di tipo stateful, aggiungendo la parola chiave **established**:

```
RouterA(config)# access-list 100 permit tcp any eq 80 192.168.1.0  
0.0.0.255 established
```

Sistemi distribuiti e tecniche di amministrazione

Modelli distribuiti per i servizi di rete e amministrazione dei sistemi

Libro vol. 3 - Modello client-server e distribuito per i servizi di rete

STUDIARE: L. Lo Russo, E. Bianchi, *Sistemi e Reti - Nuova Edizione OPENSCHOOL - Per l'articolazione INFORMATICA degli Istituti Tecnici settore Tecnologico*, vol. 3, ed. Hoepli, 2017.

- UdA6 - Modello client-server e distribuito per i servizi di rete
 - Lezione 1 - Le applicazioni e i sistemi distribuiti pp.298-304
 - Lezione 2 - Architetture dei sistemi Web pp.305-311
 - Lezione 3- Amministrazione di una rete pp.312-323
 - Lezione 4 - Active Directory pp.324-335 (*le parti di configurazione sono solo da leggere*)
 - Lezione 5 - Il troubleshooting pp.336-343 (*leggere*)
 - Lezione 6 - La sicurezza della rete pp.344-354

Macchine e servizi virtuali

La Virtualizzazione

CLIL - What Virtualization is

Watch the following video and discuss with your classmates about Virtualization.

[What is Virtualization?](#)

Le ragioni della virtualizzazione

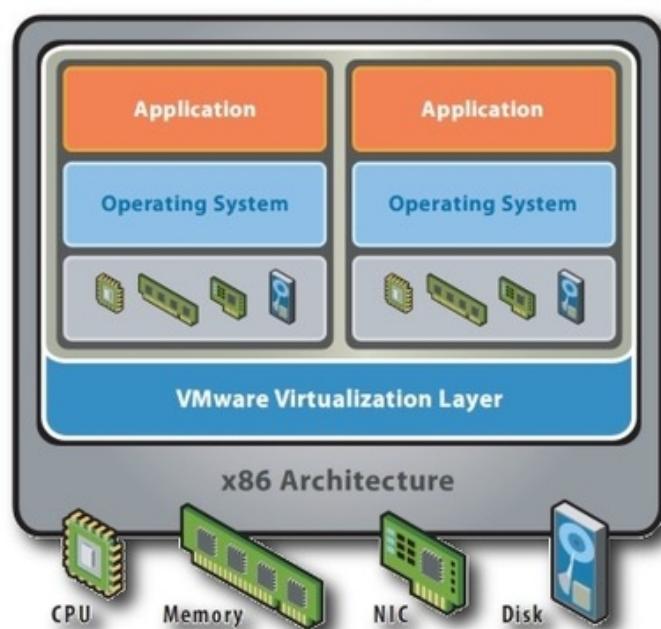
La **virtualizzazione** è una tecnologia software che permette di condividere le risorse di un computer, eseguendo simultaneamente più sistemi operativi e applicazioni sulla stessa macchina fisica.

Oggi i computer hanno raggiunto un elevatissima capacità di calcolo, di cui solo una minima parte effettivamente sfruttata. Questo si traduce in uno spreco energetico perché, in ogni caso, le risorse devono essere alimentate indipendentemente dal loro stato di utilizzo. Per ottimizzare l'utilizzo di queste risorse una soluzione molto diffusa consiste nella **virtualizzazione**.

La **virtualizzazione** permette di realizzare tramite software una condivisione delle risorse, per creare ambienti indipendenti in grado di riprodurre le stesse funzioni di un computer fisico. **Si tratta di un'astrazione dell' hardware** che permette di suddividere gli elementi fisici dando la possibilità di utilizzarli in modo più efficiente e flessibile.

Una **macchina virtuale**, allo stesso modo di una macchina fisica, dispone di una scheda madre con una memoria RAM e una scheda video, il controller della scheda di rete e l'hard disk, con la differenza che questi sono componenti virtuali che sfruttano e condividono le stesse risorse hardware.

In pratica la **macchina virtuale** è **identica ad una macchina fisica**, con la differenza che i componenti logici sono indipendenti dall'hardware su cui sono in esecuzione e si possono spostare su macchine diverse; in pratica un server virtuale **diventa un file** che può essere copiato. Inoltre una **macchina virtuale** è **compatibile con i sistemi operativi standard** e usa gli stessi driver dei dispositivi, eseguendo le applicazioni come su una macchina fisica. Infine la **macchina virtuale incapsula un sistema di elaborazione completo**



risultando allo stesso tempo totalmente isolata dalle altre macchine virtuali e operando in modo completamente distinto dalle altre, anche se tutte risiedono sulla stessa macchina fisica e condividano lo stesso hardware.

La **virtualizzazione** offre molti **vantaggi** tra cui:

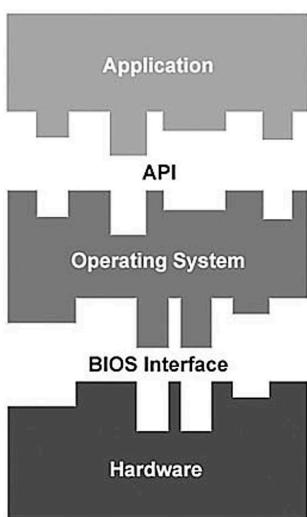
- la **riduzione del numero di server** presenti nei centri di calcolo e nelle organizzazioni;
- la **riduzione dello spazio fisico necessario per le macchine**;
- la **diminuzione dei consumi energetici**;
- l'**abbattimento dei costi di manutenzione**.

La **virtualizzazione** può essere realizzata purché siano rispettati i seguenti **requisiti**:

- **partizionamento dell'hardware**: gli elementi fisici sono messi a disposizione di ambienti multipli in grado di condividere le risorse comuni, mantenendo l'indipendenza degli ambienti;
- **mantenimento delle proprietà di esecuzione**: è compito del processo di virtualizzazione gestire correttamente le code di esecuzione, sfruttando il *time sharing* e le proprietà di *real time execution*;
- **replica dell'ambiente di astrazione hardware**: ogni ambiente virtualizzato ha un proprio livello hardware, indipendente e isolato dagli altre, anche se condivide le risorse del computer fisico che lo ospita;
- **gestione delle periferiche del computer ospitante, condivise con gli ambienti virtuali creati**: è compito del processo di virtualizzazione (**hypervisor**) gestire la condivisione delle risorse fisiche allocandone gli indirizzamenti, così come gestire le tabelle di traslazione contenenti gli indirizzi che permettono di far corrispondere le risorse visibili alle macchine virtuali con le risorse fisiche;
- **mantenimento dei criteri di sicurezza e d'isolamento**: ogni ambiente virtualizzato mantiene l'indipendenza sia verso gli ambienti virtuali sia verso il sistema di virtualizzazione stesso; non sono ammessi scambi di *processo* e *memory sharing* a livello applicativo. In questo modo le aree di memoria delle macchine virtuali non sono accessibili a nessun'altra macchina, logica o fisica che sia. Questo fornisce l'assoluta certezza dell'integrità dei dati e della sicurezza nel trattamento delle informazioni.

Architetture delle macchine virtuali

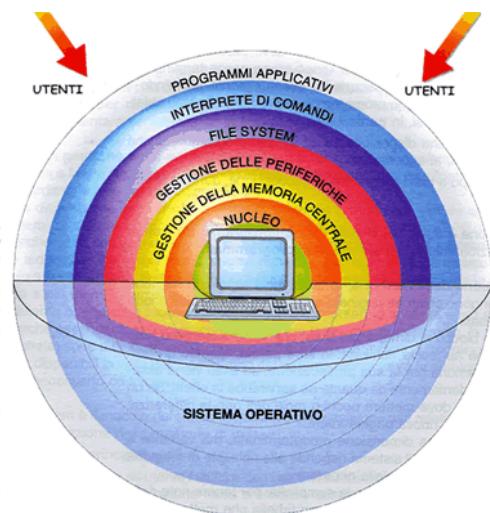
Layer fisico e layer virtuale



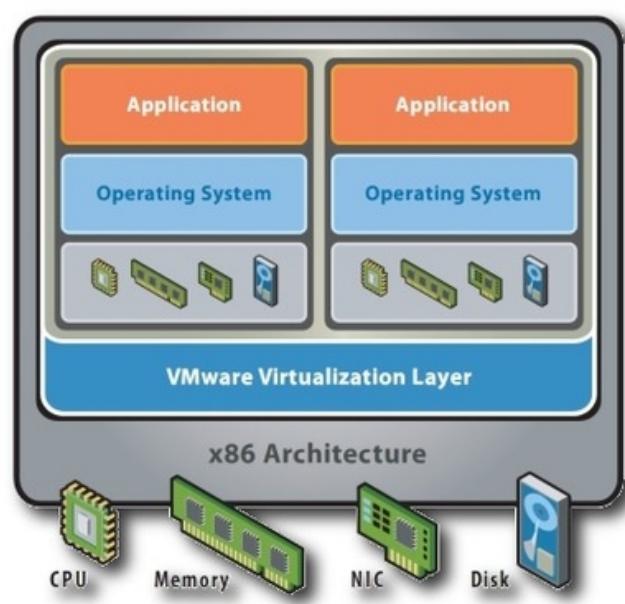
Un normale computer è costituito dall'hardware (CPU, memoria RAM, memoria di massa, periferiche di I/O) su cui poggia il BIOS (Basic Input Output System) il quale rappresenta un primo livello di software di base, che permette di inizializzare le risorse necessarie alla partenza della macchina (fase di *boot*).

In un **ambiente tradizionale** un sistema operativo si compone di un nucleo, chiamato *kernel*, che si interfaccia ai driver per la gestione delle periferiche hardware. Sul *kernel* si poggia l'interprete dei comandi, che prende il nome di *shell*.

In un **ambiente virtualizzato** invece il sistema operativo ha anche scopo di gestire le risorse fisiche e creare gli ambienti in cui saranno alloggiate le nuove macchine, a loro volta composte da uno strato hardware virtuale, un proprio BIOS, un proprio sistema operativo. Nella gestione di ambienti virtuali, la *shell* prende il nome di **hypervisor**, il cui scopo principale è quello di gestire le risorse fisiche assegnandole agli ambienti ospitati virtualmente.



Hypervisor o VMM



L'**hypervisor**, chiamato anche **VMM** (*Virtual Machine Monitor*) è il software di controllo che permette di assegnare e gestire le funzioni necessarie affinché le risorse fisiche siano rese disponibili alle macchine virtuali che saranno create.

Sopra lo strato dell'**hypervisor** si trova la replica dell'**astrazione hardware**, che riproduce perfettamente lo schema di un elaboratore, dove individuiamo gli stessi elementi tipici di un computer fisico: CPU, memoria RAM, memoria di massa, periferiche di I/O, un proprio BIOS, il sistema operativo della macchina virtuale e le applicazioni.

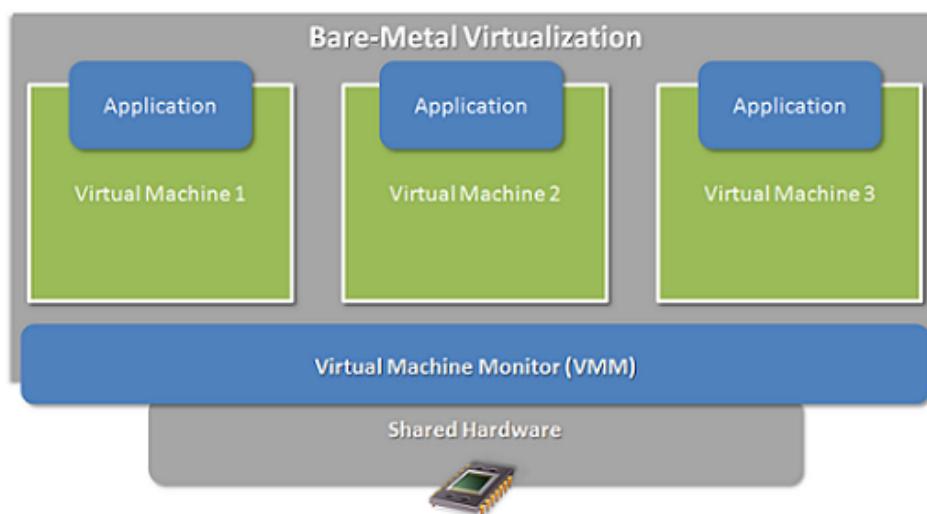
L'**hypervisor** non permette che dati e risorse si vadano a mescolare fra i diversi ambienti operativi creati, o fra questi e la macchina ospitante (*host fisico*).

L'**hypervisor** è un **programma di basso livello** che consente ai sistemi operativi ospitati (*guest*) di essere eseguiti in contemporanea su un singolo computer host. Il **codice dell'hypervisor** è ben strutturato ed **efficiente** perché deve allocare le risorse di memoria e di I/O in tempo reale.

Esistono due tipologie di **hypervisor**, il **bare metal** o **native** adatto per virtualizzazioni di grandi sistemi che richiedono un'elevata efficienza e velocità, e l'**hosted** che è facile da installare e gestire ma ha prestazioni inferiori rispetto alla precedente.

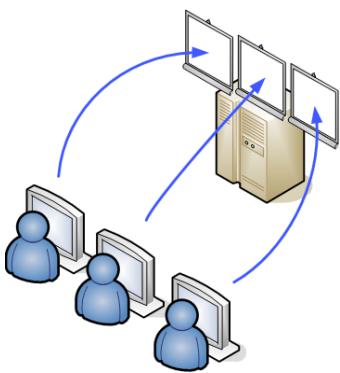
Hypervisor 1 (Bare Metal o Native)

In questa architettura il **software di virtualizzazione si integra direttamente con il kernel del sistema operativo** ed è in grado di interfacciarsi con le periferiche hardware, gestendo i processi in real time a livello del kernel stesso.



Il fatto di essere a diretto contatto con l'hardware permette una gestione **più efficiente, veloce e stabile** delle risorse assegnate alle macchine virtuali. Questa architettura è tipicamente usata per la **virtualizzazione dei server** e non prevede l'utilizzo di sistemi operativi preesistenti sulle macchine, garantendo **alte performance**. Con tale struttura è possibile consolidare diversi server, come File server, Database server, Web server, FTP server, DNS server, server di posta e di dominio. Sulla stessa macchina vengono mantenuti tutti questi servizi in esecuzione su macchine virtuali, riducendo drasticamente i costi.

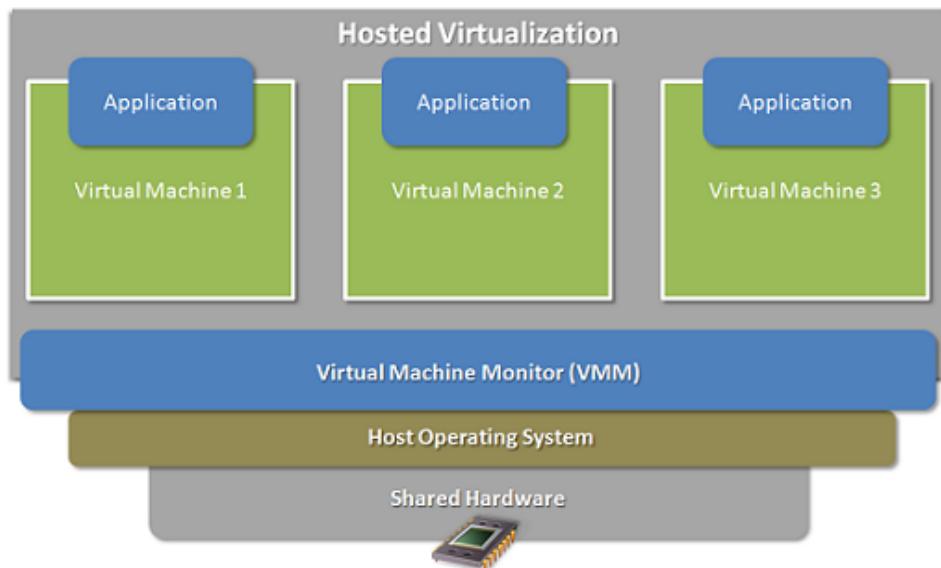
Un altro uso è quello di sfruttare le Virtual Machine per **distribuire in rete sistemi operativi desktop** che vengono utilizzati sui computer degli utenti remoti. Viene realizzata una **Virtual Desktop Infrastructure (VDI)** secondo un modello client-server. Sul server è ospitato un sistema operativo desktop all'interno di ogni macchina virtuale in esecuzione su un server centralizzato.



Si tratta di un eccellente strumento di virtualizzazione dei sistemi, in quanto consente di creare dei **sistemi desktop** completamente **personalizzati per ciascun utente**, gestiti in modo centralizzato in semplicità e sicurezza. Queste tecniche sono spesso usate in **grandi data center** e **organizzazioni di un certo livello**, ma potrebbe andare benissimo anche per le organizzazioni scolastiche perché presentano numerosi PC collegati in rete che utilizzano sistemi operativi e applicazioni diverse.

Hypervisor 2 (Hosted)

Il secondo tipo di architettura si appoggia su un **sistema operativo esistente in grado di ospitarlo (hosted)** e crea **macchine virtuale a più alto livello**. L'**hypervisor 2** fa da cuscinetto tra il sistema operativo già presente e le macchine virtuali che vengono generate e installate sopra di esso.



L'**hypervisor 2**, pur svolgendo le funzioni del tipo 1, **non ha le doti di efficienza e velocità di un sistema dedicato alla virtualizzazione**, perché **si presenta come applicazione utente** e non gestisce direttamente le risorse fisiche dell'hardware. Gli **hypervisor hosted** possono essere installati senza difficoltà sulla macchina preesistente e spesso sono disponibili come prodotti con licenza d'uso gratuita, sia open source, sia commerciali. Ciò consente di usarli, all'interno della stessa macchina, per **provare applicazioni che girano su sistemi operativi differenti** ed essere al riparo dei rischi legati a errori di funzionamento e malware.

Esempi comuni di hypervisor di tipo 2 appartenenti a questa categoria sono [VMWare Workstation](#), [VirtualBox](#) e [Microsoft Hyper-V](#).

La gestione dell'ambiente virtuale

L'host fisico

Gli ambienti di virtualizzazione avanzati sono dedicati alla gestione dei processi relativi alla virtualizzazione e, a seconda del tipo di **hypervisor** usato, permettono o meno l'esecuzione di altri programmi applicativi

Un **hypervisor** di tipo **hosted** non gestisce in modo efficiente la condivisione delle risorse fisiche in quanto viene di fatto gestito dal sistema operativo della macchina ospitante come un'applicazione utente e, l'esecuzione dell'intero processo potrebbe risultare rallentato.

Nei sistemi avanzati, usando un **hypervisor bare metal**, l'**hypervisor** si interfaccia direttamente con l'hardware, gestendo in modo esclusivo i driver delle risorse fisiche e migliorando le prestazioni. Questi ambienti sono dedicati completamente alla gestione dei processi relativi alla virtualizzazione e non permettono l'esecuzione di altri programmi applicativi. Il **sistema operativo** installato sull'host fisico deve essere **minimale**, ridotto praticamente al solo kernel e, deve **interagire direttamente con l'hypervisor**. Le altre funzioni presenti sull'host fisico, accessibili via console o tramite un'applicazione Web, servono unicamente per l'interfaccia amministrativa e per monitorare il sistema, permettendo di effettuare poche e semplici operazioni di base come l'inserimento dei parametri di rete, la regolazione della data e dell'ora, il cambio password e poco altro.

La gestione dello storage

Uno degli elementi critici a cui è necessario porre la massima attenzione è l'organizzazione dello spazio di memoria di massa su cui archiviare i dati, cioè lo **storage**.

In un ambiente non virtualizzato le risorse di memoria possono trovarsi all'interno della macchina server oppure possono essere esterne. In ogni caso il compito di accedere alla memoria di massa è delegato al sistema operativo, che organizza e struttura i dischi tramite la definizione di un file system.

In un **ambiente virtualizzato** l'architettura di storage comprende diversi livelli di astrazione, che nascondono la complessità e i particolari della memoria di massa, permettendo di creare un contenitore logico che offre un modello semplificato e omogeneo per la locazione dello spazio di memoria dedicato a ogni macchina virtuale. Questo modello nasconde la tecnologia usata per lo storage (DAS, NAS o SAN), permettendo di **rappresentare un disco virtuale come uno o più file** che possono essere spostati, aggiunti o sostituiti a caldo senza spegnere la macchina.

Negli **ambienti di virtualizzazione** (*host fisico*) il **filesystem** deve rendere disponibile il disco fisico a più hypervisor contemporaneamente e quindi dovrà godere delle seguenti **caratteristiche**:

- deve essere **orientato alla condivisione**;
- deve permettere un **rapido interfacciamento con l'hypervisor**;
- deve fornire la **sicurezza dei dati** e la piena **indipendenza fra gli ambienti ospitati**;
- deve garantire l'**integrità dei dati**, controllando i corretti semafori per l'accesso condiviso.

CLIL - DAS, NAS e SAN

Watch the two following videos in order to understand the difference among **DAS** (*Directed Attached Storage*), **NAS** (*Network Attached Storage*) and **SAN** (*Storage Area Network*) and evaluate pros and cons of using **NAS** or **SAN**.

- [SAN vs. NAS vs. DAS: Competing or Complementary?](#)
- [NAS vs SAN - Network Attached Storage vs Storage Area Network](#)

Gestione delle ridondanze

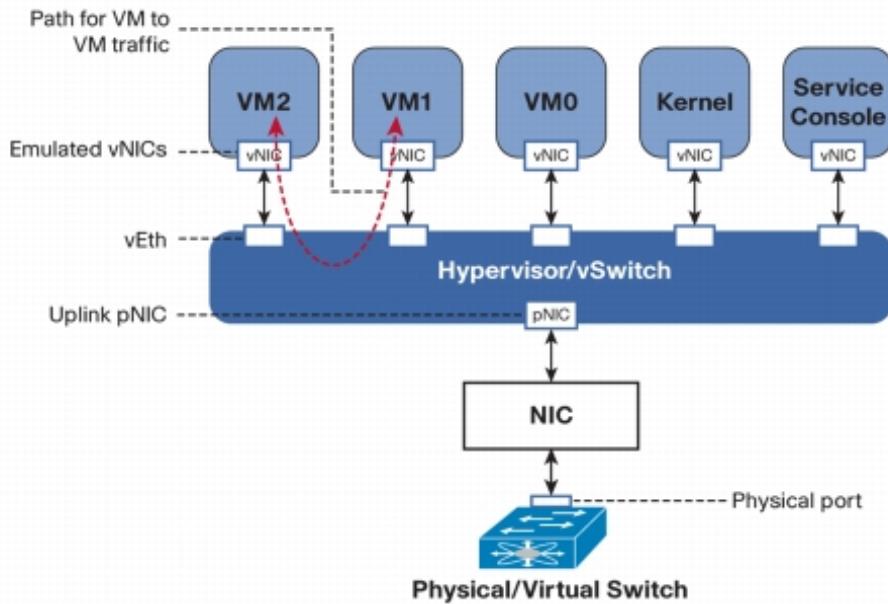
Una adeguata **gestione delle ridondanze dei dati** permette di garantirne l'integrità a fronte di guasti hardware o di attacchi che li compromettano (*business continuity*). I tre link seguenti chiariscono il funzionamento della tecnica RAID utilizzata per aumentare le performance, rendere il sistema resiliente alla perdita di uno o più dischi e poterli rimpiazzare senza interrompere il servizio.

- [RAID - Wikipedia](#)
- [What is RAID 0, 1, 5, & 10?](#)
- [RAID 5 vs RAID 6](#)

Il virtual networking

In una rete fisica le diverse macchine sono collegate tra loro per potersi scambiare dati. Analogamente, in una **rete virtuale** (*virtual network*) l'insieme delle **macchine virtuali**, in esecuzione su una singola macchina fisica, sono **collegate logicamente tra loro** in modo che possano scambiarsi dati.

Una rete fisica è costruita sui dispositivi di rete fisici (schede di rete, router, switch), che permettono l'interfacciamento alla rete e il collegamento tra i terminali. In modo analogo, gli **ambienti virtuali** forniscono, sotto forma di **software**, le funzionalità di **schede di rete virtuale**, **switch virtuale**, **router virtuali**, ecc.



Nell'ambiente virtuale uno **switch virtuale rileva le macchine virtuali** che sono logicamente collegate a ciascuna delle sue porte virtuali e utilizza le informazioni per **indirizzare il traffico**. Questo permette di gestire la connettività a livello 2 del modello ISO/OSI, prevedendo la possibilità di impostare **VLAN multiple secondo lo standard IEEE 802.1q** e permettendo collegamenti multipli sulla medesima porta fisica presente nell'host.

Al fine di garantire la massima disponibilità dei collegamenti lo **switch virtuale** ha la possibilità di **configurare più schede di rete fisiche**, associandole in modalità **LACP** (*Link Aggregation Control Protocol*). In questo modo lo **switch virtuale vede l'insieme delle schede fisiche come un'unica scheda logica**, potendo suddividere il traffico dati fra tutte le schede fisiche, incrementando le prestazioni.

Gli **switch virtuali** devono inoltre poter essere **collegati alle reti fisiche** per comunicare con il mondo esterno. A questo scopo sono utilizzati opportuni **adattatori software** (*uplink port*) associati a quelli fisici, che forniscono una **connessione tra una rete virtuale e una fisica**.

Anche se la **rete virtuale** si comporta alla stessa maniera di una rete fisica, non va dimenticato che la gestione delle risorse dei **componenti virtuali è realizzata via software** e può quindi essere facilmente **distribuita o replicata** su altri sistemi.

Affinché la **rete virtuale** venga gestita in modo efficiente la scheda di rete fisica dovrà essere affidabile e fornire prestazioni adeguate. Potrebbe essere opportuno associare più schede fisiche in modo tale da parallelizzare la trasmissione dei dati e ottenere velocità di trasmissione che sono la somma delle velocità delle singole schede. In questo modo sarà anche aumentata l'affidabilità in quanto, nel caso in cui il collegamento di una scheda venga a cadere, le rimanenti potranno prendere in carico anche i dati della connessione caduta (*business continuity*).

I driver della macchina virtuale

Le macchine virtuali sono a tutti gli effetti dei computer, e come tali hanno delle periferiche proprie. Il **sistema operativo** dell'ambiente virtuale si interfaccia con tutti i dispositivi esistenti sull'hardware virtuale, **installando** i relativi **driver software** per rendere operative le funzioni e ottimizzarne le prestazioni.

I produttori di sistemi di virtualizzazione mettono a disposizione, per i sistemi operativi supportati, un insieme di tool che permettono di effettuare gli aggiornamenti delle macchine virtuali riconoscendo correttamente le periferiche; la mancata installazione può produrre effetti negativi come il non riconoscimento di alcune periferiche con conseguente perdita di prestazioni.

Configurazione dell'hardware virtuale

Una volta preparato l'host fisico, si potrà procedere alla creazione delle macchine virtuali che saranno eseguite su di esso. Al riguardo bisogna ricordare che ogni **risorsa allocata su una macchina virtuale dovrà essere associata a una risorsa fisica**. Di conseguenza sarà possibile allocare dello spazio disco su una macchina virtuale, solo se questo è disponibile realmente sulla macchina fisica. Lo stesso discorso vale per la memoria e le altre periferiche virtuali.

È opportuno valutare con attenzione questo aspetto in quanto **una gestione non oculata delle risorse può causare gravi carenze nelle prestazioni e numerosi disservizi**: un eventuale saturazione delle risorse fisiche produrrà un degrado di tutto il sistema.

Poiché uno dei vantaggi dell'utilizzo degli ambienti virtuali è l'**ottimizzazione dei costi**, è opportuno configurare le macchine virtuali solo con le **risorse effettivamente necessarie**. Allo stesso tempo dobbiamo anche considerare che un **sottodimensionamento di risorse può essere causa di anomalie** che, a loro volta, possono influire negativamente sul complesso dell'intero ambiente virtuale. Ad esempio, se venisse configurata una quantità di memoria insufficiente su di una macchina virtuale tale da indurre il sistema ad usare continuamente porzioni di disco per lo swap, verrebbero aumentate esponenzialmente le operazioni di input/output su disco, influenzando negativamente le prestazioni dell'host fisico e creando rallentamenti su tutte le macchine virtuali ospitate.

Virtual data center

CLIL - Google Data Center

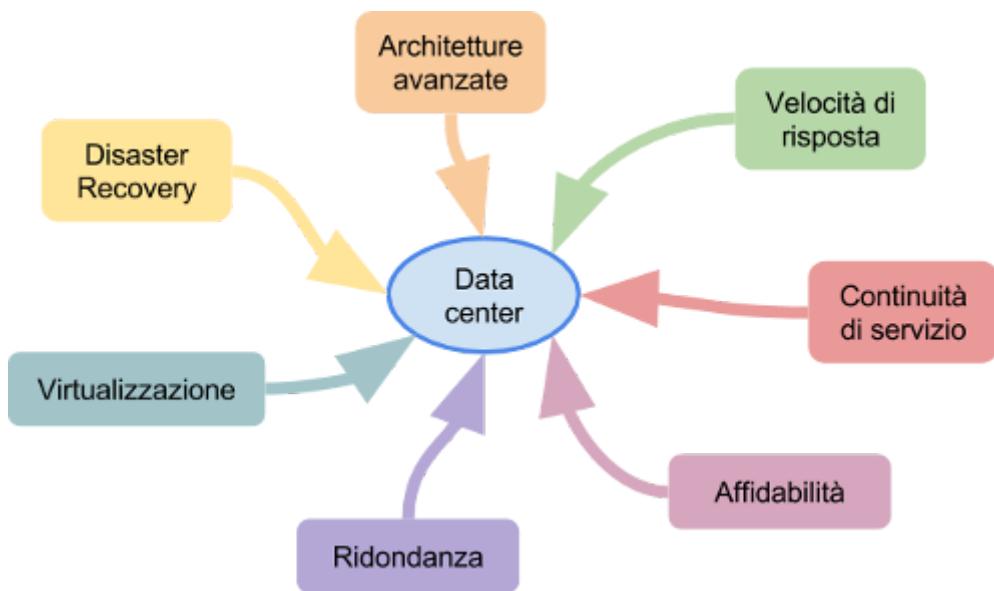
Watch the following video and discuss with your classmates about Data Center.

[Inside a Google data center](#)

I data center

All'interno di un'**azienda** le tecnologie legate all' **Information Communication Technology (ICT)** costituiscono una componente **essenziale**, così come ormai è **indispensabile** mettere in atto **politiche e sistemi** che garantiscano l'**affidabilità** e la **continuità** nell'erogazione dei servizi (*business continuity*). Infatti, se un sistema ha un guasto, le conseguenze possono essere talmente gravi per l'azienda da compromettere le transazioni operative e la produttività.

Per proteggere le apparecchiature è necessario predisporre **ambienti affidabili e sicuri**, dotati di **ridondanza** e sistemi di **backup**, in cui la probabilità di **interruzione del servizio è ridotta al minimo** ed è garantita l'**integrità dei dati**. I **data center** rispondono a queste esigenze offrendo tutte le possibili soluzioni per assicurare standard elevati per il buon funzionamento del sistema.



Un **data center** è un ambiente in cui le aziende e le organizzazioni mantengono le infrastrutture tecnologiche per l'**elaborazione**, la **memorizzazione** e la **trasmissione** delle informazioni.

I data center sono strutturati in funzione dei bisogni aziendali e possono occupare lo spazio di un armadio (*rack*) oppure estendersi fino a raggiungere dimensioni tanto grandi da consumare l'energia di una città.

Possiamo catalogare i data center in due grandi famiglie:

- **data center privati**: sono normalmente **proprietà di una singola azienda** e forniscono servizi relativi all'infrastruttura interna; vengono utilizzati per la gestione dei dati aziendali, per la gestione dei flussi amministrativi e produttivi e per la gestione intranet ed extranet;
- **data center pubblici**: sono **strutture specializzate nella gestione di servizi multipiattaforma accessibili in Internet**. I servizi spaziano dalla fornitura di applicazioni Web alla gestione della posta elettronica o alla gestione in [hosting](#) dei flussi propri di un'azienda, compresi i [server virtuali](#) a uso privato ([Hosting, Housign, Server Virtuale... o Cloud?](#))

Un **data center** è incentrato su:

- una **infrastruttura operativa** (*white space*), che ospita i **rack** contenenti i **server** di elaborazione, i **sistemi di archiviazione** e gli **apparati per la trasmissione** dei dati sulla rete, tutelati da sistemi di **sicurezza** di alto livello;
- le **infrastrutture di supporto**, costituite da dispositivi di **backup** e sistemi ridondanti che garantiscono l'**affidabilità** il **funzionamento continuativo** anche in caso di guasto o catastrofe naturale;
- il **personale operativo**, costituito dalle **figure professionali** che gestiscono il centro di elaborazione: sono suddivise in amministratori di rete, tecnici informatici, sistemisti, programmati, addetti alla sicurezza e manutenzione.

Virtual data center

Dal data center al virtual data center

Come per le singole macchine, anche per i **data center** è in atto la tendenza a passare dai sistemi fisici a quelli **virtuali**, amplificandone le funzioni.

Un **virtual data center** è un data center che opera sfruttando la tecnologia della virtualizzazione, offrendone i servizi e le risorse alle aziende che li richiedono.

I **vantaggi** dell'impiego di un **virtual data center** da parte di un'organizzazione sono diversi:

- **diminuzione degli investimenti in termini di hardware** e conseguente diminuzione dello spazio necessario al contenimento delle apparecchiature fisiche;
- **riduzione dell'energia consumata**;
- **condivisione delle risorse** con altre organizzazioni;
- possibilità di disporre di **sistemi scalabili** in funzione delle esigenze aziendali;
- fruibilità di **servizi in cloud computing**.

I **possibili rischi** di questa soluzione non sono però da trascurare, infatti portare tutti i servizi su un'unica struttura determina un punto di criticità, detto **single point of**

failure (*SPoF*) perché, in caso di guasto, si avrebbe la paralisi completa dei sistemi informativi.

Il **malfunzionamento** di un qualsiasi elemento può causare una **indisponibilità di servizio** (*downtime*) con ripercussioni economiche anche gravi. Se, ad esempio, pensiamo ad un sistema di server utilizzato da una società di e-commerce, è semplice intuire che ogni minuto di fermo-macchina comporta una notevole perdita economica. E' quindi di importanza vitale **garantire un'altra disponibilità di servizio**, detto in termini tecnici ***high availability***.

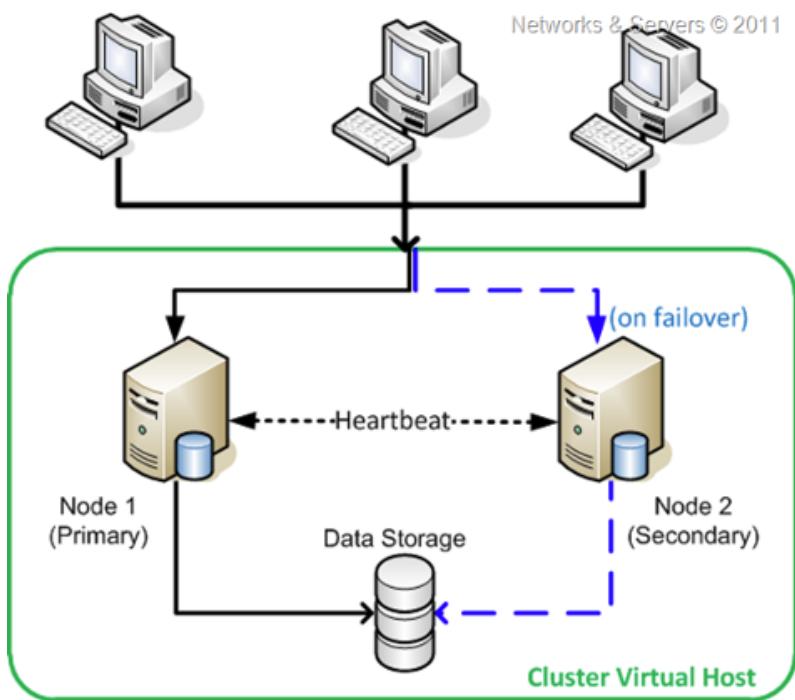
Per questa ragione, un ambiente di virtualizzazione ben progettato prevede **tecniche di ridondanza, duplicazione, cluster di host, sistemi di archiviazione**, studiati per **ridurre al minimo le conseguenze di guasti e blocchi inaspettati**, fino ad arrivare alla **replica dell'intero data center** (*business continuity*).

Cluster di failover

Uno degli aspetti fondamentali che permettono di **sopperire ai fermi macchina** è l'utilizzo di un **cluster di failover**. Questo termine identifica un **gruppo di host** che lavorano insieme per **mantenere al massimo livello la disponibilità dei dati e dei servizi, reindirizzando le richieste da un server all'altro**, permettendo così agli utenti di **accedere costantemente alle risorse** presenti nel server.

Nel caso di **malfunzionamento di una macchina fisica**, il gestore del virtual data center è in grado di **individuare il guasto, analizzare i carichi di lavoro** delle macchine fisiche e, dopo aver verificato che ci sono risorse disponibili, **spostare le virtual machine** presenti nell'host danneggiato, **sugli altri host fisici**, facendole poi ripartire.

Gli **host di un cluster sono collegate tra loro tramite una rete privata** (*heartbeat network*, la rete di importanza vitale) **non accessibile all'esterno del cluster**, che permette lo scambio continuo di informazioni per il monitoraggio del sistema.



Migrazione di macchine virtuali

La **migrazione di macchine virtuali** (*live migration*) permette di **spostare una macchina virtuale completa in esecuzione, da un host fisico a un altro**, senza

alcuna interruzione nel funzionamento. La procedura in realtà non avviene istantaneamente, ma occorre un certo tempo, seppur breve, per trasferire lo stato di esecuzione della macchina virtuale e della sua memoria in un altro host fisico.

Per assicurare l'integrità delle transazioni la macchina virtuale originaria viene sospesa solo dopo aver copiato l'intera memoria e lo stato del sistema sull'host fisico di destinazione. Il **tempo di fermo macchina** (*downtime*) e **dell'ordine di pochi secondi**. Per l'utente il trasferimento risulta completamente trasparente.

Migrazione dello storage

La **migrazione dello storage in cui si trovano le macchine virtuali**, permette di spostare i file da uno storage all'altro (o all'interno dello stesso storage) **senza interruzioni di servizio** (*live*).

Questa operazione è utile ad esempio nel caso di **manutenzione** che viene effettuata in seguito al **guasto di unità disco** appartenente ad una configurazione RAID, oppure in caso di **aggiornamento dell'hardware**. La migrazione può essere effettuata anche per **ottimizzare i carichi** dei dischi che stanno per raggiungere il limite di saturazione.

Il processo di migrazione prevede la copia dei blocchi di memoria (*mirror*) dal disco sorgente in quello di destinazione, controllando che durante l'operazione di trasferimento non vi siano state **modifiche ai blocchi sorgente**, diversamente il disco **sorgente** e quello **destinatario** risulterebbero **disallineati**, con conseguente necessità di una **nuova sincronizzazione**.

Snapshot, clonazione e template

Lo **snapshot** è una funzione che consente di **catturare lo stato di una virtual machine** così com'è nel momento in cui la funzione di snapshot è eseguita.

È utile per **definire dei punti di ripristino** da utilizzare quando abbiamo la necessità che la **macchina torni nello stato precedente a un determinato evento**, per esempio, quando si devono fare modifiche o aggiornamenti critici alla virtual machine e, temendo di sbagliare o di determinare situazioni non volute, ci si tutela creando uno **snapshot** che consenta di tornare allo stato precedente la modifica.

Durante le operazioni di **snapshot** vengono **impedite le operazioni di scrittura su file della virtual machine** e vengono invece **creati dei file paralleli**, detti *transazionali*, su cui vengono memorizzate le operazioni effettuate.

Il **ripristino della macchina virtuale da uno snapshot** comporta necessariamente la **perdita dei dati memorizzati successivamente** all'attivazione e potrebbe causare situazione di inconsistenza, soprattutto sul server dedicati alla gestione di file condivisi, database o server di email.

Lo **snapshot** non va considerato come una forma di backup e richiede solitamente una certa occupazione sul disco.

Con la **clonazione** possiamo **duplicare una virtual machine** ottenendone un'altra con la medesima configurazione. Questa tecnica è **utile**, ad esempio, per poter effettuare dei **test e paralleli** all'ambiente di produzione e creare **server di backup**. La macchina clone dovrà essere opportunamente configurata e personalizzata per non creare conflitti con la macchina clonata.

Il **template** è una tecnica che permette di **accelerare le operazioni di creazione di macchine virtuali**, utilizzando il template come **modello** di duplicazione per la generazione di macchine simili, che andranno personalizzate all'atto della creazione.

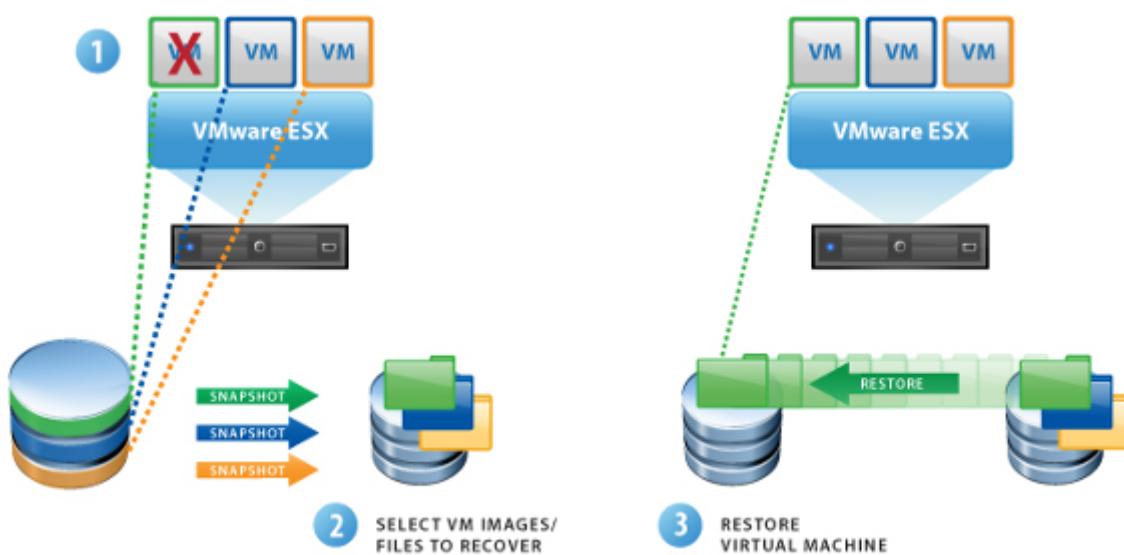
Backup del data center virtuale

Un aspetto molto importante di un qualsiasi sistema informatico è la gestione dei **backup**, cioè la realizzazione di **copie di salvataggio che permettono il ripristino dei dati in caso di perdita accidentale di informazioni**.

Negli ambienti virtuali è possibile attivare una **procedura di disaster recovery** che consente di **ripristinare l'intero server sul sistema remoto**, partendo da una copia di backup che contiene lo stato del sistema, dati compresi.

Alcuni gestori di backup di ambienti virtuali sono in grado di copiare le intere virtual machine su supporti esterni, visibili ai computer come una risorsa di rete. Il **ripristino** dei dati può avvenire **integralmente**, ricreando la virtual machine originale dalla copia, **oppure** può essere **parziale e selettivo**, individuando semplicemente i file e le cartelle di cui viene richiesto il ripristino.

Anche la creazione di **ambienti di disaster recovery online**, in grado di entrare in funzione immediatamente nel caso di crash del data center, risulta essere molto utile. In questi casi viene **ricreato remotamente una struttura di host fisici e storage** simile a quella utilizzata nel data center principale. I software di backup permette di **replicare in remoto l'intero ambiente virtuale**, in modo che tutte le attività possono essere reindirizzate tempestivamente su server remoti.



Cloud computing

Il **cloud computing** è un servizio che permette di archiviare ed elaborare le informazioni, e sfruttare applicazioni e risorse software messe a disposizione dal fornitore di servizi Internet.

Le due entità del cloud computing sono: il **fornitore** che mette a disposizione dei servizi fruibili da qualsiasi luogo in qualsiasi momento, e il **cliente** che accede da remoto senza la necessità di doversi necessariamente poggiare a risorse interne dell'azienda, risparmiando sull'infrastruttura e sui costi di manutenzione e aggiornamento di sistemi.

Le **infrastrutture di cloud computing** sono concentrate in grandi **data center** che, basandosi sulla **virtualizzazione delle risorse** sfruttano le caratteristiche di **ridondanza e disponibilità** per offrire un **servizio continuativo** all'utente, **riducendo** nel contempo i **costi** di esercizio e di **manutenzione**.

Il **cloud computing** offre il vantaggio del *self serving*, cioè la possibilità di **ottenere esclusivamente le risorse che sono richieste (on demand)**, pagando solo i servizi realmente utilizzati.

Il **NIST** (*National Institute of Standards and Technology*) ha elaborato un [documento ufficiale](#) che contiene il modello che **definisce la struttura del cloud computing**. In questo documento vengono definite le *cinque caratteristiche* che definiscono **come sono strutturati i servizi del cloud**, i *tre modelli* che rappresentano **che cosa è erogato** e i *quattro tipi* che precisano **chi usufruisce dei servizi e dove vengono erogati**.

Le cinque caratteristiche del cloud

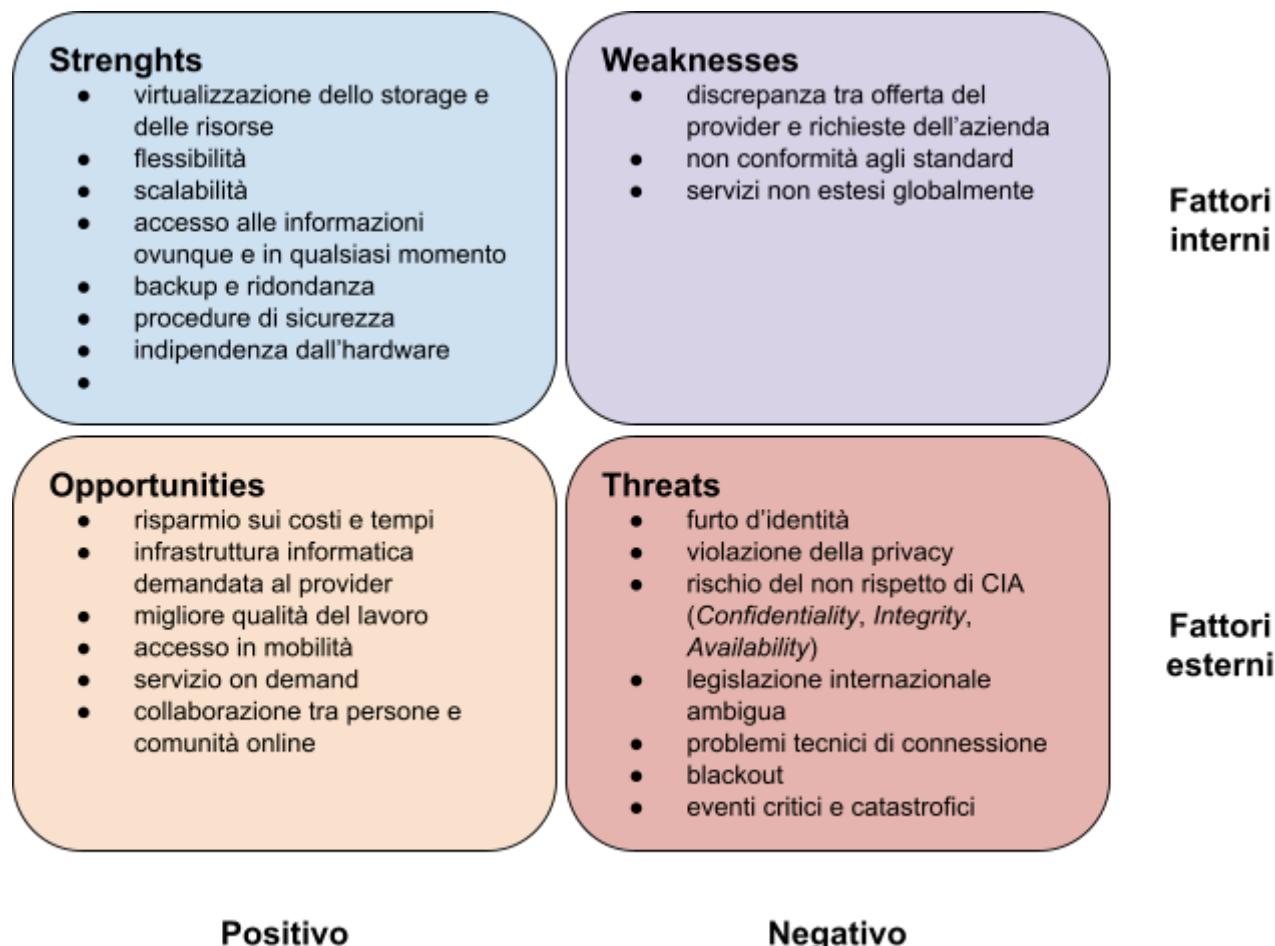
Il cloud è contraddistinto da almeno *cinque caratteristiche* che identificano **come deve essere un servizio**:

1. **On demand self-service:** un utente può **accedere autonomamente ai servizi** offerti da un fornitore di servizi (capacità di elaborazione e storage) senza la necessità di un intervento umano dei gestori.
2. **Broad Network Access:** le **funzionalità del cloud** devono essere **disponibili** in rete tramite piattaforme accessibili **da sistemi eterogenei fissi e mobili**: telefoni cellulari, tablet, computer portatili e workstation.
3. **Resource pooling:** le **risorse** di elaborazione, fisiche e logiche, fornite dal provider (larghezza di banda, potenza di calcolo, applicazioni) vengono **organizzate** in insiemi di servizi e **assegnate** e riassegnate **dinamicamente** in funzione delle esigenze degli utilizzatori.
4. **Rapid elasticity:** le **funzionalità** possono essere **assegnate e rilasciate elasticamente e rapidamente** con **costi commisurati alle risorse effettivamente utilizzate**. L'utente ha la sensazione di disporre di capacità illimitate che possono essere stanziate in qualsiasi momento e ovunque si trovi.

5. Measured service: ogni **servizio** deve poter essere **misurato e controllato in modo trasparente**, ciò permette di ottimizzarne l'uso e calcolare consumi.

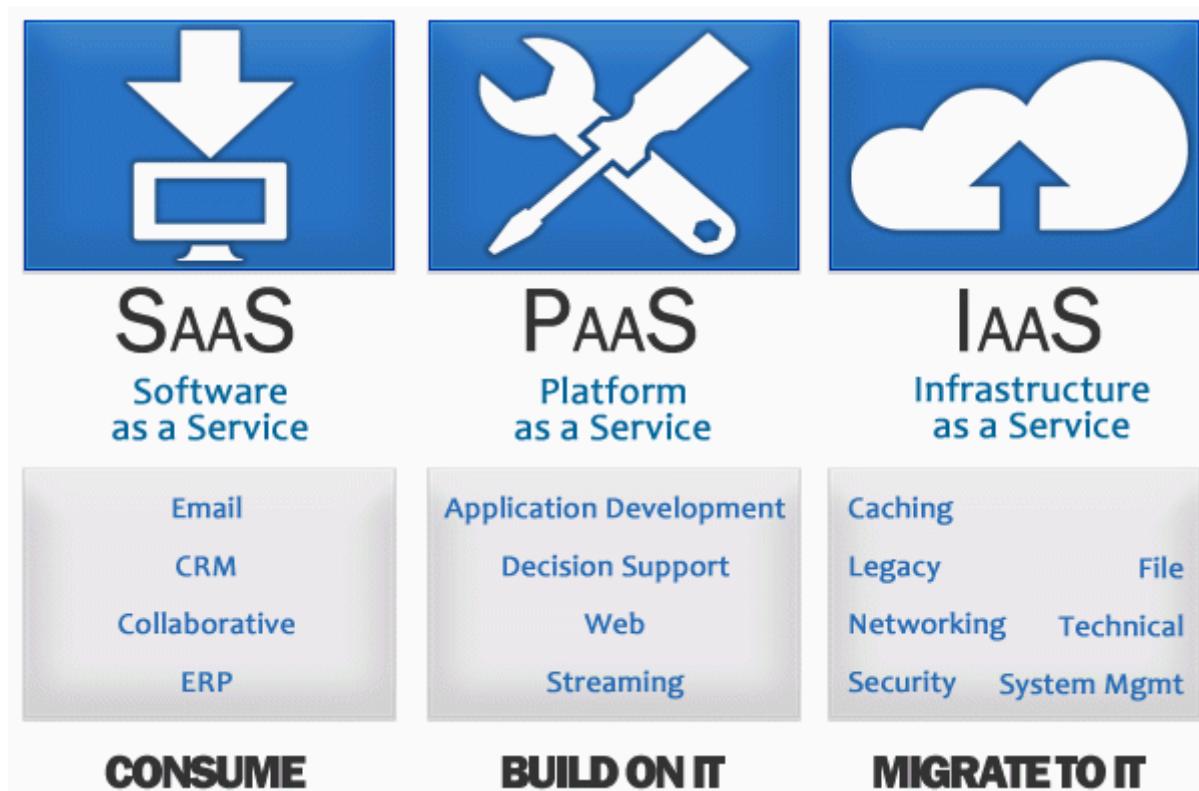
Come si vede l'elemento caratterizzante è l'**elasticità** che, plasmando i servizi, procura indubbi vantaggi che vanno dalla possibilità di **demandare** al fornitore di servizi le **infrastrutture informatiche** (*outsourced IT*), alla **scalabilità delle risorse**, ai **costi legati agli effettivi consumi**, all'**indipendenza delle piattaforme**.

Gli **svantaggi** principali riguardano la **dipendenza dalla rete Internet** (con il rischio di blackout o di caduta delle prestazioni), possibilità di **violazione della privacy**, rischio di **manipolazione, furto dei dati e spionaggio, legislazioni differenti** a seconda del paese.



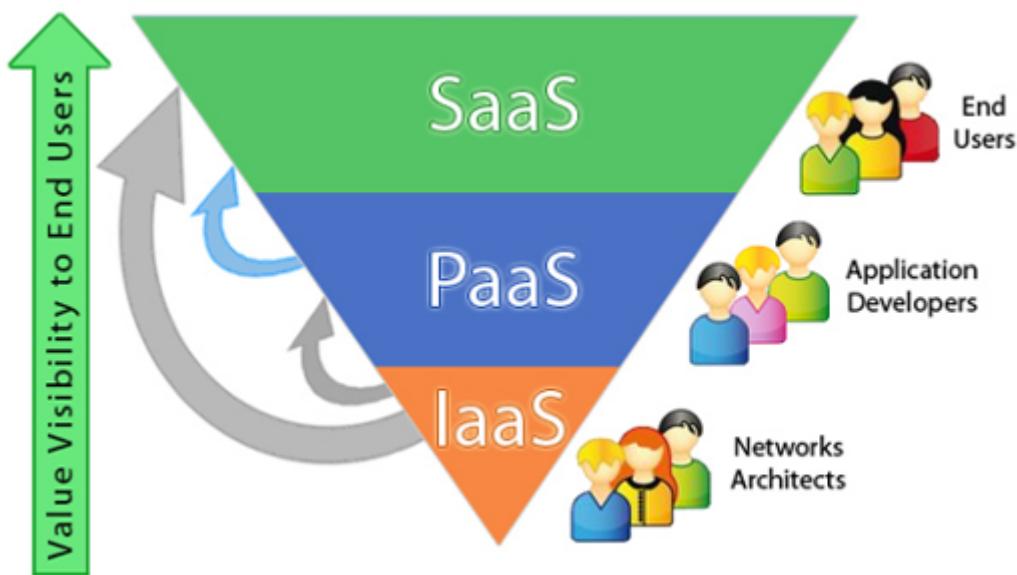
I tre modelli di servizi del cloud

Il [documento](#) del **NIST** specifica **che cosa deve erogare il cloud** viene definito dai *tre modelli di servizi* offerti dal cloud:



1. **SaaS (Software as a Service)**: è un **modello di distribuzione del software applicativo** in cui un utente, servendosi di un browser, utilizza l'applicazione che **il provider mette a disposizione**, ma non ha il potere di controllare l'infrastruttura del cloud (server, sistemi operativi, storage) né di gestire le funzionalità delle applicazioni, può solo, eventualmente, configurarle e inizializzarle. Esempi di **SaaS** sono *Google Drive*, *Dropbox*, *Box.net*. Queste applicazioni, in parte gratuite, offrono servizi di archiviazione, sincronizzazione e condivisione di documenti. La sicurezza è garantita dalla crittografia, spesso utilizzando il protocollo **TLS** per il trasferimento dei dati. I file archiviati, accessibili tramite password, sono cifrati tramite AES (*Advanced Encryption Standard*), un algoritmo di cifratura a chiave simmetrica a 256 bit, che opera su un gruppo di bit di lunghezza finita organizzati in blocchi, o altri algoritmi di crittografia simmetrica con un livello di sicurezza superiore.
2. **PaaS (Platform as a Service)**: il servizio mette a disposizione un'**intera piattaforma di elaborazione**, dotata di **sistemi operativi** e **linguaggi** diversi **per lo sviluppo di applicazioni** e il **salvataggio dei dati**. L'utente non si deve preoccupare del mantenimento dell'infrastruttura hardware e software. Rientra in questa categoria *Google Cloud Platform*.
3. **IaaS (Infrastructure as a Service)**: in questo modello viene **fornita l'intera infrastruttura informatica**. Il cliente può tenere intere macchine virtuali

compreensive di sistemi di storage e componenti di rete, per poter autonomamente erogare le proprie applicazioni e propri servizi.



Oltre ai tre principali modelli, talvolta ne viene aggiunto un quarto il **DaaS** (*Data as a Service*), che **mette a disposizione i dati dell'utente come se fossero presenti sul disco locale**. In collaborazione con SaaS, **DaaS** offre la possibilità di realizzare servizi in rete per la memorizzazione, l'elaborazione e la condivisione dei dati in vari formati.

Esempi di **PaaS** e **IaaS** sono *Amazon Elastic Compute Cloud* (Amazon EC2) e *Microsoft Azure*, entrambi molto flessibili nella configurazione e scelta dei servizi, oltre ai sistemi operativi e i linguaggi di programmazione utilizzabili, pagando in base al consumo.

CLIL - Saas, Paas and Iaas

Watch the following videos and discuss with your classmates about Saas, Paas and Iaas.

- [Cloud Computing Services Models - IaaS PaaS SaaS Explained](#)
- [The Three Ways to Cloud Compute](#)

Cloud privato, pubblico, comunitario e ibrido

Il [documento](#) del **NIST** specifica quali sono i **servizi** essenziali offerti dal cloud computing:

1. **Private cloud:** i **servizi cloud** sono **riservati a una specifica azienda** che comprende più utenti. L'infrastruttura cloud può essere di proprietà dell'organizzazione o di un provider esterno e può trovarsi dentro o fuori della sede aziendale.
2. **Community cloud:** i **servizi cloud** sono riservate ad una **comunità di utenti** che fanno parte di **organizzazioni che condividono gli stessi interessi** (ad

esempio la politica aziendale, gli obiettivi, i requisiti di sicurezza). Può essere di proprietà dell'organizzazione o di un provider esterno.

3. **Public cloud:** i **servizi cloud** sono **aperti al pubblico che accede tramite Internet**. È di proprietà di un provider (oppure di un ente accademico governativo), che fornisce la piattaforma e le applicazioni condivise con più clienti.
4. **Hybrid cloud:** i **servizi cloud** sono una **composizione di privato e pubblico** che vengono **usati in modo ibrido**. Ad esempio, servizi privati per la conservazione dei dati e servizi pubblici per l'accesso alle applicazioni. In alcuni casi la modalità ibrida è utilizzata anche per bilanciare il carico tra le diverse infrastrutture di cloud.

CLIL - Public, Private and Hybrid Cloud

Watch the following video and discuss with your classmates about Public, Private and Hybrid Cloud.

[Public Cloud vs Private Cloud vs Hybrid Cloud](#)

Cloud computing pubblico

Il tipo di **cloud** più conosciuto è indubbiamente quello **pubblico**. Questo tipo di cloud si basa sul modello standard in cui un **service provider rende disponibili su internet le risorse di elaborazione e di storage**. Di solito si ottengono elevate prestazioni e un prezzo competitivo senza l'aggravio dei costi di manutenzione, formazione del personale, licenze software, gestione degli spazi, ecc.

Il **cloud pubblico** è per sua natura molto flessibile e questo lo rende un'ottima soluzione soprattutto per le aziende che non possono permettersi un data center: attività come il controllo della sicurezza, il salvataggio dei dati, il backup, lo smaltimento dell'hardware usato costituiscono un costo notevole che, se è ripartito su molti utenti, contiene i costi senza far scadere le prestazioni.

D'altra parte, l'impossibilità di avere un controllo diretto sull'infrastruttura crea negli utilizzatori il timore, razionale meno, della perdita e dell'inquinamento dei dati.

Cloud computing privato

Il **cloud privato è riservato alle aziende** che di solito lo utilizzano come **data center virtuale** interno all'azienda o direttamente gestito da essa.

Il **cloud privato**, almeno da un punto di vista logico, consente di **mantenere i dati dentro la propria struttura operativa**, risolvendo la questione riguardante il problema della **privacy** e della **sicurezza**, che costituisce un punto di svantaggio dei cloud pubblici.

Il principale vantaggio di questo modello è la possibilità di **ottimizzare le risorse**, che vengono scalate in modo rapido a seconda delle necessità e della configurazione desiderata.

Un'azienda (solitamente di una certa dimensione) scegli questo modello quando, oltre a garantirsi la sicurezza e la privacy, ha la necessità di sfruttare al meglio le risorse e salvare gli investimenti.

Cloud computing ibrido

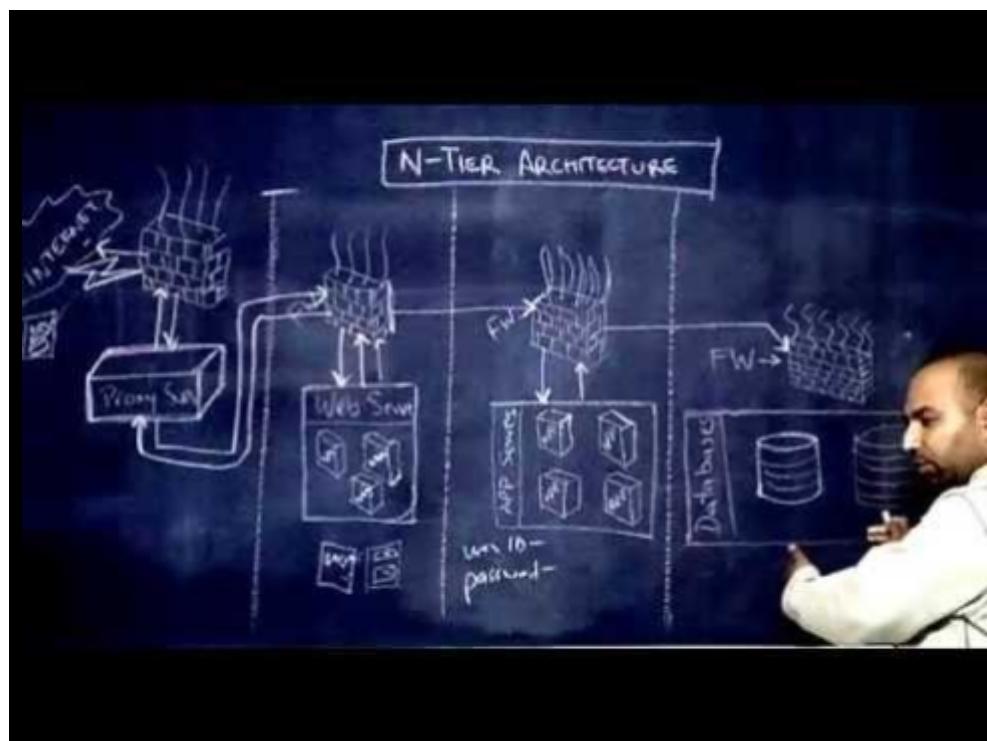
Il **modello ibrido** cerca di prendere il meglio dei modelli **pubblico** e **privato**. L'infrastruttura, ripartita tra interno ed esterno, consente di condividere le risorse tra le due entità.

Non è affatto semplice ripartire servizi tra pubblico e privato. Oltre alla difficoltà di definire le politiche di bilanciamento delle risorse per guadagnare in efficienza, uno dei rischi più grossi è quello di avere piattaforme gestite diversamente, con problematiche e interfacce dissimili, creando nell'utenza disorientamento e confusione.

Nonostante questi problemi di integrazione, **molte aziende si stanno orientando verso una soluzione ibrida**, soprattutto perché in questo modo è possibile mantenere la parte di **informazioni strategiche all'interno**, lasciando all'**esterno le attività ritenute meno critiche** a cui si può accedere con modalità meno stringenti, comodamente, da qualsiasi luogo e in ogni momento.

Approfondimenti

A - Preparazione all'Esame di Stato



Watch the video, it will help!

Requisiti fondamentali di Progettazione reti

Per affrontare con competenza la seconda prova scritta di Sistemi e Reti del corso di Informatica occorre costruirsi mentalmente uno schema dei passaggi logici che, dalle specifiche dei requisiti presenti in modo più o meno esplicito nelle richieste della prova, conducano alla progettazione della rete e al relativo software di gestione.

Spesso la prova che viene assegnata impone allo studente di avere una visione globale e aperta del sistema. I casi proposti partono da situazioni reali che vanno analizzate, interpretate e sintetizzate e nelle quali occorre integrare le varie parti, in modo che LAN, WAN, applicazioni software, database e sicurezza possano essere integrati in un unico sistema capace di rispondere alle richieste del testo.

È utile strutturare il progetto basandosi su un'impostazione che presenti uno schema generale a cui aggiungere la propria esperienza maturata durante gli anni scolastici, ottenendo così una giusta sintesi di competenze acquisite sia in ambito formale, qual è quello scolastico, sia in ambito informale.

Occorre anche evidenziare che spesso le tracce d'esame proposte dal Ministero presentano aspetti della realtà descritti nel testo non univocamente interpretabili. In questi casi è necessario formulare delle ipotesi ragionevoli, né troppo limitative né troppo complesse, che consentano di comprendere e circoscrivere il tema proposto.

Inoltre, quando si formulano delle ipotesi aggiuntive, devono essere realistiche e legate al caso specifico.

Nel caso in cui le richieste siano generiche, occorre fare scelte ponderate, in linea con il programma svolto durante l'anno, tenendo conto del tempo a disposizione. La realizzazione di una pagina Web o del codice a essa associato deve semplicemente fornire una traccia, o dei casi prova, che dimostrino la competenza raggiunta, senza cadere nell'errore di pensare alla realizzazione di un progetto completo e dettagliato.

Che cosa significa progettare una rete

Il percorso da seguire per la progettazione di una rete (anche in funzione della sua dimensione) è quello classico mostrato in figura.



In essa sono rappresentati gli elementi attraverso i quali si sviluppa l'analisi del progetto di rete e che si andrà ad analizzare in dettaglio nei prossimi paragrafi:

- Definizione dei requisiti
- Progettazione e pianificazione
- Realizzazione del progetto
- Gestione e controllo del lavoro
- Rilascio del sistema e proseguimento

Definizione dei requisiti

La comprensione dei requisiti permette di definire:

- Numero di nodi e utenti (dimensionamento della rete)
- Topologia, modalità di accesso, disposizione geografica dei nodi e delle aree da collegare
- Tipo di traffico (dati, voce, video) e banda a disposizione
- Modalità e tipologia della connessione a Internet
- Vincoli (architettonici, tecnologici, legali, di spesa)
- Livello di sicurezza e policy
- Espansione e modifiche previste nel tempo
- Know-how del personale addetto alla gestione della rete

Progettazione e pianificazione

La progettazione dell'architettura di rete deve confrontarsi non solo con vincoli di natura tecnica ed economica, ma deve anche considerare le esperienze e la cultura del personale.

In particolare, deve tener conto delle infrastrutture di supporto alla rete ed eventualmente dell'inserimento nel progetto della loro modifica o realizzazione. Pertanto si dovranno considerare:

- vincoli della struttura (passaggi, canaline, montanti, scavi tra edifici, antenne, accessi protetti, nuove costruzioni)
- punti di alimentazione elettrica (potenza, ridondanza, topologia)
- controllo temperatura/umidità degli ambienti con sistema di condizionamento, se necessario
- controllo perimetrale e protezione degli accessi ai locali dedicati
- obblighi di legge per le norme antincendio e di sicurezza sul lavoro e quanto altro viene richiesto dall'organizzazione interna all'azienda
- esperienza, autonomia e cultura tecnica del personale e di eventuali consulenti

Realizzazione

La realizzazione di una rete deve seguire i principi del Project Management, quali:

- un progetto tecnico di realizzazione dettagliato (blueprint)
- un piano dettagliato temporale con la descrizione delle parti realizzabili in parallelo (diagramma di Gantt) e le scadenze intermedie ("milestone")
- un controllo accurato del rispetto dell'avanzamento del progetto durante la sua realizzazione
- il rilascio di mappe dettagliate della realizzazione su supporto informatico in formato digitale
- il rilascio dei certificati di legge
- un piano di prove per l'accettazione delle varie componenti della rete
- le fasi di messa in funzione e collaudo

Gestione e rilascio

Una volta progettata la rete è necessario attivare un piano di follow-up per un controllo costante del funzionamento, al fine di correggere eventuali errori di progettazione. È necessario poi seguire costantemente l'evoluzione del suo utilizzo e l'evoluzione tecnologica per mantenere sempre aggiornata la rete.

In particolare occorre tenere sotto controllo due aspetti:

- **Problemi organizzativi e di coordinamento (cose e persone):** se è stata fatta una buona pianificazione delle attività da fare e se per ognuna di queste sono stati definiti chiaramente la responsabilità e il periodo di esecuzione, si dovrebbero avere pochi problemi organizzativi e di coordinamento; le fasi di definizione degli obiettivi e di pianificazione sono le attività più importanti nella gestione di un qualsiasi progetto.

- **Difficoltà, soddisfazioni e errori:** le difficoltà si superano facilmente se ci sono buona intesa e buono spirito di collaborazione tra gli stakeholders; certamente occorre prevedere nel piano di lavoro qualche possibile ritardo nell'esecuzione dei lavori, dovuto a volte a banalità o a fraintendimenti; per questo è indispensabile fare una buona pianificazione e una previsione dei rischi.

Traccia per la progettazione di una rete

I punti che seguono forniscono una traccia che facilita lo studente nell'elaborazione del progetto di rete che, tipicamente, costituisce la spina dorsale del tema d'esame.

Inquadramento generale

- Ambito
- Contesto
- Modalità di finanziamento e costi (specificare l'entità dei costi, la fonte, le modalità di finanziamento)
- Dimensionamento dell'utenza effettiva
- Modalità di fruizione dei servizi offerti
- Filosofia di impiego
- Soluzioni organizzative (specificare quali figure professionali sono coinvolte nell'uso della rete)
- Modalità di accesso (libero, regolamentato, con policy, ...)
- Analisi bisogni/benefici
- Riferimenti al modello ISO/OSI

Vincoli e progetto della rete

- Integrazione con eventuali altre reti preesistenti
- Vincoli
- Distanze (comprensorio, edificio, ...)
- Dimensionamento: specificare che cosa si vuole collegare in rete (aree funzionali, numero di host fissi, numero di host mobili wireless, numero di stampanti, server, sottoreti di interconnessione, domini di accesso, ...)
- Schema della topologia della rete
- Apparati di rete, la cui scelta deve tener conto di alcuni parametri fondamentali:
 - velocità e numero di porte di interfaccia presenti su uno switch o su un router
 - costo di un dispositivo (in funzione delle sue caratteristiche: velocità, tecnologia, sicurezza, opzioni, ridondanza, affidabilità, ...) e dei cavi di collegamento
 - scalabilità con la possibilità di inserire nuovi moduli e di espandere il numero delle porte in modo da rispondere a nuove esigenze e collegamenti con nuove reti
 - caratteristiche di funzionamento e servizi offerti: sicurezza, qualità del servizio (QoS), per classificare il traffico distinguendo, per esempio, quello per applicazioni di streaming in tempo reale e VoIP (Voice over IP)

- NAT, DHCP.
- Indirizzamento (reti/sottoreti/host). Ricordiamo che tutti gli host all'interno di una rete devono avere un indirizzo univoco. Lo schema di indirizzamento IP deve essere pianificato, documentato e conservato. Gli indirizzi possono essere privati e pubblici e l'assegnazione può avvenire manualmente (come si fa di norma con server o dispositivi intermedi), oppure tramite DHCP per i terminali utenti che possono essere abilitati ad accedere a Internet. Se si usano procedure standard (per esempio l'assegnazione di indirizzi prestabiliti ai server o a certi insiemi di terminali), non solo è più facile per l'amministratore di rete risalire a errori e malfunzionamenti, ma anche rilevare intrusioni ed essere pronti a intervenire in caso di attacco.
- Virtual Lan

Confine con l'esterno

- Connessione con la rete Internet
- Velocità di connessione
- Configurazione router: IP pubblico, range di IP privati (DHCP), IP del default gateway, maschere
- Configurazione VPN
- Configurazione firewall e proxy
- Configurazione NAT

Cablaggio strutturato

- Distanze
- Scelta dei mezzi trasmissivi
- Velocità richieste
- Riferimenti alle normative internazionali
- Schema logico (dorsali, concentratori, uscita EF)

Sistemi di sicurezza adottati

- Controllo degli accessi e contrasto delle minacce:
 - minacce ambientali, con temperature e umidità fuori norma
 - minacce elettriche ed elettromagnetiche, con variazioni di tensione, scariche elettriche, disturbi radio
 - minacce ai sistemi hardware, con danni ai server e agli apparati di rete e al cablaggio
- Documento delle "norme che regolano l'accesso e l'uso" (policy)
- Individuazione dei responsabili
- Definizione dell'ambiente di prova e di produzione
- Backup e aggiornamenti del software
- RAID (Redundant Array of Independent Disks) per garantire la tolleranza ai guasti
- Ridondanza, necessaria per garantire l'affidabilità alla rete in modo da sopperire a eventuali guasti. Per questo è consigliata la duplicazione dei dispositivi critici, come switch e router e il collegamento con la rete esterna

- Authentication, Authorization and Accounting: servizi utilizzati per l'identificazione di un utente, l'autorizzazione alle attività da svolgere in osservanza alle regole di policy, la misura delle risorse che l'utente consuma durante l'accesso
- Firewall (con filtraggio dei pacchetti, applicazioni, URL, Stateful Inspection)
- Proxy
- DMZ
- VPN, protocolli sicuri (HTTPS, SSL, ...), crittografia e certificati, SSH
- Log degli accessi e degli errori

Gestione dei servizi

- Server interni/esterni
- Sistemi operativi
- Servizi forniti: DBMS, HTTP, SMTP, POP3, ...
- Applicazioni client-server
- Hosting/Housing
- Cloud/Virtualizzazione

Gestione della rete

- Gestione utenti
- Amministratore di rete
- Manutenzione hardware e software
- Responsabili dei dati sensibili
- Analisi del traffico tramite l'uso di applicazioni e protocolli (SNMP), in modo da poter intervenire tempestivamente in situazioni critiche e ottenere informazioni sulla vulnerabilità dei sistemi e sulla possibilità di contrastare eventuali attacchi
- Documentazione delle operazioni svolte

Sviluppo applicazioni

Le applicazioni si sviluppano su tre livelli.

- Modello dei dati:
 - definizione del modello concettuale (per esempio modello E/R)
 - modello logico (per esempio modello relazionale)
- Logica di funzionamento:
 - realizzazione e codifica delle funzioni che operano sui dati (SQL, PHP, ASP, ...)
 - architettura client-server (Java, .NET)
- Interfaccia utente:
 - analisi di usabilità e accessibilità
 - interfaccia per applicazioni Web e/o per sistemi mobili (tipicamente HTML5, CSS, JavaScript, ...)

Prove d'esame

Scritto

A questo [link](#) è possibile recuperare i testi degli Esami di Stato degli anni passati.

In rete è possibile trovare anche molte soluzioni, ma si consiglia di provare prima ad ideare una propria soluzione per poi confrontarla con le soluzioni proposte, analizzandole ed evidenziando tutti i punti che non si conoscono, elencandoli in una lista, ed iniziando a ricercare sistematicamente informazioni al riguardo usando il libro di testo, gli appunti del docente e quanto reperibile tramite ricerche mirate sul Web.

1. [IIS Alessandrini Marino Teramo](#)
2. [Esami di Stati della Prof.ssa Giselda De Vita](#)
3. [Esami di Stato del Prof. Mauro De Berardis](#)
4. [Soluzione della simulazione del 28 febbraio dell'Ing. Alfredo Centinaro](#)
5. [Soluzione della simulazione del 28 febbraio del Prof. Mauro De Berardis](#)
6. [Soluzione della simulazione del 28 febbraio dell'Ing. Lindo Nepi](#)
7. [Il Wifi gratis sui treni di Trenitalia](#) utile per la prima parte della seconda simulazione della seconda prova dell'Esame di Stato del 2 aprile 2019
8. [Audio Video on demand on board](#) utile per la prima parte della seconda simulazione della seconda prova dell'esame di Stato del 2 aprile 2019
9. [Soluzione della simulazione del 2 aprile del Prof. Mauro De Berardis](#)
10. [Soluzione della simulazione del 2 aprile della Prof.ssa Gisella De Vita](#)

Esercizi sulle prove d'esame

1. [Esempio 2 Esame di Stato del 2016.](#)
 - a. PRIMA PARTE - Il candidato, formulate le opportune ipotesi aggiuntive analizzi la realtà di riferimento, produca un modello grafico che descriva il sistema, ne ponga in evidenza i vari componenti e le loro interconnessioni, motivando le scelte effettuate.
 - b. PRIMA PARTE - Il candidato, formulate le opportune ipotesi aggiuntive descriva, anche utilizzando uno schema grafico, le funzionalità tecnologiche che dovranno possedere i dispositivi a bordo degli automezzi.
 - c. PRIMA PARTE - Il candidato, formulate le opportune ipotesi aggiuntive individui i protocolli di comunicazione da adottare per garantire la sicurezza delle informazioni trasmesse, descrivendone le relative tecnologie.
 - d. SECONDA PARTE - Il candidato descriva le motivazioni che inducono alla realizzazione di una rete intranet in una organizzazione, esplicitando i principali servizi e i relativi protocolli che la rete deve fornire per

soddisfare le esigenze interne. Analizzi il protocollo relativo ad uno di tali servizi.

- e. SECONDA PARTE - Le aziende possono implementare i propri servizi informativi mediante un'infrastruttura interna oppure attraverso sistemi cloud. Si descrivano le caratteristiche delle due soluzioni e se ne analizzino i rispettivi punti di forza e di debolezza.

2. [Esame di Stato del 2018 Sessione Suppletiva.](#)

- a. PRIMA PARTE - Il candidato, formulate le opportune ipotesi aggiuntive proponga un progetto, anche grafico, dell'architettura dell'infrastruttura di rete necessaria a rispondere alle esigenze sopra descritte dettagliando:
- i. le risorse hardware e software necessarie, indicandone, ove utile, i criteri di dimensionamento;
 - ii. un opportuno piano di indirizzamento;
 - iii. le caratteristiche del collegamento ad Internet;
 - iv. le soluzioni possibili per assicurare la continuità del servizio.
- b. PRIMA PARTE - Il candidato, formulate le opportune ipotesi aggiuntive individui e descriva possibili tecniche per proteggere ciascuna start-up da accessi anche locali non autorizzati da parte di personale appartenente alle altre start-up, e per proteggere i server nel locale tecnico da attacchi esterni ed interni.
- c. PRIMA PARTE - Il candidato, formulate le opportune ipotesi aggiuntive proponga i principali servizi di rete necessari (tra cui ad es. identificazione degli utenti, assegnazione della configurazione di rete ai vari client, risoluzione dei nomi, ...), esemplificando le relative configurazioni per uno di essi a sua scelta.
- d. PRIMA PARTE - Il candidato, formulate le opportune ipotesi aggiuntive proponga due possibili soluzioni per consentire alle start-up la gestione dei propri servizi mediante accesso remoto ai server.
- e. SECONDA PARTE - Nelle reti locali è a volte necessario mantenere traccia degli accessi ai siti web visitati dagli utenti ed ottimizzare l'uso della banda di collegamento ad Internet. Descrivere le possibili soluzioni e le implicazioni dal punto di vista della privacy.
- f. SECONDA PARTE - In relazione al tema proposto nella prima parte, il candidato discuta vantaggi e svantaggi dell'adozione di eventuali macchine virtuali sui sistemi server nel locale tecnico (primo capannone) per implementare i servizi delle start-up, motivando le scelte effettuate.
- g. SECONDA PARTE - In molte transazioni in rete è di vitale importanza la riservatezza delle comunicazioni. Si descrivano i principali algoritmi e protocolli per la cifratura simmetrica (o a chiave privata), indicandone le caratteristiche ed alcune applicazioni nell'ambito delle reti di calcolatori.

3. Esame di Stato del 2018 Sessione Ordinaria.

- a. PRIMA PARTE - Il candidato, fatte le opportune ipotesi aggiuntive ipotizzi come potrà essere organizzata operativamente la nuova procedura di gestione informatizzata dei pacchi (acquisizione dei dati di mittente e destinatario, presa in carico dal mittente, metodi di identificazione e procedure operative di tracciamento in ciascuna SO e CSR fino alla consegna, tipologia delle informazioni raccolte, rilevamento dell'avvenuta consegna).
- b. PRIMA PARTE - Il candidato, fatte le opportune ipotesi aggiuntive illustri il progetto dell'infrastruttura informatica necessaria per realizzare la gestione automatizzata dei pacchi e consentirne la tracciabilità, dettagliando:
 - i. dispositivi utilizzati da trasportatori e magazzinieri per lo svolgimento delle proprie attività;
 - ii. modalità di comunicazione tra i sistemi;
 - iii. organizzazione dei server di raccolta dati ed offerta dei servizi informativi; si sviluppino e discutano due o più ipotesi alternative, di cui una totalmente interna all'azienda ed una che contempi anche il ricorso a servizi Cloud, scegliendone una motivatamente.
- c. PRIMA PARTE - Approfondisca gli aspetti legati alla sicurezza delle strumentazioni, dei dati gestiti e del servizio offerto nel caso in esame, e discuta le misure che ritiene utili per garantire la continuità del servizio (aspetti di *business continuity* e *fault tolerance*).
- d. SECONDA PARTE - In relazione al tema proposto nella prima parte, la società FastDelivery è interessata anche a poter monitorare gli spostamenti dei propri automezzi sulla strada in tempo reale. Il candidato illustri quali potrebbero essere le soluzioni tecnologiche disponibili e le modalità e i protocolli utilizzati nella comunicazione tra automezzi e centrale operativa.
- e. SECONDA PARTE - Le sfide poste dalla necessità di assicurare in qualsiasi momento l'accessibilità dei dati agli utenti autorizzati hanno portato allo sviluppo di metodologie di gestione note come clusterizzazione delle risorse hardware e virtualizzazione delle risorse software. Il candidato illustri in cosa consistono queste metodologie ed analizzi vantaggi e svantaggi di ciascuna, anche con esemplificazioni applicative.
- f. SECONDA PARTE - Le comunicazioni via email spesso necessitano dell'applicazione di specifiche precauzioni per la sicurezza. Si descrivano le possibili minacce alle comunicazioni via email e i principali protocolli e servizi per garantire la loro sicurezza.

4. Esame di Stato del 2016 Sessione Ordinaria.

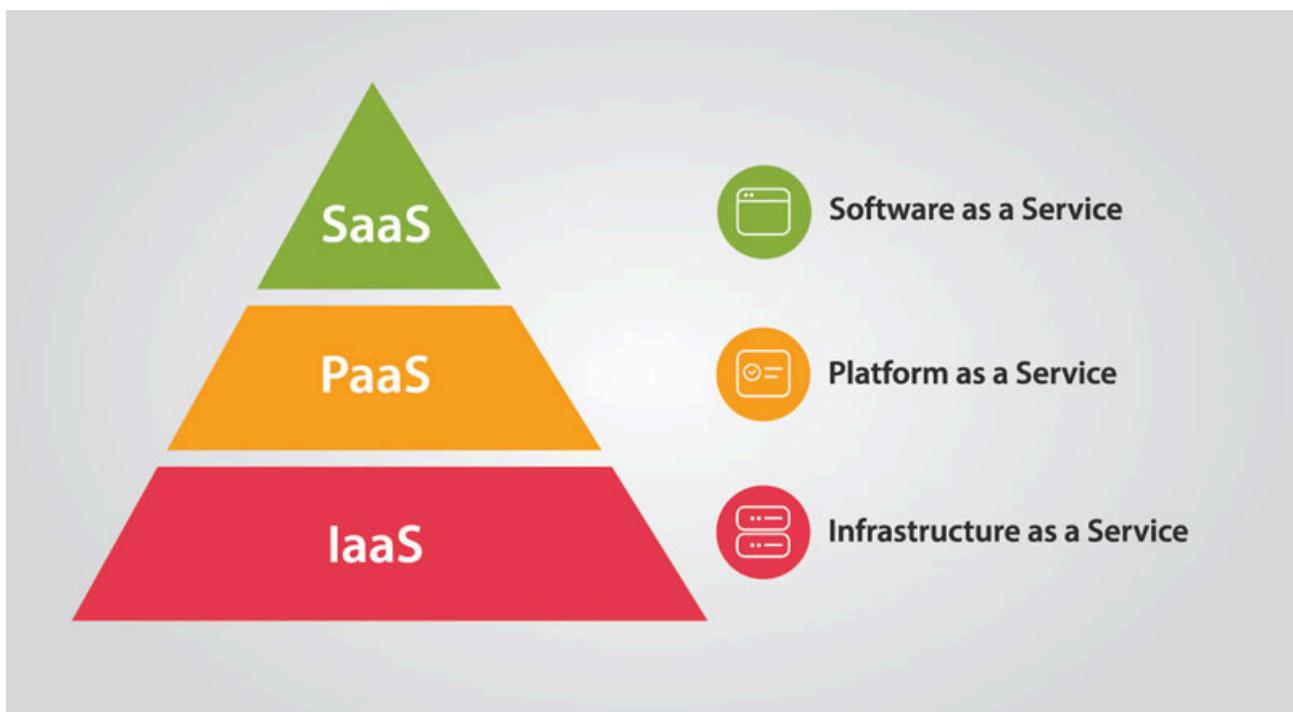
- a. PRIMA PARTE - Il candidato, formulate le opportune ipotesi aggiuntive rappresenti graficamente uno schema logico dell'infrastruttura di rete esistente.
- b. PRIMA PARTE - Il candidato, formulate le opportune ipotesi aggiuntive proponga un progetto anche grafico per l'evoluzione di tale infrastruttura, che soddisfi le esigenze sopra esplicitate, indicando le risorse hardware e software necessarie; approfondisca in particolare le caratteristiche della nuova connessione Internet, i meccanismi per mantenere la separazione del traffico tra le due reti interne, la migrazione degli apparati, gli strumenti di sicurezza, la gestione della linea ADSL di riserva
- c. PRIMA PARTE - Il candidato, formulate le opportune ipotesi aggiuntive proponga i principali servizi da implementare, esemplificando le relative configurazioni per uno di essi a sua scelta.
- d. PRIMA PARTE - Il candidato, formulate le opportune ipotesi aggiuntive specifichi le misure necessarie a prevenire possibili interruzioni nel servizio della piattaforma multimediale.
- e. SECONDA PARTE - In relazione al tema proposto nella prima parte, la scuola intende sviluppare per le classi quinte una didattica basata sul principio del BYOD (Bring Your Own Device), che consiste nell'utilizzo in classe dei dispositivi mobili degli studenti (smartphone, tablet, Pc portatili, ...) per la didattica ordinaria, con accesso ad Internet. Il candidato integri opportunamente il progetto, evidenziando in particolare:
 - i. l'hardware e i servizi necessari all'implementazione di tale infrastruttura;
 - ii. le modalità di limitazione dell'accesso a docenti e studenti delle quinte;
 - iii. le problematiche che si potrebbero presentare e le possibili soluzioni.
- f. SECONDA PARTE - Vista la crescente quantità di informazioni che transitano sulla rete Internet, le tecniche che consentono di garantire la riservatezza delle comunicazioni rivestono sempre maggiore importanza. A tale proposito il candidato esponga le caratteristiche principali della crittografia simmetrica e asimmetrica e le loro modalità di impiego.
- g. SECONDA PARTE - Le società che possiedono più sedi, o che hanno personale che opera in trasferta, necessitano di tecnologie idonee ad uno scambio dati in tempo reale ma al tempo stesso sicuro. Si espongano le possibili soluzioni che rispondono a questo tipo di esigenza, discutendone in dettaglio le caratteristiche a livello di protocolli.

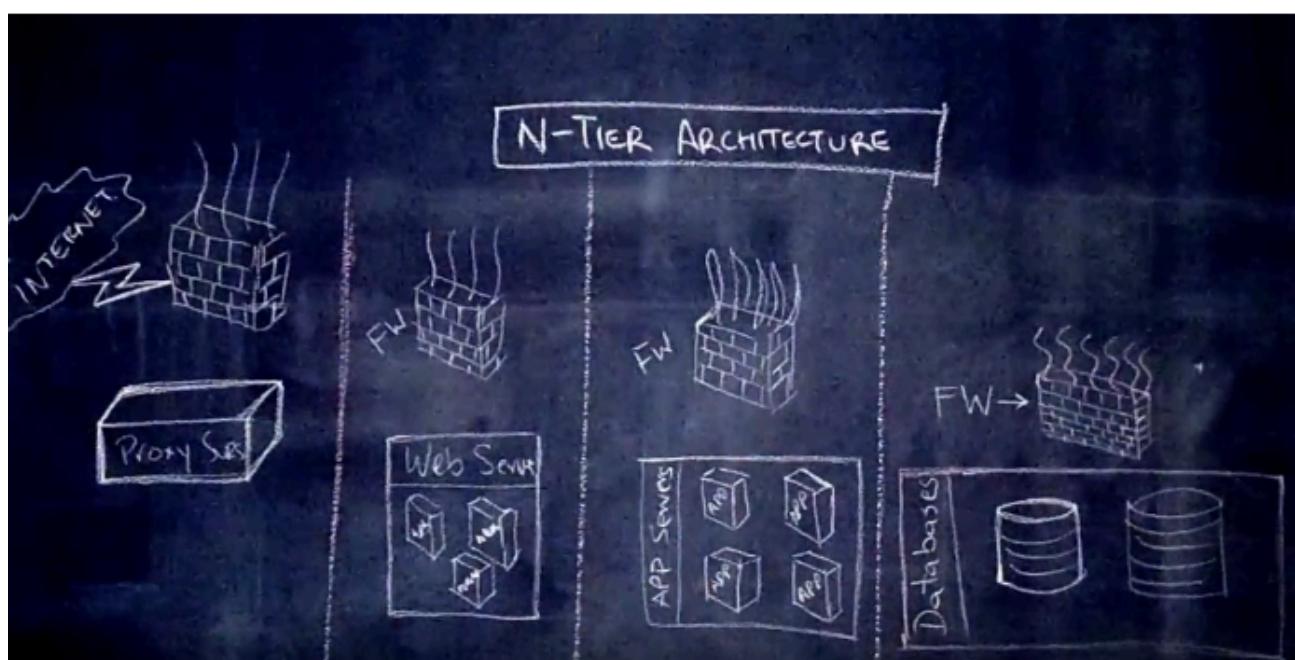
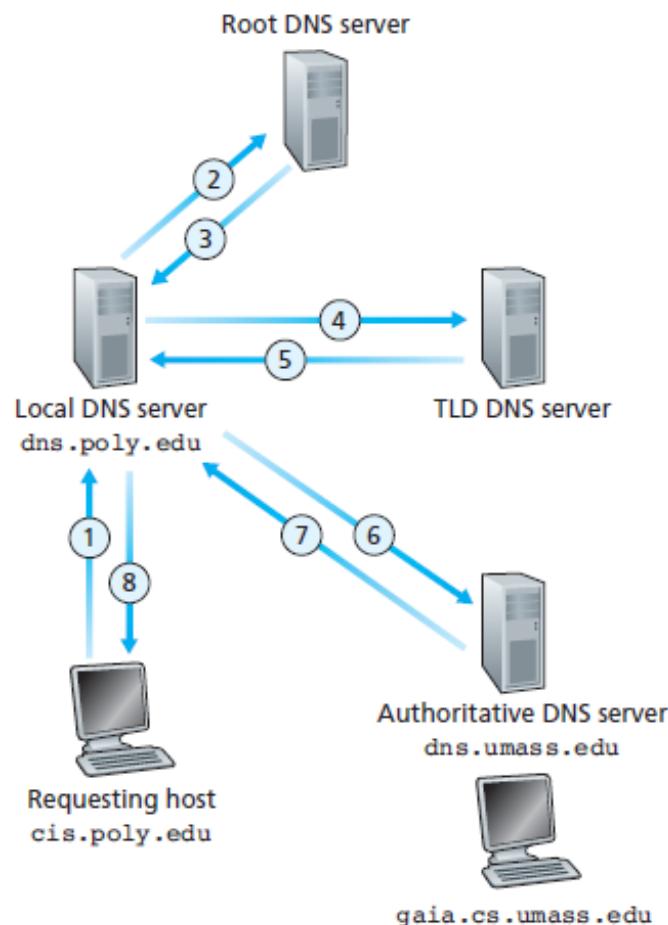
5. [Esame di Stato del 2016 Sessione Suppletiva.](#)

- a. PRIMA PARTE - Il candidato, formulate le opportune ipotesi aggiuntive analizzi la realtà di riferimento e proponga uno schema generale che descriva la soluzione adottata per rispondere alle richieste della compagnia, ne ponga in evidenza i vari componenti e le loro interconnessioni, motivando le scelte effettuate.
- b. PRIMA PARTE - Il candidato, formulate le opportune ipotesi aggiuntive descriva in modo dettagliato le possibili modalità di comunicazione attraverso le quali i clienti richiedono i servizi di trasporto, evidenziando le specificità di ciascuna e la relativa procedura di gestione della richiesta.
- c. PRIMA PARTE - Il candidato, formulate le opportune ipotesi aggiuntive definisca le tecnologie di comunicazione tra la centrale e i mezzi in servizio.
- d. SECONDA PARTE - In relazione al tema proposto nella prima parte, si consideri il caso in cui la compagnia voglia consentire ai clienti di registrarsi al proprio sito per usufruire di campagne promozionali e di servizi aggiuntivi. Il candidato esponga le tecnologie hardware e i servizi software necessari a garantire un adeguato standard di sicurezza a protezione dei dati acquisiti.
- e. SECONDA PARTE - Negli ultimi anni lo sviluppo tecnologico ha portato ad una maggiore apertura delle infrastrutture informatiche, ormai ampiamente interconnesse. La sicurezza dei dati è diventata di conseguenza un aspetto fondamentale nell'ambito del trattamento delle informazioni. Il candidato descriva i possibili tipi di minacce alla sicurezza di un sistema informatico.
- f. SECONDA PARTE - Le informazioni che viaggiano attraverso la rete Internet riguardano, sempre di più, aspetti rilevanti e delicati della vita degli individui e delle aziende. Tale mole di dati necessita di sistemi che garantiscono l'identità dei soggetti, l'integrità dei dati e la loro confidenzialità. Il candidato descriva le caratteristiche dell'infrastruttura di sicurezza basata sulle chiavi pubbliche (PKI) evidenziando il ruolo delle Autorità di Certificazione.

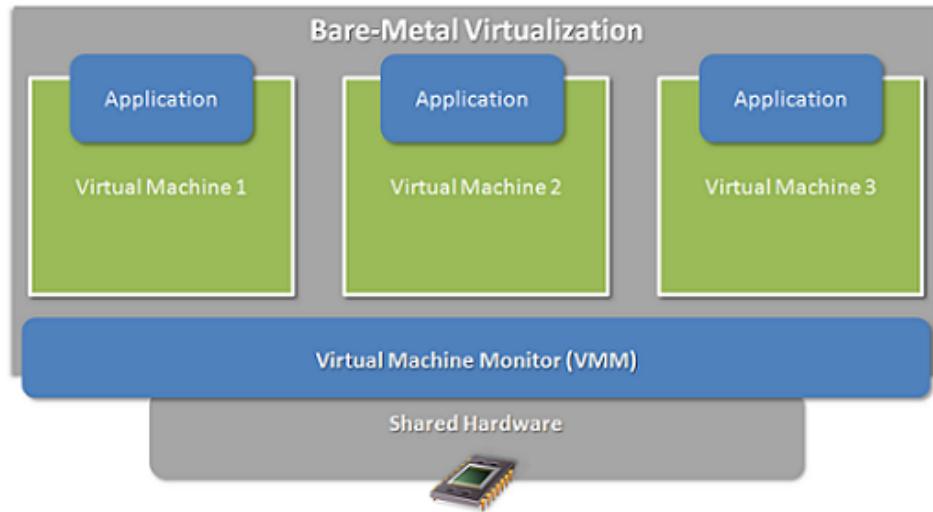
Colloquio

Il candidato analizzi e commenti il materiale fornito, facendo i collegamenti necessari anche con altre discipline.





SERVIZIO



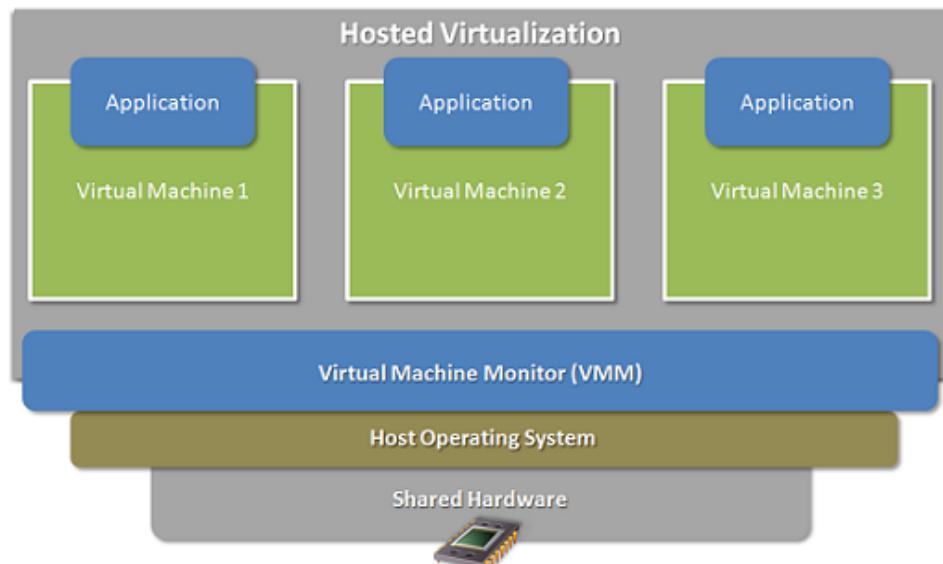
AAA



PKI

SIMMETRICO

ASIMMETRICO



HASH

B - IPv6 Network Addresses - CLIL

Prefazione

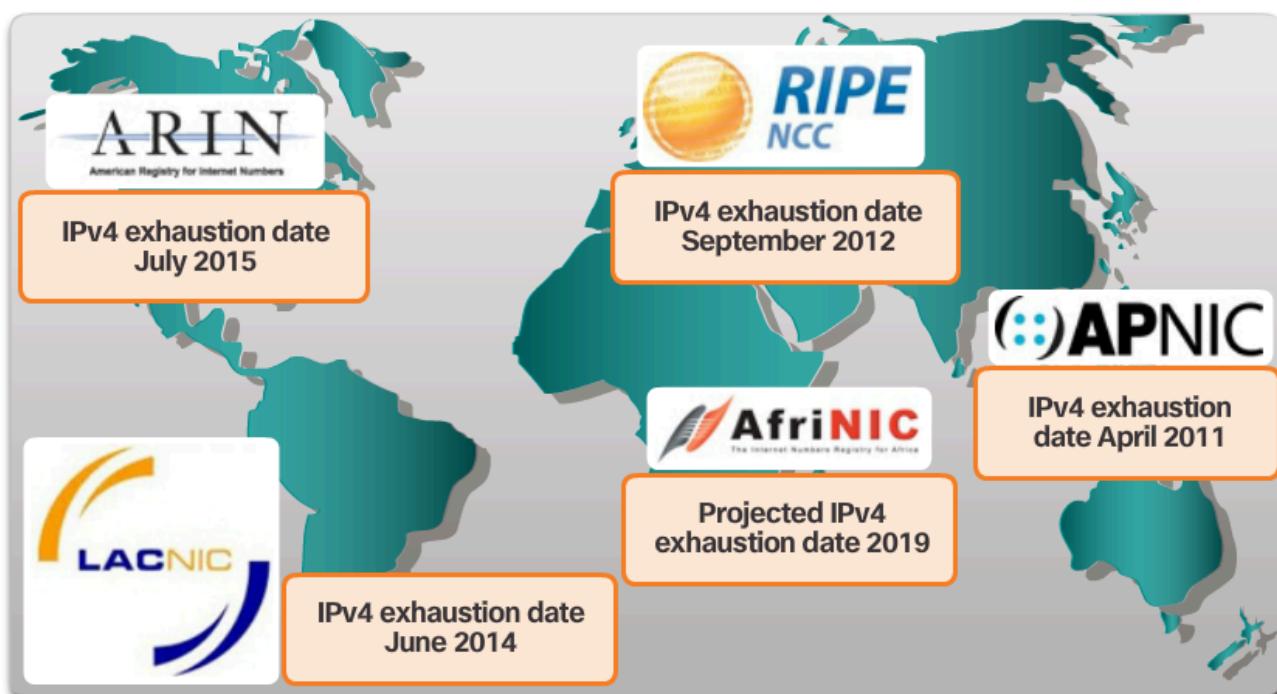
I seguenti appunti sono un estratto del corso Router & Switch della CISCO seguito durante l'a.s. 2016/2017. Dato che l'argomento non era stato trattato in modo esaustivo si riportano qui alcune delle informazioni presenti sul sito della [netacad.com](#).

IPv4 Issues

The Need for IPv6

IPv6 is designed to be the successor to IPv4. **IPv6** has a larger **128-bit address space**. However, IPv6 is more than just larger addresses. When the IETF began its development of a successor to IPv4, it used this opportunity to fix the limitations of IPv4 and include additional enhancements. One example is Internet Control Message Protocol version 6 (ICMPv6), which includes address resolution and address auto-configuration not found in ICMP for IPv4 (ICMPv4). ICMPv4 and ICMPv6 will be discussed later in this chapter.

The *depletion* of IPv4 address space has been the motivating factor for moving to IPv6. As Africa, Asia and other areas of the world become more connected to the Internet, there are not enough IPv4 addresses to accommodate this growth. As shown in the figure, four out of the five RIRs³⁶ have run out of IPv4 addresses.



IPv4 has a theoretical maximum of 4.3 billion addresses. Private addresses in combination with Network Address Translation (NAT) have been instrumental in slowing the depletion of IPv4 address space. However, NAT breaks many applications and has limitations that severely impede peer-to-peer communications.

The Internet of today is significantly different than the Internet of past decades. The Internet of today is more than email, web pages, and file transfer between computers. The evolving Internet is becoming an **Internet of things**. No longer will the only

³⁶ A RIR (Regional Internet Registry) is an organization that manages the allocation and registration of Internet number resources within a particular region of the world. Internet number resources include IP addresses and autonomous system (AS) numbers.

devices accessing the Internet be computers, tablets, and smartphones. The sensor-equipped, Internet-ready devices of tomorrow will include everything from automobiles and biomedical devices, to household appliances and natural ecosystems.

With an increasing Internet population, a limited IPv4 address space, issues with NAT and an Internet of Everything, the time has come to begin the transition to IPv6.

Writing/Speaking Activity - The Need for IPv6

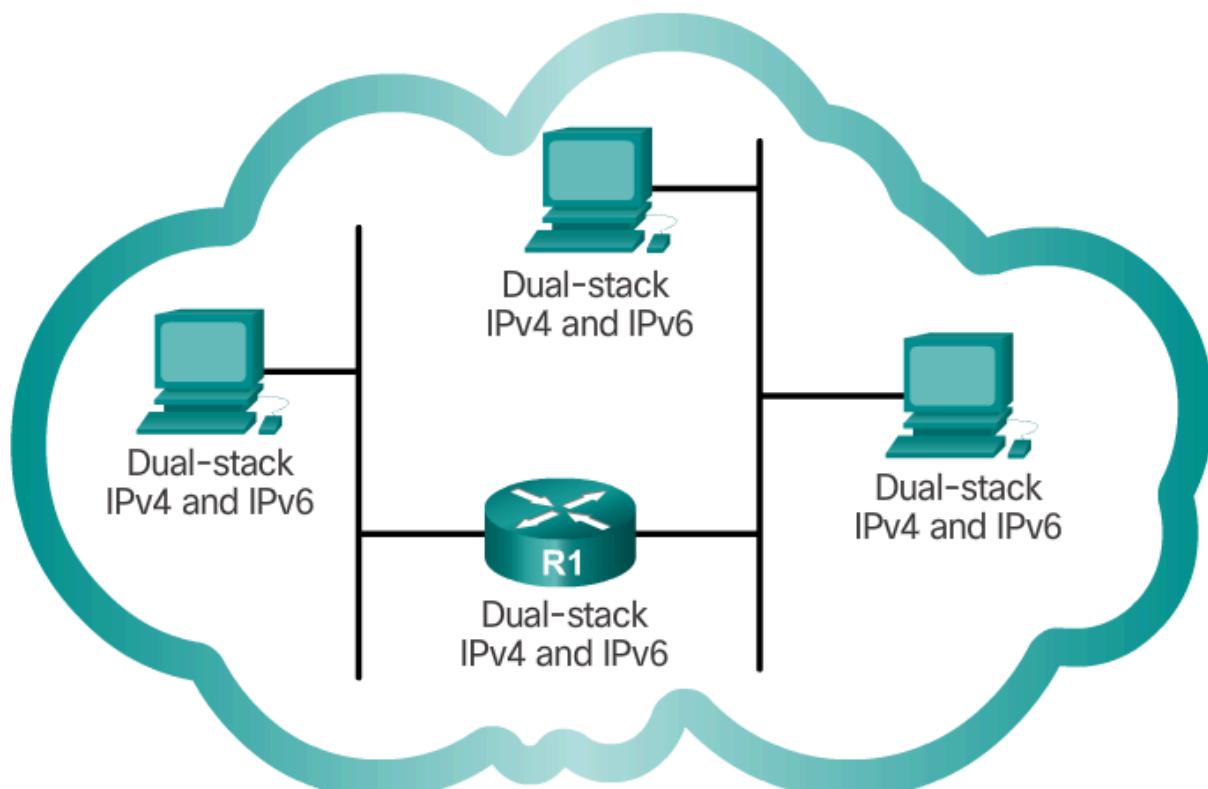
Answer to the following questions. Create a Google document and share it with your teacher and your classmate.

1. How many bits are used for an IPv6 address?
2. Which was the main reason of IPv6 address development?

IPv4 and IPv6 Coexistence

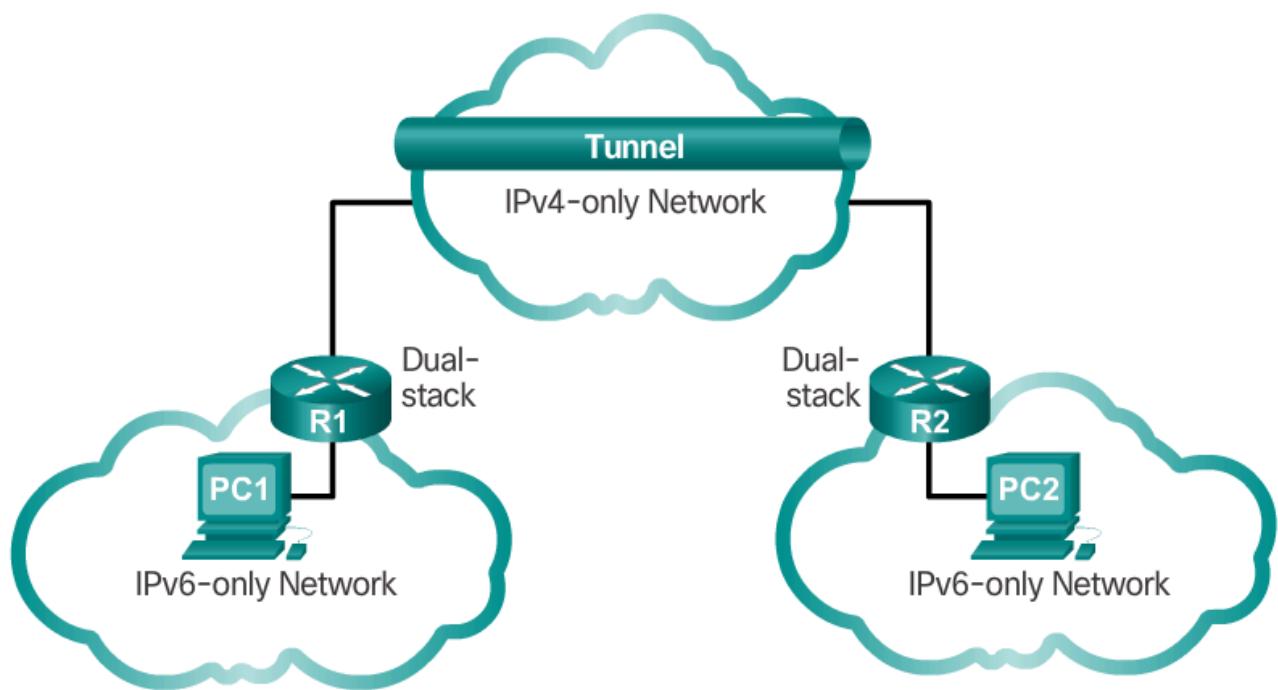
There is not a single date to move to IPv6. For the *foreseeable* future, both IPv4 and IPv6 will coexist. The transition is expected to take years. The IETF has created various protocols and tools to help network administrators migrate their networks to IPv6. The migration techniques can be divided into three categories:

- **Dual Stack** – As shown in the figure, dual stack allows IPv4 and IPv6 to coexist on the same network segment. Dual stack devices run both IPv4 and IPv6 protocol stacks simultaneously.

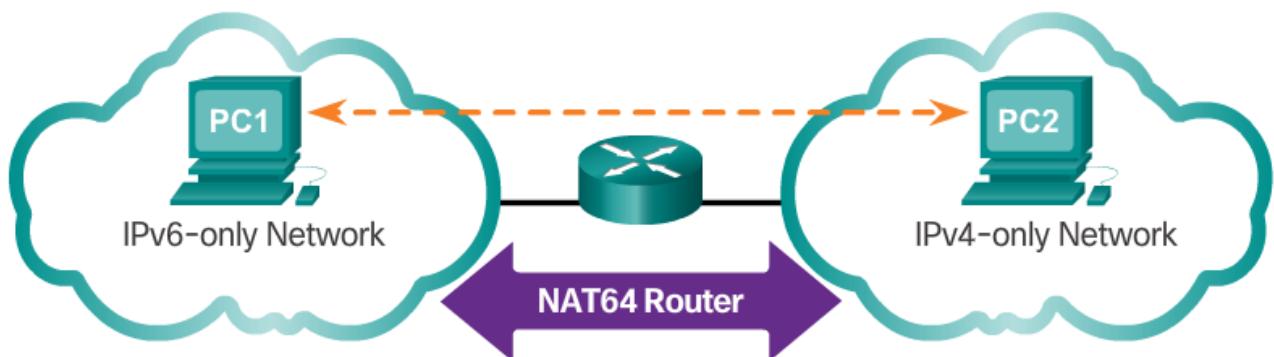


- **Tunneling** – As shown in the figure, tunneling is a method of transporting an IPv6 packet over an IPv4 network. The IPv6 packet is encapsulated inside an

IPv4 packet, similar to other types of data.



- **Translation** – As shown in the figure, Network Address [Translation](#) 64 (NAT64) allows IPv6-enabled devices to communicate with IPv4-enabled devices using a translation technique similar to NAT for IPv4. An IPv6 packet is translated to an IPv4 packet and vice versa.



Note: Tunneling and translation are only used where needed. The goal [should](#)³⁷ be native IPv6 communications from source to destination.

Writing/Speaking Activity - IPv4 and IPv6 Coexistence

Give the following description. Write your answer in the Google document you have already created and shared with your teacher.

3. Describe the three protocols migration [created by](#)³⁸ IETF.

³⁷ Grammar - Modal verbs

³⁸ Grammar - Active / Passive Verb Forms

Listening/Writing Activity - What is IPv6?

Vocabulary

Deployed, exhaustion, feature, stood, ever-growing, huge, depletion, always-on, contactable, furthermore, stage, enhanced, reachability, scalability, hierarchical, carried over, mandatory, revised, neighbour, built-in.

Activity

Watch [this](#) video and write a bullet list containing the main ideas explained by the speaker. Near to each main concepts write a brief explanation.

[This](#) video is also a good explanation of what IPv6 is, but it is challenging because the guy speaks really fast. Anyway it is a good listening exercise.

Activity – IPv4 Issues and Solutions

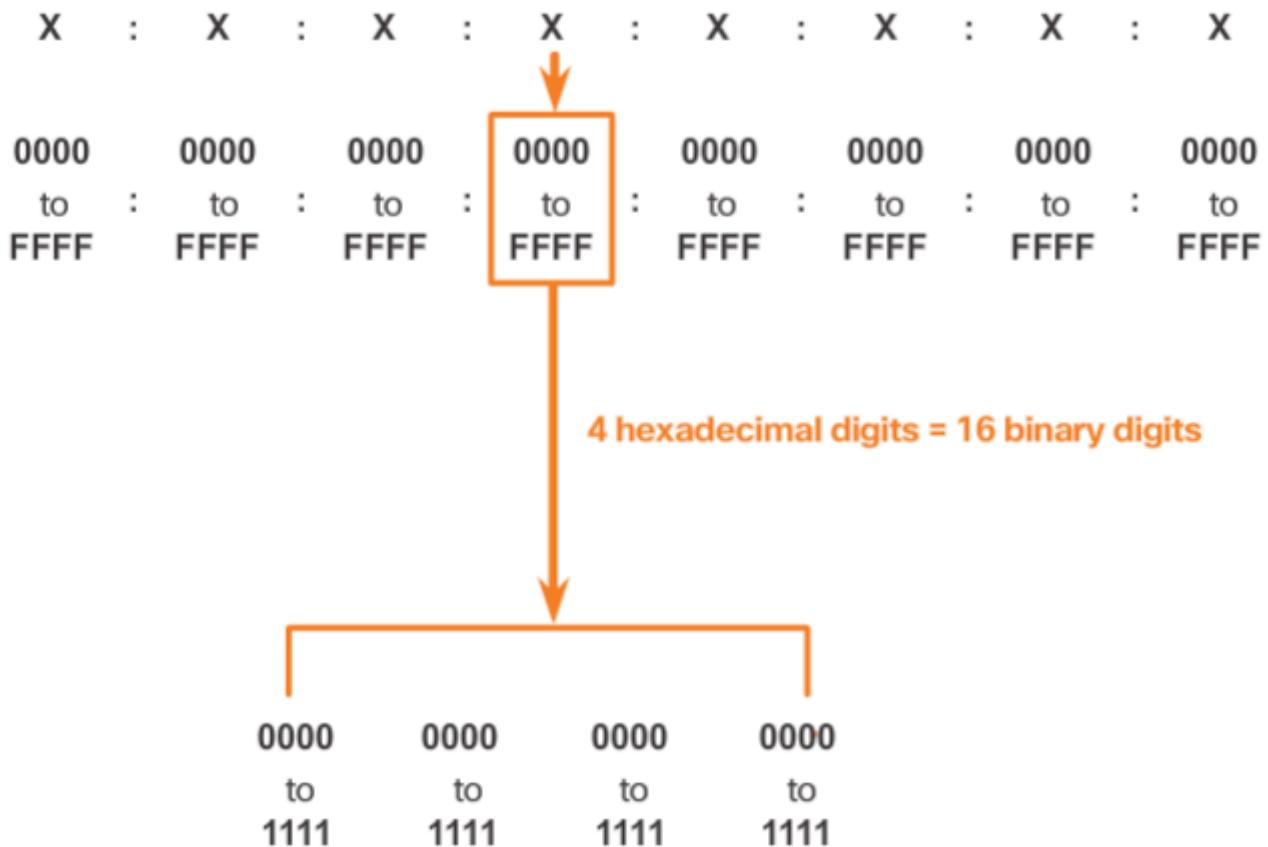
Match the IPv4 and IPv6 terms to the appropriate description.

1) Dual Stack	a) 128-bit address
2) IPv4	b) Uses NAT64 to convert between IPv6 and IPv4
3) Tunneling	c) Transports an IPv6 packet over IPv4 networks
4) IPv6	d) 32-bit address
5) Translation	e) Allows IPv4 and IPv6 to coexist on the same network segment

IPv6 Addressing

IPv6 Address Representation

IPv6 addresses are 128 bits in length and written as a string of [hexadecimal](#) values. Every 4 bits is represented by a single hexadecimal digit; for a total of 32 hexadecimal values, as shown in figure. IPv6 addresses are not case-sensitive and can be written in either lowercase or uppercase.



Preferred format

The preferred format [for writing](#)³⁹ an IPv6 address is x:x:x:x:x:x:x:x, with each “x” consisting of four hexadecimal values. When referring to 8 bits of an IPv4 address we use the term octet. In IPv6, a **hextet is the unofficial term used to refer to a segment of 16 bits** or four hexadecimal values. Each “x” is a single hextet, 16 bits or four hexadecimal digits.

Preferred format means the IPv6 address is written using all 32 hexadecimal digits. It does not necessarily mean it is the ideal method for representing the IPv6 address. In the following pages, we will see two rules to help reduce the number of digits needed to represent an IPv6 address.

³⁹ Grammar - The Use of Gerunds after Prepositions

In the following figure there is a review of the relationship between decimal, binary and hexadecimal.

Decimal and Binary equivalents of 0 to F Hexadecimal

Decimal	Binary	Hexadecimal
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F

In the following figure there are examples of IPv6 addresses in the preferred format.

2001	:	0DB8	:	0000	:	1111	:	0000	:	0000	:	0000	:	0200
2001	:	0DB8	:	0000	:	00A3	:	ABCD	:	0000	:	0000	:	1234
2001	:	0DB8	:	000A	:	0001	:	0000	:	0000	:	0000	:	0100
2001	:	0DB8	:	AAAA	:	0001	:	0000	:	0000	:	0000	:	0200
FE80	:	0000	:	0000	:	0000	:	0123	:	4567	:	89AB	:	CDEF
FE80	:	0000	:	0000	:	0000	:	0000	:	0000	:	0000	:	0001
FF02	:	0000	:	0000	:	0000	:	0000	:	0000	:	0000	:	0001
FF02	:	0000	:	0000	:	0000	:	0000	:	0001	:	FF00	:	0200
0000	:	0000	:	0000	:	0000	:	0000	:	0000	:	0000	:	0001
0000	:	0000	:	0000	:	0000	:	0000	:	0000	:	0000	:	0000

Omit Leading 0s

The first rule to help reduce the notation of IPv6 addresses is to *omit* any *leading* 0s (zeros) in any 16-bit section or hexet. For example:

- **01AB** can be represented as **1AB**
- **09F0** can be represented as **9F0**
- **0A00** can be represented as **A00**
- **00AB** can be represented as **AB**

This rule **only applies to leading 0s, NOT to trailing 0s**, otherwise the address would be ambiguous. For example, the hexet "ABC" could be either "**0ABC**" or "**ABC0**", **but these do not represent the same value.**

The following table shows several examples of how omitting leading 0s can be used to reduce the size of an IPv6 address. For each example, the preferred format is shown. Notice how omitting the leading 0s in most examples results in a smaller address representation.

Preferred	No leading 0s
2001:0DB8:0000:1111:0000:0000:0000:0200	2001:DB8:0:1111:0:0:0:200
2001:0DB8:0000:A300:ABCD:0000:0000:1234	2001:DB8:0:A300:ABCD:0:1234
2001:0DB8:000A:1000:0000:0000:0000:0100	2001:DB8:A:1000:0:0:0:100
FE80:0000:0000:0000:0123:4567:89AB:CDEF	FE80:0:0:0:123:4567:89AB:CDEF
FF02:0000:0000:0000:0000:0000:0001	FF02:0:0:0:0:0:0:1
FF02:0000:0000:0000:0001:FF00:0200	FF02:0:0:0:0:1:FF00:200
0000:0000:0000:0000:0000:0000:0001	0:0:0:0:0:0:0:1
0000:0000:0000:0000:0000:0000:0000	0:0:0:0:0:0:0:0

Omit All 0 Segment

The second rule to help reduce the notation of IPv6 addresses is that a double colon (:) can replace any single, *contiguous* string of one or more 16-bit segments (hexets) consisting of all 0s.

The double colon (:) can only be used once within an address, otherwise there would be more than one possible resulting address. When used with the omitting leading 0s technique, the notation of IPv6 address can often be greatly reduced. This is commonly known as the compressed format.

Incorrect address:

- 2001:0DB8::ABCD::1234

Possible expansions of ambiguous compressed addresses:

- 2001:0DB8::ABCD:0000:0000:1234
- 2001:0DB8::ABCD:0000:0000:0000:1234
- 2001:0DB8:0000:ABCD::1234
- 2001:0DB8:0000:0000:ABCD::1234

The following table shows several examples of how using the double colon (::) and omitting leading 0s can reduce the size of an IPv6 address.

Preferred	Compressed
2001:0DB8:0000:1111:0000:0000:0000:0200	2001:DB8:0:1111::200
2001:0DB8:0000:0000:ABCD:0000:0000:0100	2001:DB8::ABCD:0:0:100 OR 2001:DB8:0:0:ABCD::100
2001:0DB8:000A:1000:0000:0000:0000:0100	2001:DB8:A:1000::100
FE80:0000:0000:0000:0123:4567:89AB:CDEF	FE80::123:4567:89AB:CDEF
FF02:0000:0000:0000:0000:0000:0000:0001	FF02::1
FF02:0000:0000:0000:0000:0001:FF00:0200	FF02::1:FF00:200
0000:0000:0000:0000:0000:0000:0000:0001	::1
0000:0000:0000:0000:0000:0000:0000:0000	::

Activity – Practicing IPv6 Address Representations

Convert the IPv6 addresses into short (omit the leading zeroes) and compressed forms.

Preferred format:	2001:0000:0DB8:1111:0000:0000:0000:0200
Omit leading zeroes:	
Compressed format:	
Preferred format:	2013:0000:0123:4567:89AB:CDEF:0000:0001
Omit leading zeroes:	
Compressed format:	
Preferred format:	2012:ABCD:EF01:2345:0678:0910:AAAA:BBBB
Omit leading zeroes:	
Compressed format:	
Preferred format:	AB1E:2B00:0000:1234:5678:9101:1112:1113
Omit leading zeroes:	

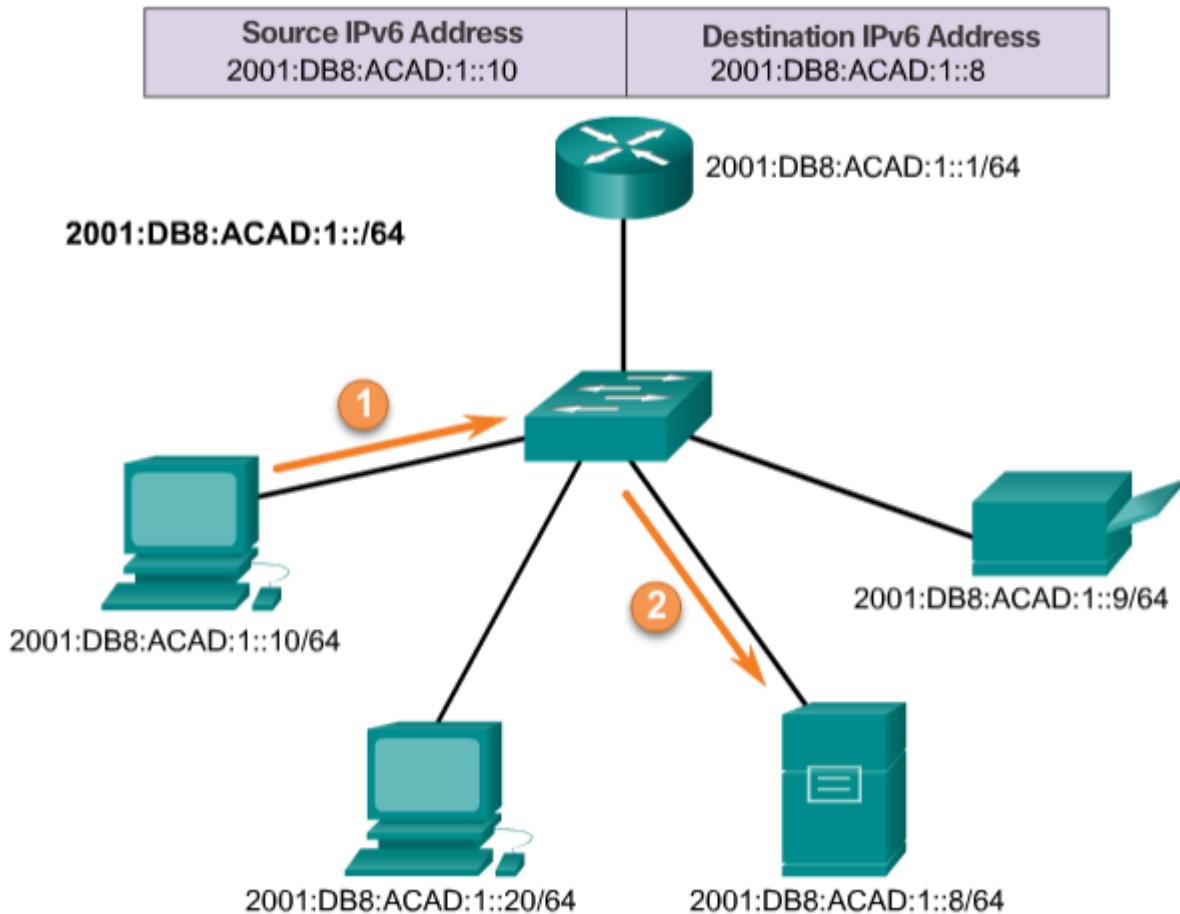
Compressed format:	
Preferred format:	BB2B:EF12:BFF3:9125:1111:0101:1111:0101
Omit leading zeros:	
Compressed format:	
Preferred format:	1129:1984:2233:4455:6677:0000:0000:0101
Omit leading zeros:	
Compressed format:	
Preferred format:	1031:1976:0001:0002:0003:0004:0000:1010
Omit leading zeros:	
Compressed format:	
Preferred format:	0000:0000:0000:1234:6678:9101:0000:34AB
Omit leading zeros:	
Compressed format:	

Types of IPv6 Addresses

IPv6 Address Types

There are three types of IPv6 addresses:

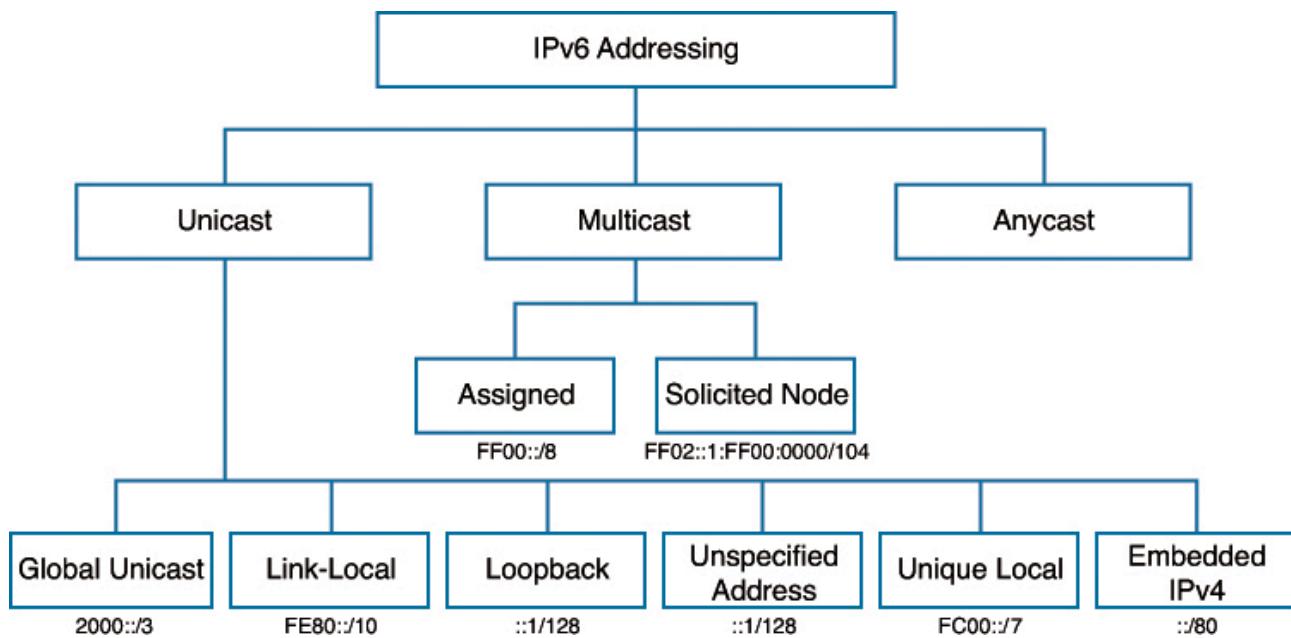
- **Unicast** - An IPv6 unicast address uniquely identifies an interface on an IPv6-enabled device. As shown in the figure, a source IPv6 address must⁴⁰ be a unicast address.



- **Multicast** - An IPv6 multicast address is used to send a single IPv6 packet to multiple destinations.
- **Anycast** - An IPv6 anycast address is any IPv6 unicast address that can be assigned to multiple devices, typically a router not a host. An anycast address must not be used as the source address of an IPv6 packet. A packet sent to an anycast address is routed to the nearest device having that address.

Unlike IPv4, IPv6 does not have a broadcast address. However, there is an IPv6 all-nodes multicast address that essentially gives the same result.

⁴⁰ Grammar - Modals.

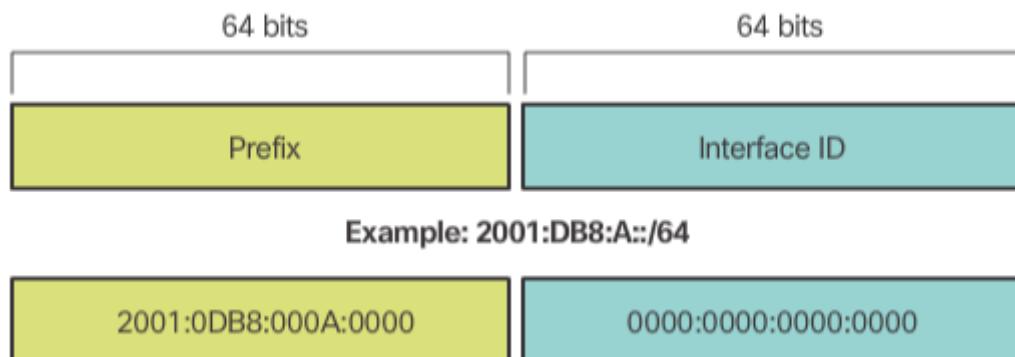


IPv6 Prefix Length

Recall that the prefix, or network portion, of an IPv4 address, can be identified by a dotted-decimal subnet mask or prefix length (slash notation). For example, an IPv4 address of 192.168.1.10 with dotted-decimal subnet mask 255.255.255.0 is equivalent to 192.168.1.10/24.

IPv6 uses the prefix length to represent the prefix portion of the address. IPv6 does not use the dotted-decimal subnet mask notation. The prefix length is used to indicate the network portion of an IPv6 address using the IPv6 address/prefix length.

The prefix length can range from 0 to 128. A typical IPv6 prefix length for LANs and most other types of networks is /64. This means the prefix or network portion of the address is 64 bits in length, leaving another 64 bits for the interface ID (host portion) of the address.



Listening Activity - IPv6 Address Types

Vocabulary

Come across, as well, first up, at (the very) least, prefix.

Activity

Watch [this](#) video and answer to the following questions:

1. Which kind of IPv6 address is assigned to a single interface?
 - a. anycast;
 - b. multicast;
 - c. unicast;
 - d. embedded IPv4.
2. Which one of the following IPv6 unicast addresses are similar to an IPv4 public address?
 - a. link-local;
 - b. global unicast;
 - c. loopback;
 - d. embedded IPv4;
 - e. unique local.
3. Is it possible to forward a packet on Internet using a link local address?
 - a. yes;
 - b. no.
4. Which one of the following IPv6 unicast addresses are similar to an IPv4 private address?
 - a. link-local;
 - b. global unicast;
 - c. loopback;
 - d. embedded IPv4;
 - e. unique local.
5. Sign which of the following statements are true or false about multicast address:
 - T | F - multicast address does not behaves the same way in IPv6 and IPv4;
 - T | F - multicast address identifies a group of interfaces located in different devices;
 - T | F - if you want reach all the member of a group you just send the packet to the multicast IPv6 address;
 - T | F - in IPv6 a device cannot join or leave a group whenever it wants;
 - T | F - IPv6 supports broadcast like IPv4.
6. Sign which of the following statements are true or false about anycast address:
 - T | F - a single interface of a device is represented by multiple anycast addresses;
 - T | F - when a packet is sourced to an anycast address, it is only going to be delivered to one of the host in the anycast group;

- T | F - when a packet is sourced to an anycast address, it is going to be delivered to every host in the anycast group;
- T | F - when a packet is sourced to an anycast address, it is going to be delivered to *farthest* host in the anycast group;

IPv6 Unicast Addresses

An IPv6 **unicast address** uniquely identifies an interface on an IPv6-enabled device. A packet sent to a unicast address is received by the interface that is assigned that address. Similar to IPv4, a source IPv6 address must be a unicast address. The destination IPv6 address can be either a unicast, a multicast or⁴¹ an anycast address.

The most common types of IPv6 unicast addresses are global unicast addresses (GUA) and link-local unicast addresses.

IPv6 Link-Local Unicast Addresses

Link-local addresses are used to communicate with other devices on the same local link and only on that link (subnet). With IPv6, the term **link refers to a subnet**. Link-local addresses are confined to a single link and their uniqueness must only be confirmed on that link because **they are not routable beyond the link**. In other words, routers will not forward packets with a link-local source or destination address.

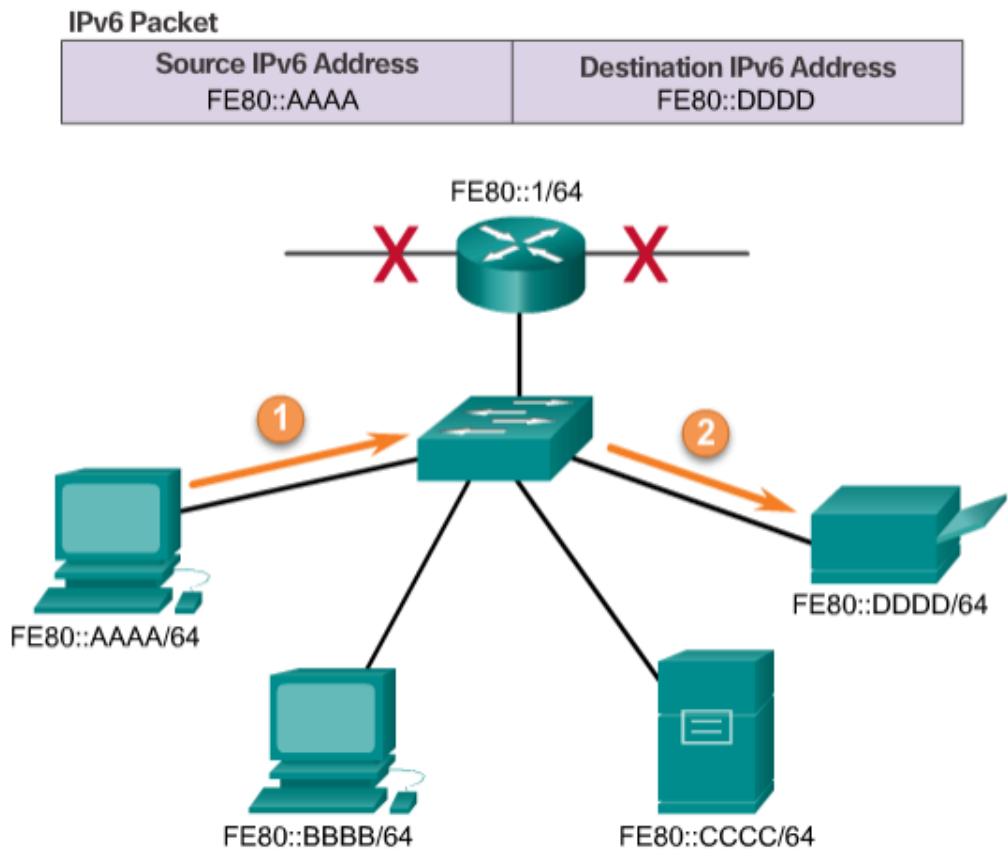
The global unicast address is not a requirement. However, **every IPv6-enabled network interface is required to have a link-local address**.

If a link-local address is not configured manually on an interface, the device will automatically create its own without communicating with a DHCP server. IPv6-enabled hosts create an IPv6 link-local address even if the device has not been assigned a global unicast IPv6 address. This allows IPv6-enabled devices to communicate with other IPv6-enabled devices on the same subnet. This includes communication with the default gateway (router).

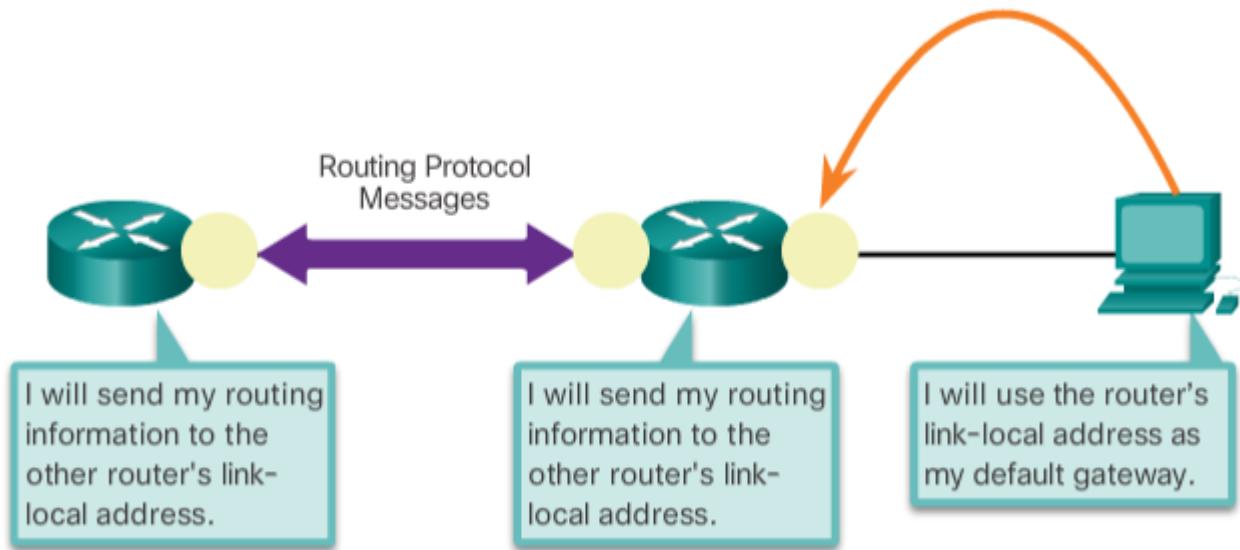
IPv6 link-local addresses are in the FE80::/10 range. The /10 indicates that the first 10 bits are 1111 1110 10**xx xxxx**. The first hextet has a range of 1111 1110 10**00 0000** (FE80) to 1111 1110 10**11 1111** (FEBF).

The following figure shows an example of communication using IPv6 link-local addresses.

⁴¹ Grammar - Either ... or ...



The following figure shows some of the uses for IPv6 link-local addresses.



Note: Typically, it is the link-local address of the router and not the global unicast address, that is used as the default gateway for other devices on the link.

IPv6 Unique local address

Another type of unicast address is the **unique local unicast address**. IPv6 unique local addresses have some similarity to RFC 1918 private addresses for IPv4, but there are significant differences. **Unique local addresses are used for local**

addressing within a site or between a limited number of sites. These addresses should not be routable in the global IPv6 and should not be translated to a global IPv6 address. Unique local addresses are in the range of FC00::/7 to FDFF::/7.

With IPv4, private addresses are combined with NAT/PAT to provide a many-to-one translation of private-to-public addresses. This is done because of the limited availability of IPv4 address space. Many sites also use the private nature of RFC 1918 addresses to help secure or hide their network from potential security risks. However, this was never the intended use of these technologies, and the IETF has always recommended that sites take the proper security precautions on their Internet-facing router. **Unique local addresses can be used for devices that will never need or have access from another network.**

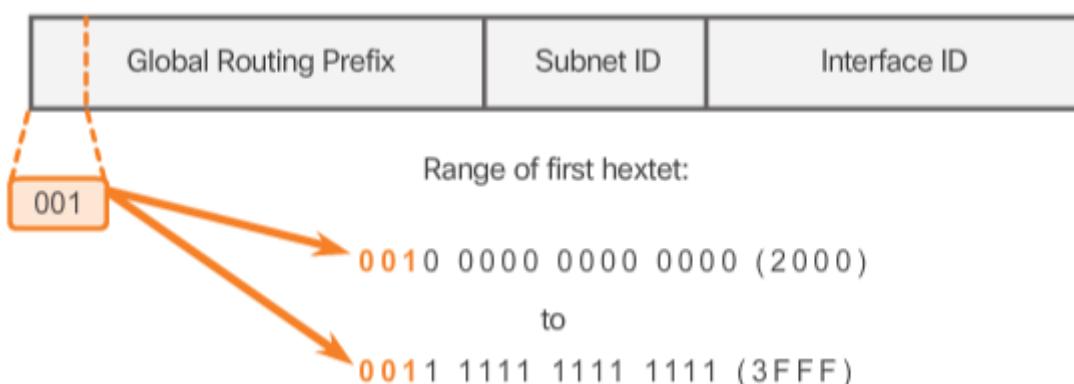
IPv6 Global Unicast Address

IPv6 global unicast addresses are globally unique and routable on the IPv6 Internet. These addresses are **equivalent to public IPv4 addresses.** Global unicast addresses can be configured statically or assigned dynamically.

The Internet Committee for Assigned Names and Numbers (ICANN), the operator for IANA, allocates IPv6 address blocks to the five RIRs⁴². Currently, only global unicast addresses with the first three bits of 001 or 2000::/3 are being assigned. This is only 1/8th of the total available IPv6 address space, excluding only a very small portion for other types of unicast and multicast addresses.

Note: The 2001:0DB8::/32 address has been reserved for documentation purposes, including use in examples.

The figure below shows the structure and range of a global unicast address.



A global unicast address has three parts:

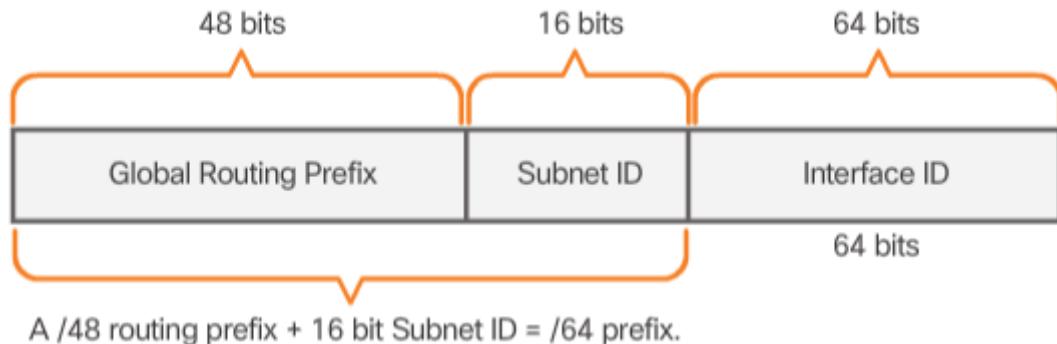
- Global routing prefix
- Subnet ID
- Interface ID

⁴² A RIR (Regional Internet Registry) is an organization that manages the allocation and registration of Internet number resources within a particular region of the world. Internet number resources include IP addresses and autonomous system (AS) numbers.

Global Routing Prefix

The **global routing prefix** is the prefix, or **network**, portion of the address that is assigned by the provider, such as an ISP, to a customer or site. Typically, RIRs assign a /48 global routing prefix to customers. This can include everyone from enterprise business networks to individual households.

The figure below shows the structure of a global unicast address using a /48 global routing prefix. /48 prefixes are the most common global routing prefixes assigned.



For example, the IPv6 address 2001:0DB8:ACAD::/48 has a prefix that indicates that the first 48 bits (3 hextets) (2001:0DB8:ACAD) is the prefix or network portion of the address. The double colon (::) prior to the /48 prefix length means the rest of the address contains all 0s.

The size of the global routing prefix determines the size of the subnet ID.

Subnet ID

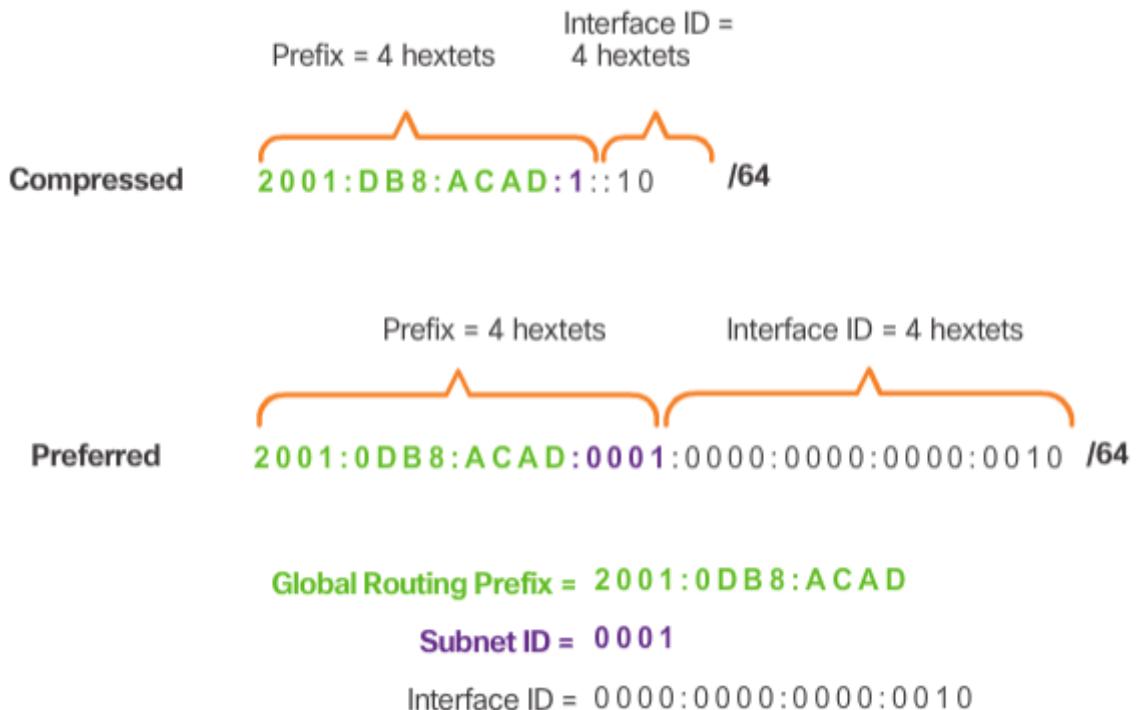
The **Subnet ID** is used by an organization to **identify subnets** within its site. The larger the subnet ID, the more subnets available.

Interface ID

The **IPv6 Interface ID is equivalent to the host portion of an IPv4 address**. The term Interface ID is used because a single host may have multiple interfaces, each having one or more IPv6 addresses. It is highly recommended that in most cases /64 subnets should be used. In other words a 64-bit interface ID as shown in the previous figure.

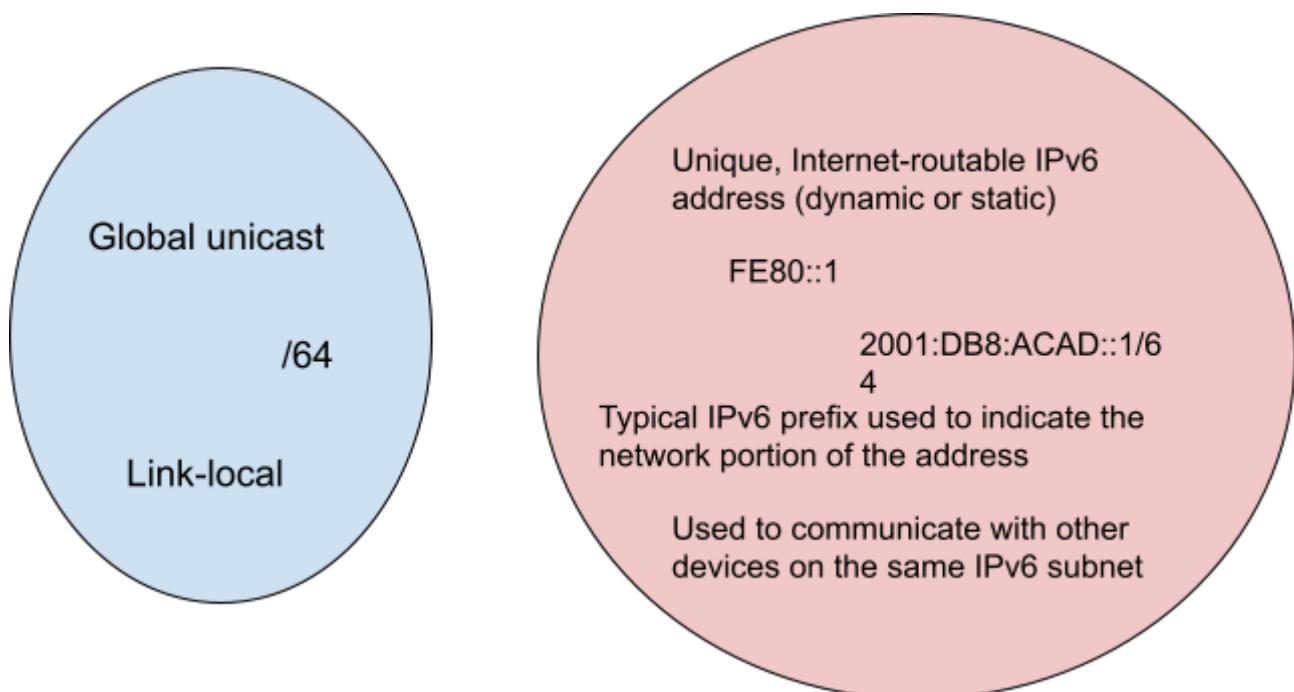
Note: Unlike IPv4, in IPv6, the **all-0s and all-1s host addresses can be assigned to a device**. The all-1s address can be used due to the fact that broadcast addresses are not used within IPv6. The all-0s address can also be used, but is reserved as a Subnet-Router anycast address, and should be assigned only to routers.

An easy way to read most IPv6 addresses is to count the number of hextets. As shown in the following figure, in a /64 global unicast address the first four hextets are for the network portion of the address, with the fourth hextet indicating the Subnet ID. The remaining four hextets are for the Interface ID.



Activity – Identify Types of IPv6 Addresses

Connect the IPv6 address type to the most appropriate description. Create a complete sentence using the IPv6 address type and all associated descriptions.



Static Configuration of a Global Unicast Address

Router Configuration

Most IPv6 configuration and verification commands in the Cisco IOS are similar to their IPv4 counterparts. In many cases, the only difference is the use of `ipv6` in place of `ip` within the commands.

The command to configure an IPv6 global unicast address on an interface is

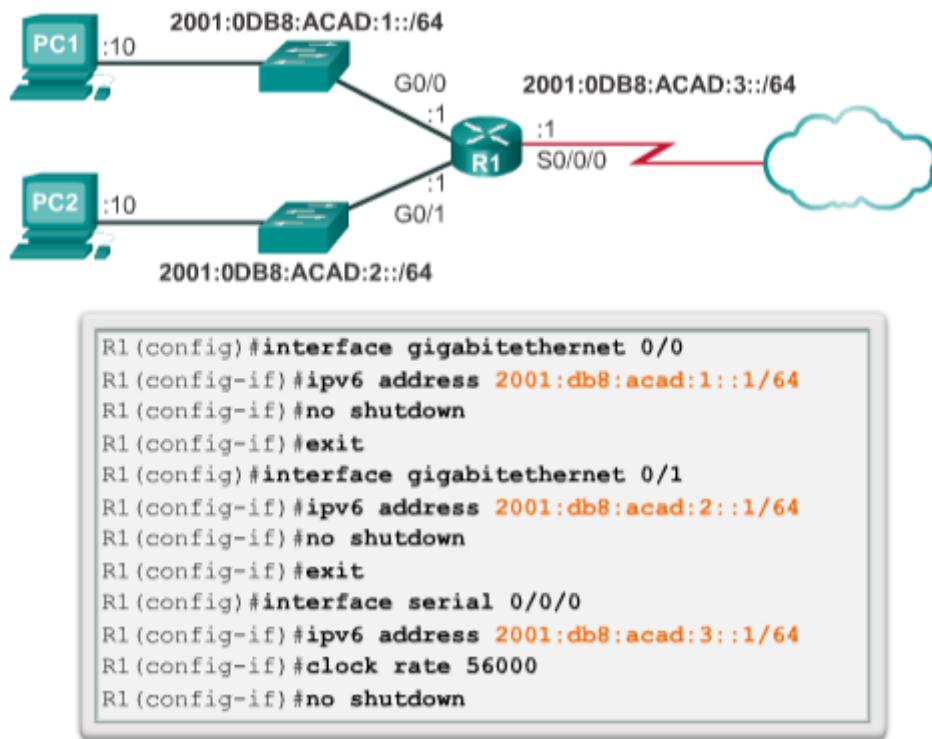
```
#ipv6 address ipv6-address/prefix-length
```

Notice that there is not a space between `ipv6-address` and `prefix-length`.

The example configuration shown in the figure below uses the topology in the picture and these IPv6 subnets:

- 2001:0DB8:ACAD:0001:/64 (or 2001:DB8:ACAD:1::/64)
- 2001:0DB8:ACAD:0002:/64 (or 2001:DB8:ACAD:2::/64)
- 2001:0DB8:ACAD:0003:/64 (or 2001:DB8:ACAD:3::/64)

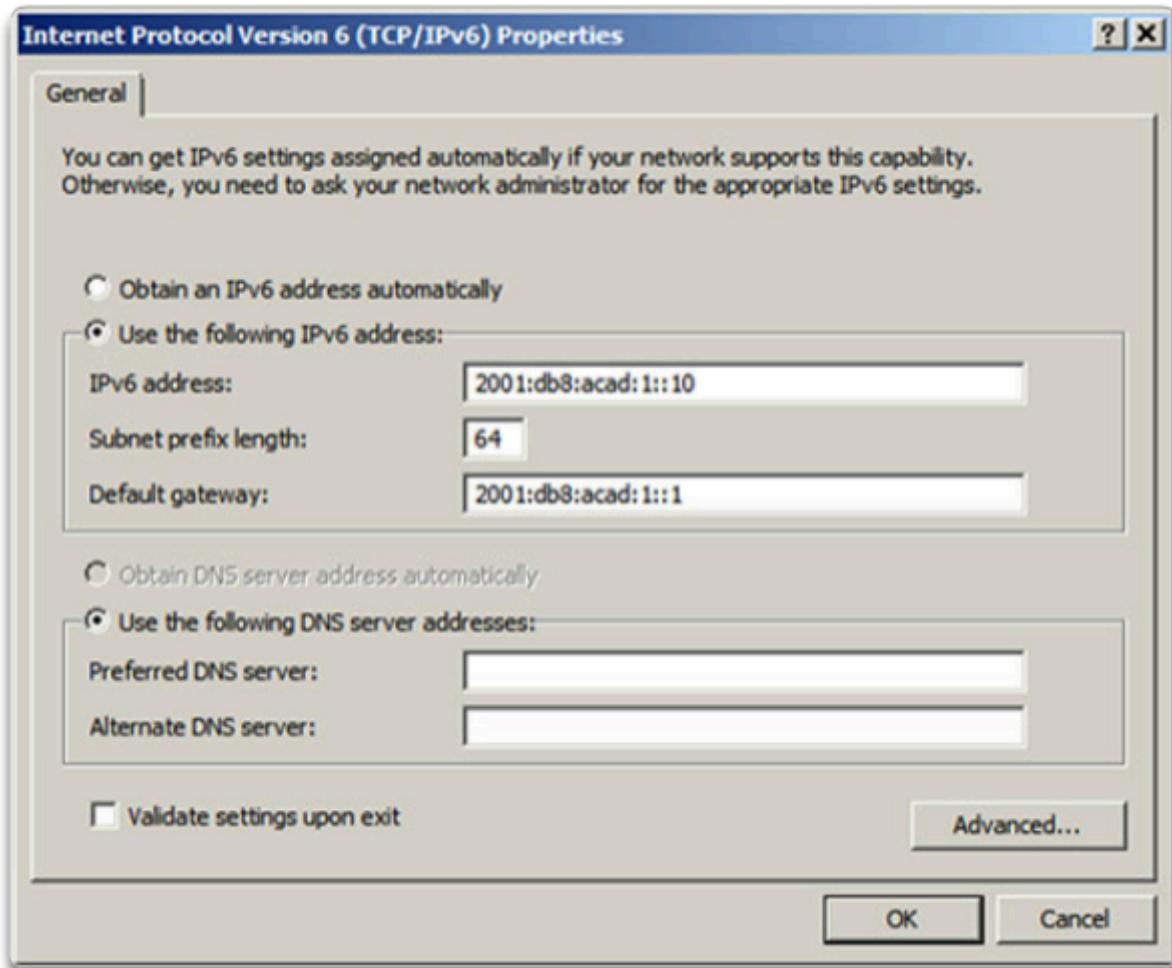
The figure also shows the commands required to configure the IPv6 global unicast address on the GigabitEthernet 0/0, GigabitEthernet 0/1, and Serial 0/0/0 interface of R1.



Host Configuration

Manually configuring the IPv6 address on a host is similar to configuring an IPv4 address.

As shown the figure below, the default gateway address configured for PC1 is 2001:DB8:ACAD:1::1. This is the global unicast address of the R1 GigabitEthernet interface on the same network. Alternatively, the default gateway address can be configured to match the link-local address of the GigabitEthernet interface. Either configuration will work.



Dynamic Configuration

Just as with IPv4, configuring static addresses on clients does not scale to larger environments. For this reason, most network administrators in an IPv6 network will enable dynamic assignment of IPv6 addresses.

There are two ways in which a device can obtain an IPv6 global unicast address automatically:

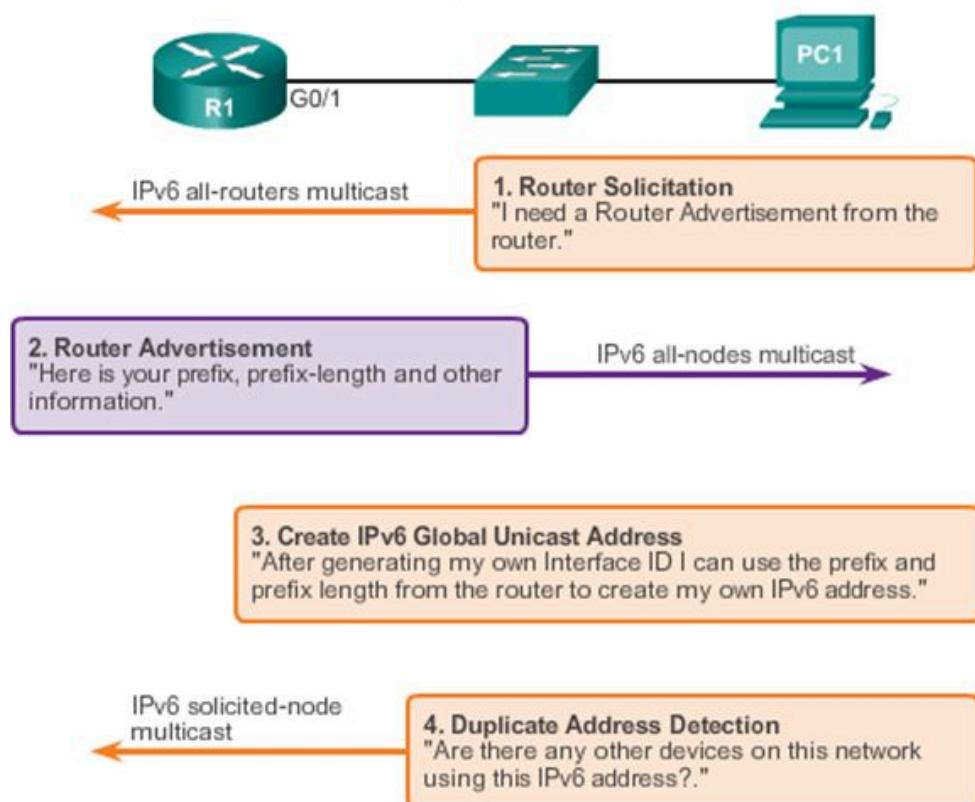
- Stateless Address Autoconfiguration (**SLAAC**)
- **DHCPv6**

Note: When DHCPv6 or SLAAC is used, the local router's link-local address will automatically be specified as the default gateway address.

Dynamic Configuration - SLAAC

Stateless Address Autoconfiguration (SLAAC) is a method that allows a device to obtain its prefix, prefix length, default gateway address, and other information from an IPv6 router without the use of a DHCPv6 server. Using SLAAC, devices rely on the local router's **ICMPv6 Router Advertisement (RA)** messages to obtain the necessary information.

IPv6 routers periodically send out ICMPv6 RA messages, every 200 seconds, to all IPv6-enabled devices on the network. An RA message will also be sent in response to a host sending an **ICMPv6 Router Solicitation (RS)** message.



The last phase is optional, a host uses this phase to verify if the generated IPv6 address is not already in use, e.i. for a statically address assignments.

IPv6 routing is not enabled by default. To enable a router as an IPv6 router, the `ipv6 unicast-routing` global configuration command must be used.

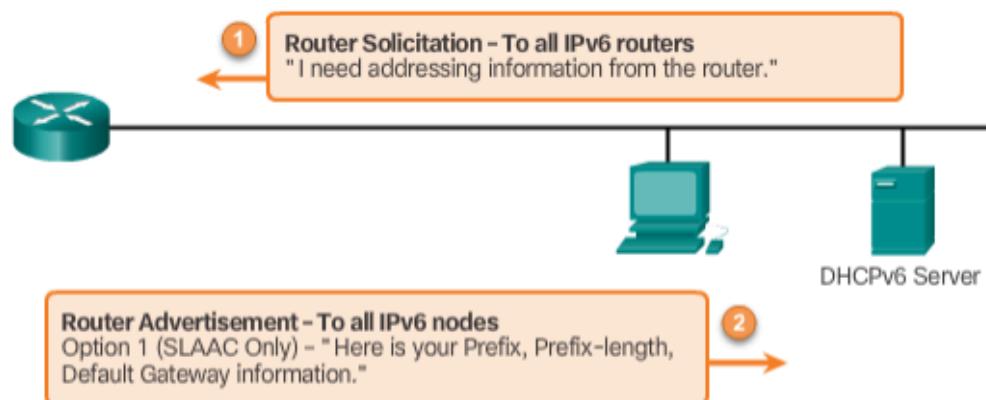
Note: IPv6 addresses can be configured on a router without it being an IPv6 router.

The ICMPv6 RA (Router Advertisement) message is a suggestion to a device on how to obtain an IPv6 global unicast address. The ultimate decision is up to the device's operating system. The ICMPv6 RA message includes:

- **Network prefix and prefix length** – Tells the device which network it belongs to.
- **Default gateway address** – This is an IPv6 link-local address, the source IPv6 address of the RA message.
- **DNS addresses and domain name** – Addresses of DNS servers and a domain name.

As shown in the figure below, there are three options for RA messages:

- Option 1: SLAAC (the only one it will be seen)
- Option 2: SLAAC with a stateless DHCPv6 server
- Option 3: Stateful DHCPv6 (no SLAAC)



Router Advertisement Options

Option 1 (SLAAC Only) – "I'm everything you need (Prefix, Prefix-length, Default Gateway)"

Option 2 (SLAAC and DHCPv6) – "Here is my information but you need to get other information such as DNS addresses from a DHCPv6 server."

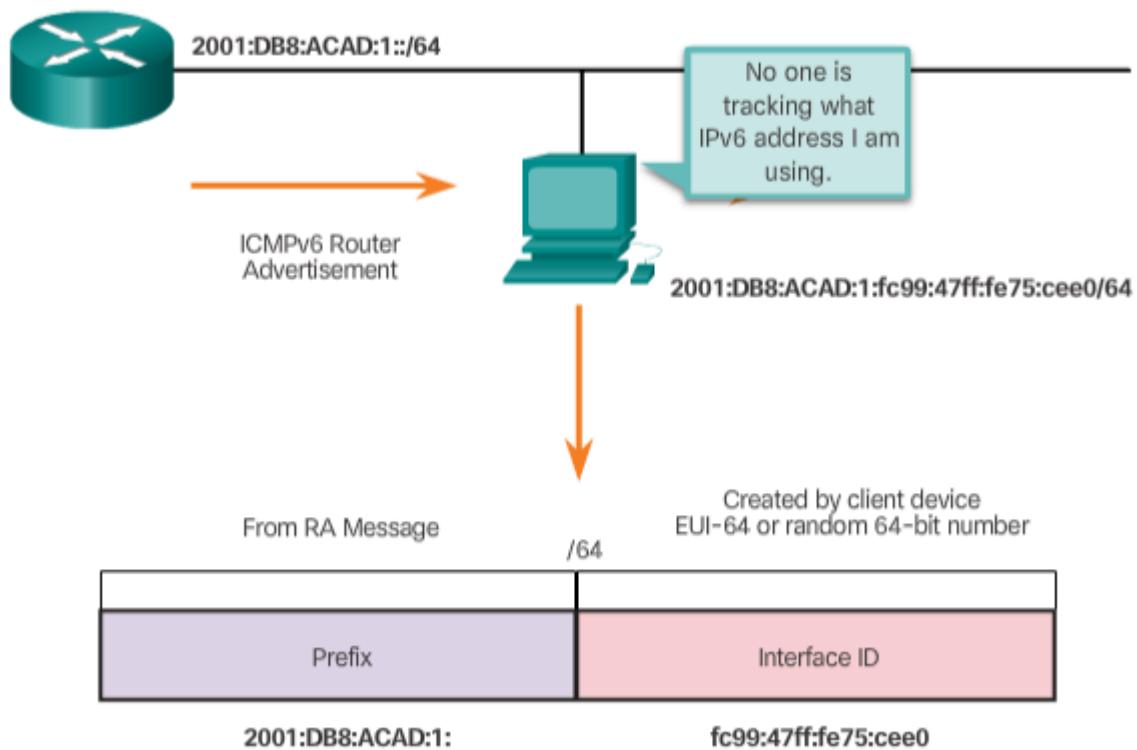
Option 3 (DHCPv6 Only) – "I can't help you. Ask a DHCPv6 server for all your information."

RA Option 1: SLAAC

By default, the RA (Router Advertisement) message suggests that the receiving device use the information in the RA message to create its own IPv6 global unicast address and for all other information. The services of a DHCPv6 server are not required.

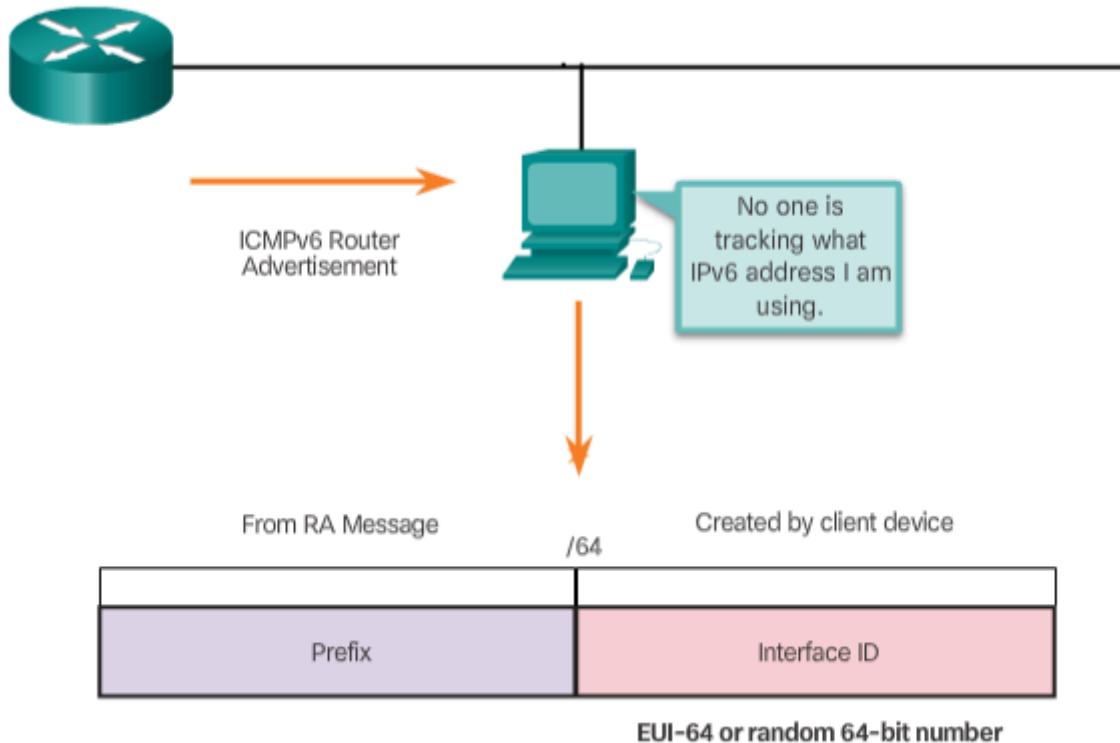
SLAAC is stateless, which means there is no central server (for example, a stateful DHCPv6 server) allocating global unicast addresses and keeping a list of devices and their addresses. With SLAAC, the client device uses the information in the RA message to create its own global unicast address. As shown in the figure below, the two parts of the address are created as follows:

- **Prefix** – Received in the RA message
- **Interface ID** – Uses the EUI-64 process or by generating a random 64-bit number



EUI-64 Process and Randomly Generated

When the RA message is either SLAAC or SLAAC with stateless DHCPv6, the client must generate its own Interface ID. The client knows the prefix portion of the address from the RA message but must create its own Interface ID. The Interface ID can be created using the EUI-64 process or a randomly generated 64-bit number, as shown in following figure.



The automatic EUI-64 address generation can be obtained using the following command:

```
hostname(config-if)#ipv6 address autoconfig
```

EUI-64 Process to create IPv6 address

IEEE defined the **Extended Unique Identifier** (EUI) or modified EUI-64 process. This process uses a client's 48-bit Ethernet MAC address, and inserts another 16 bits in the middle of the 48-bit MAC address to create a 64-bit Interface ID.

Ethernet MAC addresses are usually represented in hexadecimal and are made up of two parts:

- **Organizationally Unique Identifier (OUI)** – The OUI is a 24-bit (6 hexadecimal digits) vendor code assigned by IEEE.
- **Device Identifier** – The device identifier is a unique 24-bit (6 hexadecimal digits) value within a common OUI.

An **EUI-64 Interface ID** is represented in binary and is made up of three parts:

- **24-bit OUI from the client MAC address**, but the **7th bit** (the Universally/Locally (U/L) bit) **is reversed**. This means that if the 7th bit is a 0, it becomes a 1, and vice versa.
- The inserted **16-bit value FFFE** (in hexadecimal)
- **24-bit Device Identifier** from the client MAC address

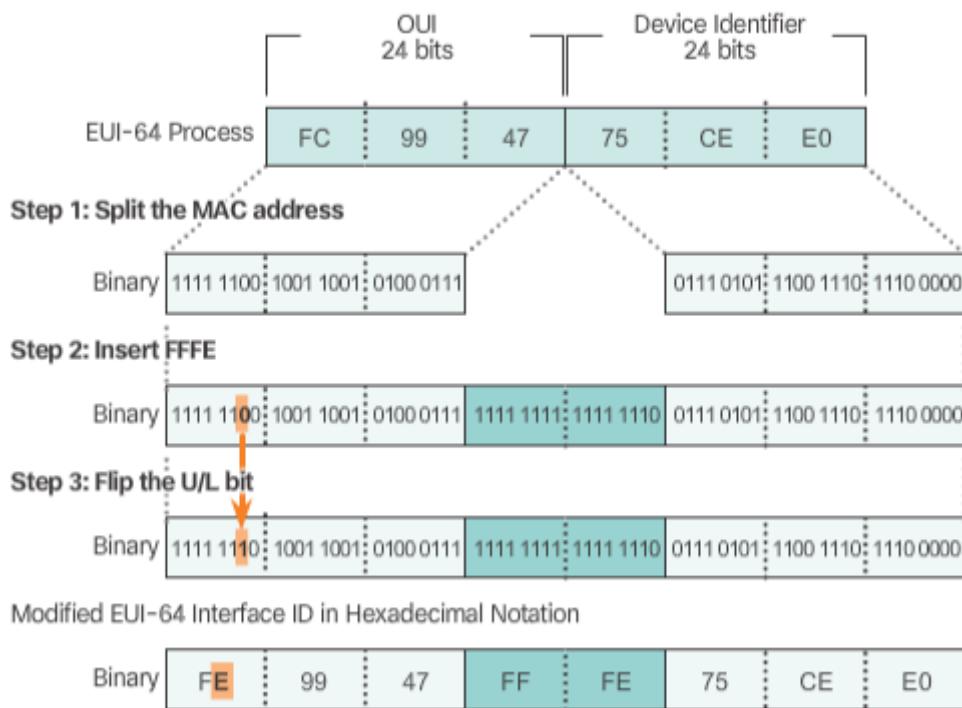
The EUI-64 process is illustrated in the figure below, using the MAC address FC99:4775:CEE0.

Step 1: Divide the MAC address between the OUI (FC99:47) and device identifier (75:CEE0).

Step 2: Insert the hexadecimal value FFFE, which in binary is: 1111 1111 1111 1110 (FC99:47FF:FE75:CEE0).

Step 3: Convert the first 2 hexadecimal values of the OUI to binary and flip the U/L bit (bit 7). In this example, the 0 in bit 7 is changed to a 1 ((FC)₁₆=(11111100)₂ flipping the 7th bit it *turns into* (11111110)₂=(FE)₁₆).

The result is an EUI-64 generated Interface ID of FE99:47FF:FE75:CEE0.



Activity - EUI-64 process

Watch this [video](#) to better understand the procedure to create an EUI-64 IPv6 address.

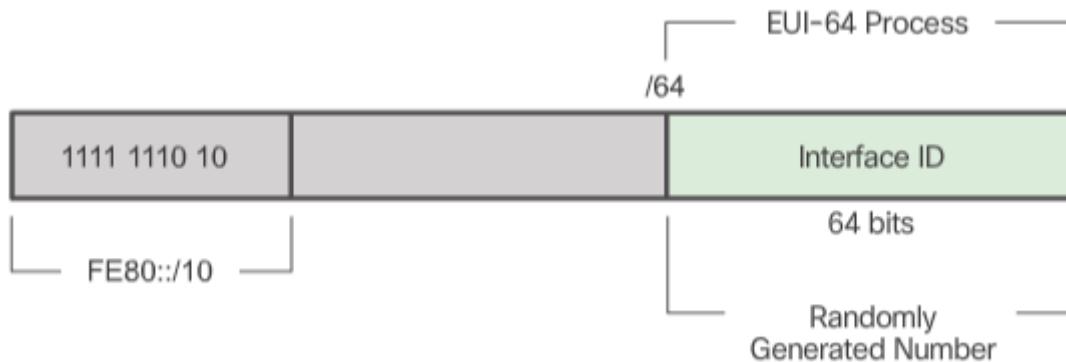
Given the following MAC addresses apply the EUI-64 process to determine the Interface ID.

MAC	EUI-64 Interface ID
000D:BD44:6E2B	
FC99.4775.C3E0	
60A4.4C40.74C9	

Dynamic Link-Local Addresses

All IPv6 devices must have an IPv6 link-local address. A link-local address can be established dynamically or configured manually as a static link-local address.

The figure below shows the link-local address is dynamically created using the FE80::/10 prefix and the Interface ID using the EUI-64 process or a randomly generated 64-bit number.



Operating systems will typically use the same method for both a SLAAC created global unicast address and a dynamically assigned link-local address, as shown in the following figure.

EUI-64 generated Interface ID

```
PCA> ipconfig
Windows IP Configuration
Ethernet adapter Local Area Connection:
  Connection-specific DNS Suffix  :
  IPv6 Address. . . . . : 2001:db8:acad:1:fc99:47ff:fe75:cee0
  Link-local IPv6 Address . . . . : fe80::fc99:47ff:fe75:cee0
  Default Gateway   . . . . . : fe80::1
```

Random 64-bit generated Interface ID

```
PCB> ipconfig
Windows IP Configuration
Ethernet adapter Local Area Connection:
  Connection-specific DNS Suffix  :
  IPv6 Address. . . . . : 2001:db8:acad:1:50a5:8a35:a5bb:66e1
  Link-local IPv6 Address . . . . : fe80::50a5:8a35:a5bb:66e1
  Default Gateway   . . . . . : fe80::1
```

Automatically configuration of an IPv6 link-local address on the interface while also enabling the interface for IPv6 processing is performed using the following command:

```
R1(config-if)#ipv6 enable
```

Cisco routers automatically create an IPv6 link-local address whenever a global unicast address is assigned to the interface. By default, Cisco IOS routers use EUI-64 to generate the Interface ID for all link-local address on IPv6 interfaces. For serial interfaces, the router will use the MAC address of an Ethernet interface. Recall that a link-local address must be unique only on that link or network. However, a drawback to using the dynamically assigned link-local address is its length, which makes it challenging to identify and remember assigned addresses. The figure below displays the MAC address on router R1 GigabitEthernet 0/0 interface. This address is used to dynamically create the link-local address on the same interface.

```
R1# show interface gigabitethernet 0/0
GigabitEthernet0/0 is up, line protocol is up
  Hardware is CN Gigabit Ethernet, address is fc99.4775.c3e0
  (bia fc99.4775.c3e0)
<Output Omitted>

R1# show ipv6 interface brief
GigabitEthernet0/0      [up/up]
  FE80::FE99:47FF:FE75:C3E0
  2001:DB8:ACAD:1::1
GigabitEthernet0/1      [up/up]
  FE80::FE99:47FF:FE75:C3E1
  2001:DB8:ACAD:2::1
Serial0/0/0             [up/up]
  FE80::FE99:47FF:FE75:C3E0
  2001:DB8:ACAD:3::1
Serial0/0/1             [administratively down/down]
  unassigned
R1#
```

The diagram shows the output of the 'show ipv6 interface brief' command on router R1. It lists three interfaces: GigabitEthernet0/0, GigabitEthernet0/1, and Serial0/0/0. Each interface has two IPv6 addresses: a link-local address (highlighted in orange) and a global unicast address. Orange arrows point from each of the three link-local addresses to a callout box labeled 'Link-local addresses using EUI-64'.

To make it easier to recognize and remember these addresses on routers, it is common to statically configure IPv6 link-local addresses on routers.

Static Link-Local Addresses

Configuring the link-local address manually provides the ability to create an address that is recognizable and easier to remember.

Link-local addresses can be configured manually using the same interface command used to create IPv6 global unicast addresses but with the additional `link-local` parameter. When an address begins with this hexet within the range of FE80 to FEBF, the `link-local` parameter must follow the address.

The following commands show the configuration of link-local addresses using the `ipv6 address` interface command. The link-local address `FE80::1` is used to make it easily recognizable as belonging to router R1. The same IPv6 link-local address is configured on all of R1 interfaces. `FE80::1` can be configured on each link because it only has to be unique on that link.

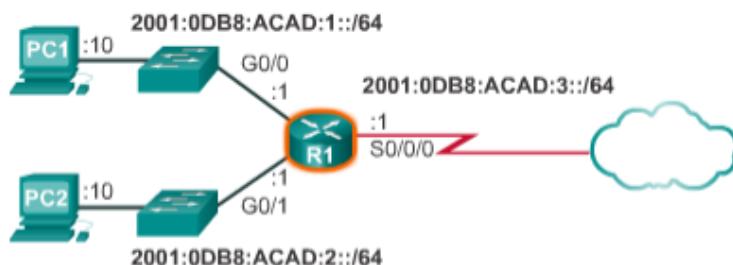
```
R1(config)#interface gigabitethernet0/0
R1(config-if)#ipv6 address fe80::1 link-local
R1(config-if)#no shutdown
R1(config-if)#exit

R1(config)#interface gigabitethernet0/1
R1(config-if)#ipv6 address fe80::1 link-local
R1(config-if)#no shutdown
R1(config-if)#exit

R1(config)#interface serial0/0/0
R1(config-if)#ipv6 address fe80::1 link-local
R1(config-if)#no shutdown
R1(config-if)#exit
```

Verifying IPv6 Address Configuration

As shown in the figure below, the command to verify the IPv6 interface configuration is similar to the command used for IPv4.



```
R1# show ipv6 interface brief
GigabitEthernet0/0      [up/up]
  FE80::FE99:47FF:FE75:C3E0
  2001:DB8:ACAD:1::1
GigabitEthernet0/1      [up/up]
  FE80::FE99:47FF:FE75:C3E1
  2001:DB8:ACAD:2::1
Serial10/0/0            [up/up]
  FE80::FE99:47FF:FE75:C3E0
  2001:DB8:ACAD:3::1
Serial10/0/1            [administratively down/down]
  unassigned
R1#
```

The `show interface` command displays the MAC address of the Ethernet interfaces. EUI-64 uses this MAC address to generate the Interface ID for the link-local address. Additionally, the `show ipv6 interface brief` command displays abbreviated output for each of the interfaces. The `[up/up]` output on the same line as the interface

indicates the Layer 1/Layer 2 interface state. This is the same as the Status and Protocol columns in the equivalent IPv4 command.

Notice that each interface has two IPv6 addresses. The second address for each interface is the global unicast address that was configured. The first address, the one that begins with FE80, is the link-local unicast address for the interface. Recall that the link-local address is automatically added to the interface when a global unicast address is assigned.

Also, notice that R1 Serial0/0/0 link-local address is the same as its GigabitEthernet0/0 interface. Serial interfaces do not have Ethernet MAC addresses, so Cisco IOS uses the MAC address of the first available Ethernet interface. This is possible because link-local interfaces only have to be unique on that link.

The link-local address of the router interface is typically the default gateway address for devices on that link or network.

As shown in the following figure, the `show ipv6 route` command can be used to verify that IPv6 networks and specific IPv6 interface addresses have been installed in the IPv6 routing table. The `show ipv6 route` command will only display IPv6 networks, not IPv4 networks.

```
R1# show ipv6 route
IPv6 Routing Table - default - 7 entries
Codes: C - Connected, L - Local, S - Static, U - Per-user
Static

<output omitted>

C 2001:DB8:ACAD:1::/64 [0/0]
    via GigabitEthernet0/0, directly connected
L 2001:DB8:ACAD:1::1/128 [0/0]
    via GigabitEthernet0/0, receive
C 2001:DB8:ACAD:2::/64 [0/0]
    via GigabitEthernet0/1, directly connected
L 2001:DB8:ACAD:2::1/128 [0/0]
    via GigabitEthernet0/1, receive
C 2001:DB8:ACAD:3::/64 [0/0]
    via Serial0/0/0, directly connected
L 2001:DB8:ACAD:3::1/128 [0/0]
    via Serial0/0/0, receive
L FF00::/8 [0/0]
    via Null0, receive
R1#
```

Within the route table, a **C** next to a route indicates that this is a directly connected network. When the router interface is configured with a global unicast address and is in the “up/up” state, the IPv6 prefix and prefix length is added to the IPv6 routing table as a connected route.

The `ping` command for IPv6 is identical to the command used with IPv4, except that an IPv6 address is used. As shown in the figure below, the command is used to verify Layer 3 connectivity between R1 and PC1. When pinging a link-local address from a router, Cisco IOS will prompt the user for the exit interface. Because the destination

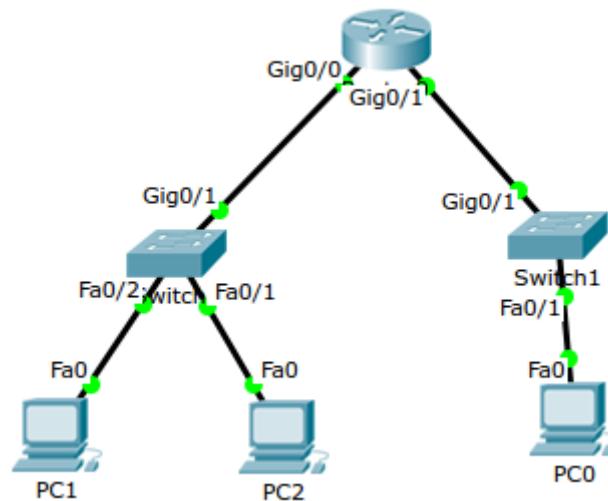
link-local address can be on one or more of its links or networks, the router needs to know which interface to send the ping to.

```
R1# ping 2001:db8:acad:1::10
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:ACAD:1::10, timeout
is 2 seconds:
!!!!!
Success rate is 100 percent (5/5)
R1#
```

Activity – Link-local connectivity

Watch [this](#) video in order to understand how link-local addresses are configured.

Now create the following [scenario](#) and configure link-local addresses for all the devices.



PC1 and PC2 are automatically configured to use link-local connectivity.

PC1

```
C:\>ipconfig  
FastEthernet0 Connection: (default port)  
Link-local IPv6 Address.....: FE80::20D:BDFF:FE44:6E2B  
IPv6 Address.....: ::/0  
Default Gateway.....: ::
```

How is the link-local made?

MAC 000D:BD44:6E2B

EUI-64 002D:BDFF:FE44:6E2B

$$\begin{aligned}(000D)_{16} &= (0000 \ 00\text{00} \ 0000 \ 1011)_2 \\(0260)_{16} &= (0000 \ 00\text{10} \ 0000 \ 1011)_2\end{aligned}$$

LINK-LOCAL

FE80:0000:0000:0000:020D:BDFF:FE44:6E2B
FE80::20D:BDFF:FE44:6E2B

PC1 and PC2 can ping each other because they have the link-local address configured.

ROUTER

Routing configuration has to be done using the following command:

```
Router(config) #ipv6 unicast-routing
```

After it has to be configured router interfaces that can have the same link-local address because these addresses have only local meaning.

```
Router(config) #interface GigabitEthernet0/0
Router(config-if) #ipv6 address FE80::1 link-local
Router(config-if) #no shutdown
Router(config-if) #exit
Router(config) #interface GigabitEthernet0/1
Router(config-if) #ipv6 address FE80::1 link-local
Router(config-if) #no shutdown
Router(config-if) #exit
```

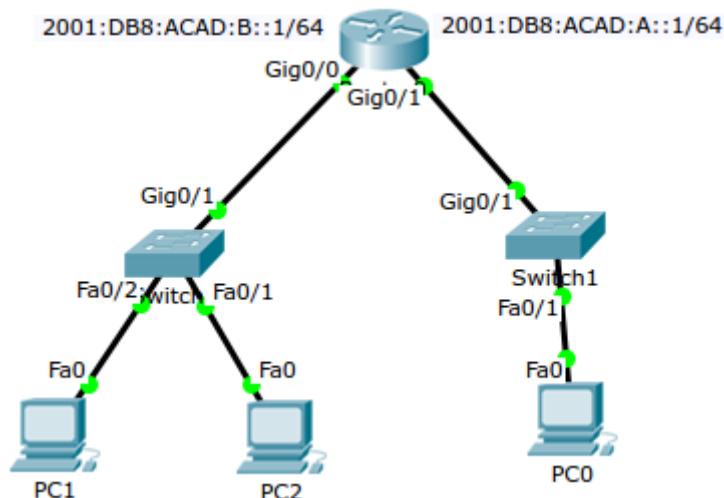
Ping each configured interfaces to verify connectivity.

Activity – Global unicast connectivity and SLAAC

Watch [this](#) video in order to understand how global unicast addresses are configured.

Now create the following scenario and configure global unicast addresses for all the devices.

The two networks can reach each other *whether* global unicast addresses are configured on router ports. Use the addresses shown in the figure.



ROUTER

```
Router(config)#interface GigabitEthernet0/0
Router(config-if)#ipv6 address 2001:DB8:ACAD:B::1/64
Router(config-if)#no shutdown
Router(config-if)#exit
Router(config)#interface GigabitEthernet0/1
Router(config-if)#ipv6 address 2001:DB8:ACAD:A::1/64
Router(config-if)#no shutdown
Router(config-if)#exit
```

```
Router#show ipv6 interface brief
GigabitEthernet0/0          [up/up]
  FE80::1
  2001:DB8:ACAD:B::1
GigabitEthernet0/1          [up/up]
  FE80::1
  2001:DB8:ACAD:A::1
Vlan1                      [administratively down/down]
```

PC1

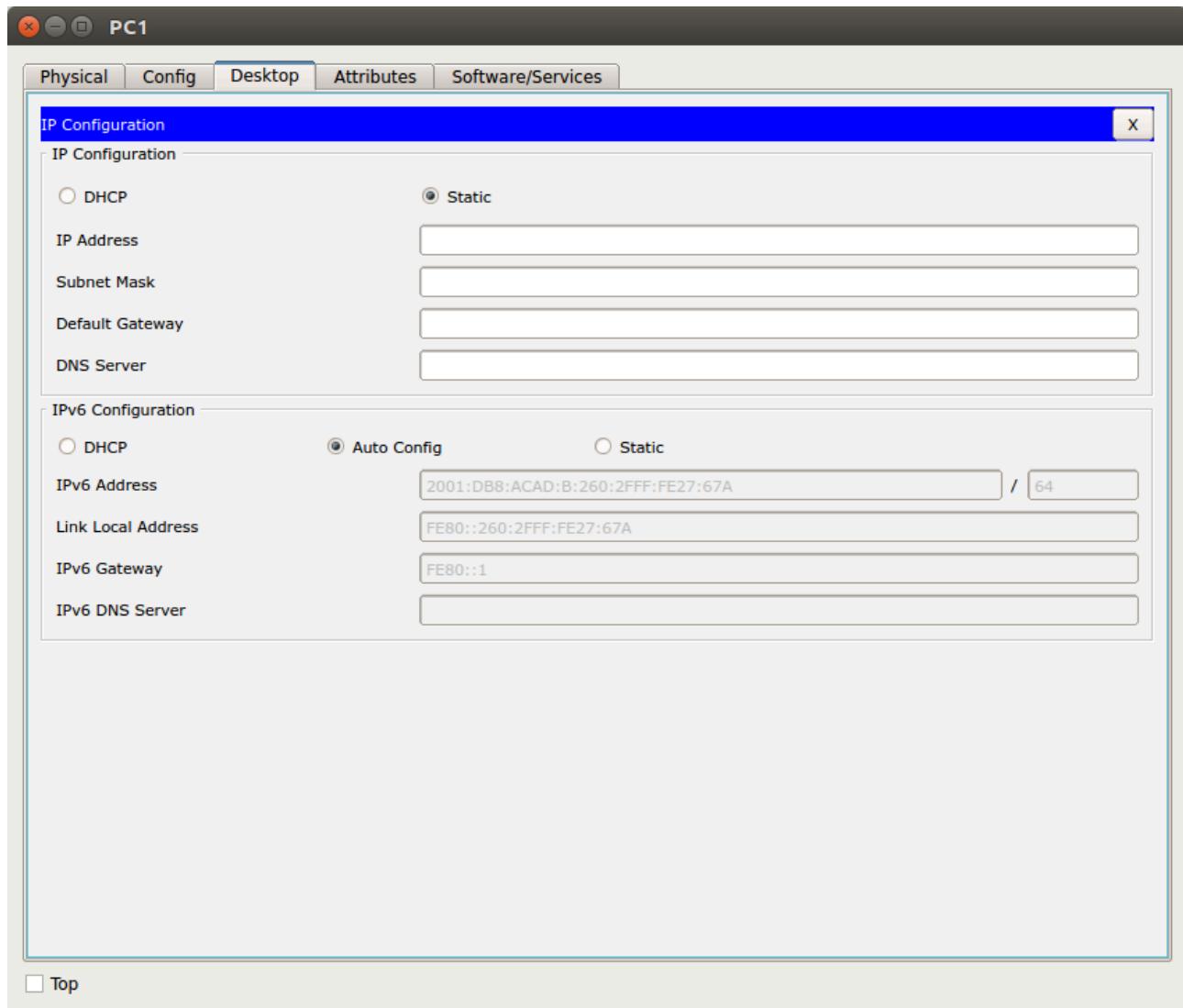
```
C:\>ipconfig

FastEthernet0 Connection: (default port)

Link-local IPv6 Address.....: FE80::260:2FFF:FE27:67A
```

```
IPv6 Address..... ::/0
Default Gateway..... :::
[...]
```

Proceed auto configuring PC1.



```
C:\>ipconfig
```

```
FastEthernet0 Connection: (default port)

Link-local IPv6 Address.....: FE80::260:2FFF:FE27:67A
IPv6 Address.....: 2001:DB8:ACAD:B:260:2FFF:FE27:67A/64
Default Gateway.....: FE80::1
[...]
```

Configure all PCs at the same way and try the connectivity using the ping command from PC1 to PC0.

C - Il livello Applicazione

Prefazione

I seguenti appunti sul livello applicativo di rete sono basati sul lavoro svolto dal Prof. Pecoraro e pubblicati sul [suo sito](#). A lui e a tutti gli autori e appassionati di conoscenza da lui citati nel sito, vanno i miei più sinceri ringraziamenti.

Libro vol.3 - Il livello delle applicazioni

STUDIARE: L. Lo Russo, E. Bianchi, Sistemi e Reti - Nuova Edizione OPENSCHOOL - Per l'articolazione INFORMATICA degli Istituti Tecnici settore Tecnologico, vol. 3, ed. Hoepli, 2017

- Il livello delle applicazioni nei modelli ISO/OSI e TCP/IP pp.2-8

Introduzione

Le reti di calcolatori hanno lo scopo di consentire la cooperazione tra le applicazioni di rete. Per poter comunicare tra loro, le applicazioni di rete si servono di protocolli. Sono state sviluppate numerose applicazioni per Internet, che sono diventate parte integrante delle attività quotidiane delle persone.

Le prime applicazioni per Internet gestivano solo il testo: messaggi di posta elettronica, terminale virtuale, trasferimento di file e newsgroup (bacheche in cui venivano affissi articoli, recensioni, ecc.). Agli inizi degli anni '90 nacque il World Wide Web, una tecnologia che sfruttava il canale telefonico per dare agli utenti la possibilità di ricercare e consultare documenti, e realizzare il commercio elettronico. Le successive applicazioni create per Internet furono la messaggistica istantanea e la condivisione dei file, un modello di comunicazione P2P. Agli inizi del 2000 sono nate le applicazioni per far viaggiare sulla rete anche la voce e i video, le principali applicazioni sono voice-over-IP (VoIP) e la videoconferenza su IP come Skype; si è diffusa anche la distribuzione di video amatoriali come YouTube; e la trasmissione di film su richiesta come Netflix. In questo stesso periodo si sono sviluppati anche i giochi in rete, come Second Life e World of Warcraft. La nuova generazione di applicazioni riguarda i social network, come Facebook e Twitter, che hanno stabilito una rete di relazioni tra le persone che si appoggia sulla rete Internet formata da router e canali.

Generalità sulle applicazioni di rete

Le applicazioni di rete devono essere concepite per trovarsi in esecuzione su sistemi terminali diversi e possano comunicare tramite la rete. Ad esempio, in un'applicazione Web ci sono due programmi che comunicano tra loro: il browser in esecuzione sul computer utente (desktop, laptop, tablet, smartphone, ...); e il server Web in esecuzione su un altro computer. Oppure, in una applicazione P2P di condivisione di file c'è un programma in ogni host che partecipa alla comunità di condivisione dei file. In questo caso i programmi nei vari host possono essere simili o identici.

L'applicazione di rete può essere scritta, ad esempio, in C, in Java, in Python, e si serve delle funzionalità offerte dalle apparecchiature di rete, come router o switch, che lo sviluppatore usa senza preoccuparsi di programmarle.

Architettura delle Applicazioni di rete

Come regola, prima di addentrarsi nella codifica del software, si deve schematizzare l'architettura dell'applicazione. Dal punto di vista dello sviluppatore di applicazioni, l'architettura di rete è fissa e fornisce un insieme specifico di servizi alle applicazioni. L'architettura dell'applicazione, invece, viene progettata dallo sviluppatore e descrive come l'applicazione è strutturata sui vari sistemi terminali, secondo uno dei due modelli: **client-server** o **peer-to-peer** (P2P).

Nell'**architettura client-server**, c'è sempre un host, chiamato server, che soddisfa le richieste degli altri host, chiamati client. Un esempio classico è il Web. Un server Web, sempre attivo, risponde alle richieste dei browser in esecuzione sugli host client. Quando il server Web riceve una richiesta da un client, risponde inviando l'oggetto richiesto al client. Si noti che, con l'architettura client-server, i client non comunicano direttamente tra loro; per esempio nel Web, due browser non comunicano direttamente. Un'altra caratteristica dell'architettura client-server è che il server ha un indirizzo IP fisso e noto. Un client può sempre contattare il server inviando un pacchetto al suo indirizzo IP. Alcune delle applicazioni più note con architettura client-server sono il Web, FTP, Telnet e l'e-mail.

Nell'architettura client-server il livello di traffico che deve essere sostenuto dal server è un fattore cruciale. Un singolo server riesce a sostenere richieste da parte di molti client. Ma un sito di social networking molto popolare può congestionarsi rapidamente se ha un solo server che gestisce tutte le sue richieste. In questi casi viene usato un **data center**, costituito da un **potente server virtuale che bilancia il carico di lavoro tra numerosi host**. Molti servizi che si trovano su Internet, come i motori di ricerca (Google), il commercio elettronico (Amazon ed eBay), la posta elettronica con interfaccia Web (Gmail e Yahoo) e i social network (Facebook e Twitter) impiegano uno o più data center. [Google](#) ha da 30 a 50 data center distribuiti in tutto il mondo, che insieme gestiscono i servizi di Google: la ricerca, YouTube, Gmail e altro. Un data center può avere centinaia di migliaia di server, che devono essere alimentati e manutenzionati. Inoltre, richiedono una elevata larghezza di banda.

In un'**architettura P2P**, non c'è nessun server dedicato in un data center. L'applicazione sfrutta la **comunicazione diretta tra coppie di host che si connettono occasionalmente, chiamati peer**. I peer non sono di proprietà del fornitore di servizi, ma sono invece computer controllati dagli utenti, in casa o in ufficio. L'architettura è chiamata peer-to-peer perché **i peer comunicano senza passare attraverso un server dedicato**. Molte delle applicazioni più popolari e ad alta intensità di traffico sono basate su architetture P2P. Queste applicazioni includono la condivisione di file (ad esempio, BitTorrent), l'accelerazione del download assistito (ad esempio Xunlei), la Telefonía (Skype), e IPTV, la diffusione radio e televisiva (ad esempio, Kankan e PPStream). Alcune applicazioni hanno **architetture ibride**, combinano elementi client-server ed elementi P2P. Ad esempio, per molte applicazioni di messaggistica istantanea, **i server vengono utilizzati per mantenere gli indirizzi IP degli utenti, ma i messaggi tra gli utenti vengono inviati direttamente tra gli host degli utenti**, senza passare attraverso il server.

Con il termine peer si indica un computer che può svolgere contemporaneamente il ruolo di client e il ruolo di server. È detto anche un computer paritetico, nel senso che è di pari grado agli altri. In un server l'amministratore decide, per ogni categoria di utente, le risorse a cui può accedere, in un peer è l'utente che decide le risorse da condividere con qualsiasi altro utente.

Una delle caratteristiche più interessanti dell'architettura P2P è l'auto-scalabilità. Ad esempio, in un'applicazione di condivisione di file P2P, quando un sistema scarica un file, lo può poi rendere disponibile agli altri peer, che in tal modo non dovranno sovraccaricare lo stesso sistema. Inoltre le architetture P2P non richiedono un'infrastruttura server e una larghezza di banda adeguata.

La scalabilità di una rete o di un'applicazione è la proprietà di variare la propria dimensione, secondo un fattore di scala, senza che venga degradato il servizio.

Comunicazione tra Processi

Prima di costruire l'applicazione di rete, è necessario conoscere il modo in cui i processi, in esecuzione su più sistemi terminali, comunicano tra di loro. Quando i processi in esecuzione sullo stesso sistema devono comunicare tra loro, applicano le regole del sistema operativo. Ma se i due processi si trovano su due sistemi terminali diversi, comunicano tra loro scambiandosi messaggi attraverso la rete.

Un processo mittente crea ed invia messaggi sulla rete; un processo destinatario riceve questi messaggi ed eventualmente risponde con altri messaggi. I processi che comunicano tra loro risiedono nel livello applicazione.

Processi Client e processi Server

Un'applicazione di rete è costituita da due processi che si scambiano messaggi attraverso la rete. Ad esempio, nell'applicazione Web un browser scambia messaggi con un server Web. In un sistema di condivisione di file P2P, un file viene trasferito da

un processo in esecuzione su un peer ad un processo in un esecuzione su un altro peer. In ogni coppia di processi in comunicazione, uno dei due viene denominato il client e l'altro il server. Nel Web, un browser è un processo client e un server Web è un processo server. Con la condivisione di file P2P, il peer che sta scaricando il file viene chiamato client e il peer che sta fornendo il file viene chiamato server.

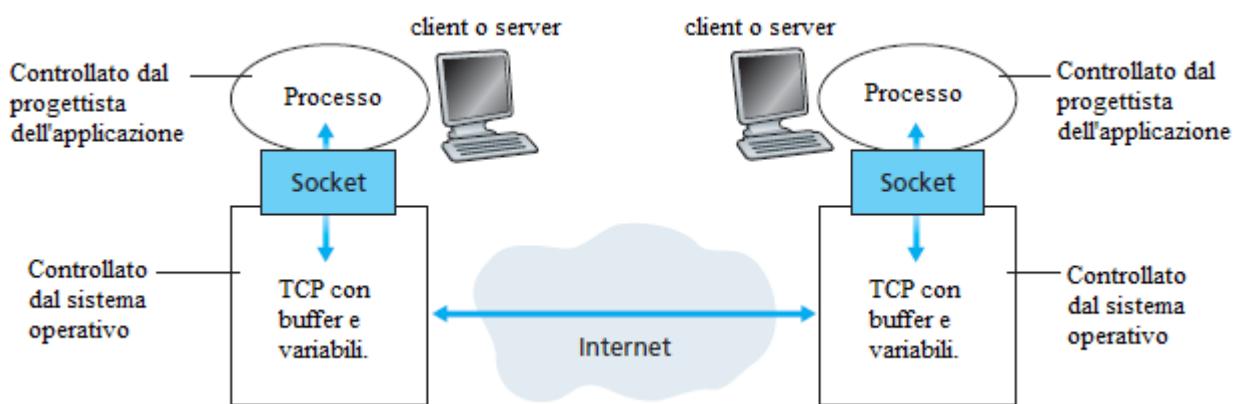
Nelle applicazioni di tipo P2P, un processo può essere sia un client che un server. In effetti, tale ruolo viene assunto solo per la durata di una data sessione di comunicazione. I processi client e server sono definibili nel seguente modo:

Nel contesto di una sessione di comunicazione tra una coppia di processi, il processo che avvia la comunicazione (cioè, contatta per primo l'altro processo) viene etichettato come il client. Il processo che attende di essere contattato per iniziare la sessione è il server.

Nel Web, un browser inizia la comunicazione con un server Web; quindi il browser è il client e il server Web è il server. Nella condivisione di file P2P, quando il Peer A chiede al Peer B di inviargli un file, il Peer A è il client e il Peer B è il server, ma solo durante questa sessione di comunicazione. A volte si usa anche la terminologia "lato client e lato server di un'applicazione".

L'interfaccia tra il Processo e la rete di calcolatori

Ogni messaggio che un processo invia all'altro processo deve passare attraverso la rete. Un processo del livello applicazione invia e riceve messaggi attraverso un'interfaccia chiamata socket.



La figura mostra la comunicazione mediante i socket tra due processi che si scambiano messaggi usando la rete. (In figura si assume che il protocollo di trasporto utilizzato dai processi sia TCP). Come mostrato in questa figura, un **socket è l'interfaccia tra il livello applicazione e il livello trasporto all'interno di un host**. Un socket è una **Application Programming Interface (API)** tra l'applicazione e la rete, in quanto **il socket mette a disposizione del programmatore le funzionalità per costruire le applicazioni di rete**. Lo sviluppatore dell'applicazione ha il controllo sul lato dell'applicazione ma non può modificare il livello di trasporto. Le sole possibilità che ha lo sviluppatore dell'applicazione sul livello trasporto sono:

1. la scelta del protocollo di trasporto

2. l'impostazione di alcuni parametri quali le dimensioni massime del buffer e del segmento.

Una volta che lo sviluppatore dell'applicazione sceglie un protocollo di trasporto, l'applicazione viene costruita utilizzando i servizi che questo rende disponibili.

Indirizzamento dei Processi

Un processo in esecuzione su un host, può inviare pacchetti ad un processo in esecuzione su un altro host solo se questo possiede un indirizzo. Per identificare il processo ricevente devono essere specificate due informazioni:

1. l'indirizzo dell'host;
2. un identificatore che specifica il processo nello host di destinazione.

In Internet, un **host** è identificato dal suo **indirizzo IP**. Un indirizzo IP è un numero di 32 bit che identifica in modo univoco lo host. Oltre a conoscere l'indirizzo dello host a cui è destinato un messaggio, il processo mittente deve anche **identificare il processo** destinatario (**più esattamente, il socket ricevente**) in esecuzione sullo host. Queste informazioni sono necessarie perché, in generale, su un host ci potrebbero essere molte applicazioni di rete in esecuzione. A questo scopo viene usato un **numero di porta** di destinazione. Alle applicazioni più diffuse sono stati assegnati dei numeri di porta specifici. Ad esempio, un server Web è in ascolto sulla porta numero 80, mentre un server di posta (che utilizza il protocollo SMTP) è in ascolto sulla porta numero 25.

Servizi del livello Trasporto disponibili alle applicazioni

Un socket è l'interfaccia tra il processo del livello applicazione e il protocollo del livello trasporto. Dal lato del mittente, l'applicazione passa il messaggio al socket e il protocollo del livello Trasporto preleva i messaggi dal socket.

A livello di trasporto sono disponibili diversi protocolli, e quando si sviluppa un'applicazione di rete è necessario sceglierne uno. Per scegliere opportunamente il protocollo di trasporto si devono conoscere i servizi forniti dai vari protocolli di questo livello, e poi scegliere quello con i servizi che meglio corrispondono alle esigenze dell'applicazione di rete.

Le caratteristiche dei possibili servizi di un protocollo del livello Trasporto sono:

- trasferimento affidabile dei dati;
- flusso (throughput);
- temporizzazione;
- sicurezza.

Trasferimento affidabile o non affidabile dei dati

I pacchetti possono perdere all'interno della rete. Per esempio, un pacchetto può venire sovrascritto mentre si trova nel buffer di un router congestionato, o può essere scartato da un host perché illeggibile in conseguenza del rumore che lo ha modificato sul canale. **Per molte applicazioni**, come la posta elettronica, il trasferimento di file, l'accesso remoto, la consultazione di pagine Web o le transazioni bancarie, la perdita di dati può avere conseguenze devastanti. Pertanto, per utilizzare queste applicazioni, **si deve garantire che i dati in partenza da un sistema terminale vengano consegnati correttamente e senza perdita all'altro sistema terminale**. Se un protocollo fornisce un servizio con garanzia della consegna dei dati, da processo a processo, si dice che **il trasferimento dei dati è affidabile**. Quando un protocollo di trasporto fornisce questo servizio, il processo mittente che passa i suoi dati mediante il socket ha la certezza che i dati arriveranno senza errori al processo di destinazione.

Quando un protocollo del livello trasporto non fornisce il trasferimento affidabile dei dati, alcuni dei dati inviati dal processo potrebbero non arrivare mai al processo destinazione. Questo può essere accettabile per le applicazioni che possono tollerare la perdita di qualche pacchetto, ad esempio per le applicazioni multimediali come le conversazioni audio/video. In queste applicazioni multimediali, i dati persi non compromettono la comprensione del messaggio.

Flusso (throughput)

Il throughput, nel contesto di una sessione di comunicazione tra due processi collegati tramite una rete, è il tasso a cui il processo sorgente consegna bit al processo ricevente. Poiché altre sessioni di comunicazione condividono il canale, e poiché altre sessioni si aggiungono o terminano, il **throughput disponibile varia nel tempo**.

Queste osservazioni portano a un **altro servizio** naturale che un **protocollo** a livello di **trasporto** potrebbe fornire, cioè, **garantire il throughput ad un certo tasso specificato**. Con tale servizio, l'applicazione potrebbe richiedere una velocità garantita di **r** bit/sec, e il protocollo di trasporto dovrebbe quindi assicurare che il throughput disponibile resti sempre almeno **r** bit/sec. Tale servizio di throughput garantito serve a molte applicazioni. Ad esempio, se un'applicazione di telefonia su Internet codifica la voce a 32 kbps, è necessario che i dati inviati sulla rete vengano consegnati all'applicazione ricevente a questo ritmo. Se il protocollo di trasporto non può fornire tale velocità, l'applicazione ha bisogno di codificare a un tasso inferiore (e di ricevere dati dal processo a velocità inferiore) o potrebbe dover rinunciare, in quanto, ad esempio, la minore velocità renderebbe inutilizzabile l'applicazione di telefonia. **Le applicazioni che hanno requisiti di throughput si dicono applicazioni bandwidth-sensitive.** Molte applicazioni multimediali attuali dipendono dalla larghezza di banda, anche se alcune applicazioni multimediali possono utilizzare tecniche di codifica adattative per codificare voce o video digitalizzato ad una velocità che corrisponde alla velocità realmente disponibile.

Le applicazioni dipendenti dalla larghezza di banda hanno requisiti di throughput specifici, le applicazioni flessibili, invece, usano un elevato o un basso throughput, a seconda della disponibilità della banda. La posta elettronica, il trasferimento di file e le pagine Web sono applicazioni flessibili. Naturalmente, si desidera massimizzare il flusso.

Temporizzazione

Un **protocollo del livello trasporto** può anche fornire **garanzie di temporizzazione**. Come con il flusso garantito, la temporizzazione può essere garantita in varie forme. Ad esempio si potrebbe garantire che ogni bit consegnato dal mittente nel socket impiegherà non più di 100 ms per giungere al socket del ricevitore. Questo servizio dovrebbe soddisfare applicazioni interattive in tempo reale, come la telefonia, la teleconferenza, e i giochi multiplayer. Ognuna di **queste applicazioni richiede che la consegna dei dati avvenga entro limiti di tempo ben definiti, altrimenti sono inefficaci**. Lunghi ritardi nella telefonia via Internet, per esempio, introducono pause innaturali nella conversazione; in una partita multigiocatore o in un ambiente interattivo virtuale, un intervallo lungo tra il prendere un'azione e l'osservare la reazione dall'ambiente (ad esempio, da un altro giocatore all'altro sistema terminale di una connessione) rende l'applicazione meno realistica. Per le applicazioni non real-time, il minimo ritardo è gradito, ma non rappresenta un vincolo.

Sicurezza

Un **protocollo di trasporto** può **fornire uno o più servizi di sicurezza**. Ad esempio, nello host mittente, un protocollo di trasporto può cifrare tutti i dati trasmessi dal processo sorgente e, nello host di destinazione, il protocollo di trasporto può decodificare i dati prima di consegnarli al processo ricevente. **Tale servizio fornirebbe la riservatezza della comunicazione ai due processi**. Un protocollo di trasporto può anche fornire **altri servizi di sicurezza** in aggiunta alla riservatezza, tra cui l'**integrità** dei dati e l'**autenticazione** dei processi.

Servizi di Trasporto offerti da Internet

Fino a questo punto, sono stati considerati i servizi di trasporto che una rete di computer potrebbe fornire in generale. Adesso, più nello specifico, si esamina il tipo dei servizi di trasporto forniti da Internet. Internet e più in generale **le reti TCP/IP, offrono due protocolli di trasporto alle applicazioni: UDP e TCP**. Quando uno sviluppatore crea una nuova applicazione di rete, una delle prime decisioni da prendere è se utilizzare UDP o TCP. Ciascuno di questi protocolli offre un diverso insieme di servizi alle applicazioni. La tabella seguente mostra i requisiti di servizio per alcune applicazioni.

Applicazione	Perdita di pacchetti	Throughput	Temporizzazione
File transfer/download	Senza perdita	Flessibile	No
E-mail	Senza perdita	Flessibile	No
Pagine Web	Senza perdita	Flessibile (pochi kbps)	No
Telefonia su Internet Video conferenza	leggera tolleranza	Audio: da pochi kbps a 1 Mbps Video: da 10 kbps a 5 Mbps	Sì: dell'ordine delle centinaia di msec
Streaming di file audio/video	leggera tolleranza	Audio: da pochi kbps a 1 Mbps Video: da 10 kbps a 5 Mbps	Sì: pochi secondi
Giochi Interattivi	leggera tolleranza	da pochi kbps a 10 kbps	Sì: dell'ordine delle centinaia di msec
Instant messaging	Senza perdita	Flessibile	Sì e No

Servizi TCP

Il **TCP** include un servizio **orientato alla connessione** (apertura connessione tramite 3-way handshake, connessione full-duplex e chiusura connessione), un servizio di **trasferimento dati affidabile** (dati senza errori, consegna all'applicazione in ordine, conferma dei byte inviati) e un meccanismo di **controllo della congestione**, sia a **livello di host** (sliding window), sia a **livello di rete** (slow start).

Quando un'applicazione usa TCP come protocollo di trasporto, l'applicazione riceve tutti questi servizi da TCP.

Sicurezza di TCP

Né TCP né UDP applicano la crittografia. I dati che il processo mittente consegna al protocollo di trasporto attraverso il socket sono gli stessi dati trasmessi attraverso la rete verso il processo di destinazione. Così, per esempio, se il processo mittente invia una password in chiaro (cioè non criptata) nel suo socket, la password in chiaro viaggerà su tutti i collegamenti tra mittente e destinatario, con il rischio che, in un punto qualsiasi, venga intercettata e scoperta. Poiché la riservatezza è un

requisito fondamentale per molte applicazioni, TCP può essere preceduto da un Secure Sockets Layer (SSL).

TCP in combinazione con SSL aggiunge, ai servizi TCP, servizi di sicurezza nella comunicazione da processo a processo, ivi compresa la cifratura, l'integrità dei dati e l'autenticazione.

SSL non è un terzo protocollo di trasporto, come TCP e UDP, ma è un **miglioramento di TCP**, usato dal processo del livello applicazione. In particolare, se un'applicazione vuole utilizzare i servizi di SSL, deve includere il codice SSL (le librerie e le classi) sia nel client sia nel lato server dell'applicazione. SSL ha una propria API del socket che è simile al tradizionale API del socket TCP. Quando un'applicazione utilizza SSL, il processo mittente passa il testo in chiaro al socket di SSL; SSL nello host sorgente critta i dati e li consegna al socket TCP. I dati criptati viaggiano su Internet fino al socket TCP nello host destinazione. Da questo socket i dati criptati giungono a SSL, che decripta i dati. Infine, SSL passa i dati in chiaro attraverso il socket SSL al processo ricevente.

Servizi UDP

UDP è un **protocollo di trasporto leggero**, che offre servizi minimi. **UDP è senza connessione**, quindi non c'è sincronizzazione prima che i due processi inizino a comunicare. **UDP non fornisce un servizio di trasferimento dei dati affidabile** cioè, quando un processo invia un messaggio in un socket UDP, il protocollo UDP non fornisce alcuna garanzia che il messaggio raggiungerà il processo destinatario. Inoltre, **i messaggi** che arrivano al processo destinatario **possono arrivare fuori ordine**. **UDP non prevede un meccanismo di controllo della congestione**, di conseguenza il processo sorgente di UDP può passare dati al livello sottostante (livello di rete) ogni volta che ha dati da trasferire.

Servizi non forniti dai protocolli di Trasporto di Internet

I servizi del protocollo di trasporto necessari alle applicazioni di rete sono il trasferimento affidabile dei dati, il flusso, la temporizzazione e la sicurezza.

TCP fornisce il trasferimento affidabile dei dati end-to-end e, con SSL, fornisce servizi di sicurezza alle applicazioni. **Ma ne TCP, ne UDP forniscono garanzie di temporizzazione e di flusso.** Questo significa che applicazioni come la telefonia via Internet non potrebbero essere eseguite in Internet. Queste applicazioni spesso funzionano abbastanza bene perché sono state progettate con funzionalità che sopperiscono, per quanto possibile, a questa mancanza di garanzia.

Applicazioni quali e-mail, accesso a terminali remoti, il Web, e il trasferimento di file usano TCP. Per queste applicazioni si è scelto TCP soprattutto perché fornisce il trasferimento affidabile dei dati, garantendo che tutti i dati giungano a destinazione. Poiché le applicazioni di telefonia (come Skype) possono tollerare qualche perdita, ma richiedono un tasso minimo per essere efficaci, gli sviluppatori delle applicazioni di telefonia di solito preferiscono eseguire le applicazioni su UDP, aggirando in tal modo il

meccanismo di controllo della congestione di TCP accettando il rischio di perdita dei pacchetti. Ma poiché molti firewall sono configurati per bloccare il traffico UDP, le applicazioni di telefonia sono progettate per ripiegare sul protocollo TCP se la comunicazione UDP fallisce.

Protocolli del livello Applicazione

I processi di rete comunicano tra loro mediante l'invio di messaggi attraverso i socket. **Un protocollo del livello applicazione definisce il modo in cui i processi applicativi**, in esecuzione su sistemi terminali diversi, **si scambiano messaggi**. In particolare, **un protocollo del livello applicazione definisce**:

- il **tipo dei messaggi scambiati**, per esempio, i messaggi di richiesta e i messaggi di risposta;
- la **sintassi dei vari tipi di messaggio**, ad esempio in quali campi è strutturato e come questi sono delimitati;
- la **semantica dei campi**, cioè il significato delle informazioni contenute nei campi;
- le **regole** per determinare quando e come un processo **invia messaggi e risponde ai messaggi**.

Alcuni protocolli del livello applicazione sono specificati negli RFC (Request For Comment) e sono quindi di pubblico dominio. Ad esempio, il protocollo HTTP (HyperText Transfer Protocol) è disponibile nello RFC 2616. Se uno sviluppatore di browser segue le regole di questo RFC, il browser sarà in grado di recuperare le pagine Web da qualsiasi server Web che rispetti le stesse regole.

Molti altri protocolli del livello applicazione sono proprietari e volutamente non disponibili al pubblico dominio. Ad esempio, Skype utilizza protocolli proprietari del livello applicazione.

È importante **distinguere tra applicazioni di rete e protocolli del livello applicazione**. **Un protocollo del livello applicazione è solo un pezzo di un'applicazione di rete**. Ad esempio **il Web è un'applicazione client-server** che consente agli utenti di ottenere a richiesta i documenti da un server Web. L'applicazione Web è costituita da molti componenti, tra cui uno standard per il formato dei documenti (HTML), un browser (Firefox, Google Chrome), un server Web (Apache) e di un protocollo del livello applicazione. **Il protocollo del livello applicazione del Web è HTTP che definisce il formato e la sequenza di messaggi scambiati tra il browser e il server Web**. HTTP è solo un pezzo dell'applicazione Web.

Come altro esempio si può considerare l'applicazione di posta elettronica, anch'essa costituita da molti componenti, tra cui i server di posta che ospitano le caselle postali degli utenti, i client di posta elettronica (Outlook, Thunderbird) che permettono agli utenti di leggere e comporre messaggi, uno standard per definire la struttura di un messaggio di posta elettronica, i protocolli del livello applicazione che definiscono il modo in cui i messaggi vengono passati tra server, il modo in cui i messaggi vengono passati tra server e client di posta e il modo in cui il contenuto delle intestazioni dei messaggi devono essere interpretati. I protocolli del livello applicazione per la posta

elettronica sono SMTP (Simple Mail Transfer Protocol) [[RFC 5321](#)], POP3 (Post Office Protocol) [[RFC 1939](#)] e IMAP (Internet Message Access Protocol) [[RFC 3501](#)].

Il protocollo HTTP

Libro vol.3 - Il livello delle applicazioni

STUDIARE: L. Lo Russo, E. Bianchi, Sistemi e Reti - Nuova Edizione OPENSCHOOL - Per l'articolazione INFORMATICA degli Istituti Tecnici settore Tecnologico, vol. 3, ed. Hoepli, 2017

- Il Web:HTTP e FTP pp.9-17

Introduzione

Fino al 1990 la rete Internet era utilizzata principalmente da ricercatori, accademici e studenti universitari per accedere a host remoti, per trasferire file da host locali a host remoti e viceversa, per ricevere e inviare notizie e per ricevere e inviare posta elettronica. Sebbene queste applicazioni erano (e continuano ad essere) estremamente utili, Internet era sostanzialmente sconosciuta al di fuori delle comunità accademiche e di ricerca. Poi, dal 1990, una nuova importante applicazione è arrivata sulla scena, il World Wide Web. Il **Web** è stata la prima **applicazione Internet** che ha catturato l'interesse del pubblico in generale. Il modo in cui le persone interagiscono dentro e fuori dei loro ambienti di lavoro è cambiato e continua a cambiare radicalmente. Internet è passata da una delle tante reti dati alla sola e unica rete dati.

Il Web funziona su richiesta, in quanto gli utenti ricevono quello che vogliono, quando lo vogliono. Questo è diverso dalle trasmissioni radio e televisive tradizionali, che costringono gli utenti ad aspettare quando il fornitore rende disponibili i contenuti. Sul Web ogni persona può pubblicare articoli a costi estremamente bassi. I collegamenti ipertestuali e i motori di ricerca aiutano a navigare attraverso i siti web. La grafica produce effetti molto efficaci. Ci sono molte tecniche per interagire con le pagine web. Sul web si appoggiano le più diffuse applicazioni nate dopo il 2003, tra cui YouTube, Gmail e Facebook

Panoramica di HTTP

L'**HyperText Transfer Protocol (HTTP)** è il **protocollo del Web**. È definito negli [RFC 1945](#) e [RFC 2616](#) (e successivi). **HTTP è implementato in due programmi, uno client e uno server**. Il programma client e il programma server, **in esecuzione su sistemi terminali diversi, dialogano tra loro scambiandosi messaggi**. HTTP definisce la struttura di questi messaggi e il modo in cui il client e il server si scambiano i messaggi.

Una pagina Web (chiamata anche un documento) consiste di oggetti. Un oggetto è semplicemente un file, ad esempio un file HTML, un'immagine JPEG, una applet Java, o un videoclip, indirizzabili con un singolo URL. La maggior parte delle pagine Web sono costituite da un file HTML di base e diversi oggetti riferiti. Ad esempio, se una

pagina Web contiene un testo HTML e cinque immagini JPEG, la pagina Web ha sei oggetti: il file HTML di base più le cinque immagini. Il file HTML di base contiene i riferimenti (URL - *Uniform Resource Locator*) agli altri oggetti da mostrare nella pagina. Ogni URL ha due componenti: il nome host del server che ospita l'oggetto e il percorso dell'oggetto. Ad esempio, l'URL:

`http://www.scuola.edu/dipartimento/figura.gif`

ha **www.scuola.edu** come hostname e **/dipartimento/figura.gif** come nome di percorso.

Poiché il lato client di HTTP è implementato dai browser Web (come Firefox e Google Chrome), nel contesto del Web i due termini browser e client sono usati spesso come sinonimi. I Server Web, che implementano il lato server dell'HTTP, ospitano gli oggetti indirizzabili con un URL e richiesti dal client.

HTTP definisce come i client Web debbano richiedere le pagine Web ai server Web e come i server Web debbano trasferire le pagine Web ai client.

Quando un utente richiede una pagina Web (ad esempio, fa clic su un collegamento ipertestuale), il browser invia al server i messaggi di richiesta HTTP per gli oggetti contenuti nella pagina. Il server riceve le richieste e invia a sua volta i messaggi di risposta HTTP che contengono gli oggetti.

Dato che **il protocollo HTTP utilizza TCP come protocollo di trasporto sottostante**, il client deve chiedere di aprire una connessione TCP con il server. Una volta stabilita la connessione, i processi del client (browser) e del server hanno accesso al protocollo TCP attraverso le loro interfacce socket. Sul lato client dell'interfaccia socket c'è la porta tra il processo client e la connessione TCP; sul lato server c'è la porta tra il processo server e la connessione TCP.

Attraverso l'interfaccia del socket, il client invia messaggi di richiesta HTTP e riceve i messaggi di risposta HTTP inviati dal server. Allo stesso modo, attraverso la sua interfaccia socket, il server riceve i messaggi di richiesta HTTP del client ed invia i messaggi di risposta HTTP. Una volta che il client invia un messaggio sulla sua interfaccia socket, il messaggio è sotto il controllo del protocollo TCP.

Dato che il protocollo TCP fornisce a HTTP un servizio di trasferimento affidabile dei dati, ciò significa che ciascun messaggio di richiesta HTTP inviato da un processo client arriverà intatto al server. Analogamente, ciascun messaggio di risposta HTTP inviato dal processo server arriverà intatto al client. Qui si vede uno dei grandi vantaggi dell'architettura stratificata: **HTTP non si deve preoccupare della perdita dei dati o di come TCP recupera i dati persi né come questi verranno riordinati, in quanto questo è il compito del protocollo TCP e dei protocolli dei livelli sottostanti.**

Si consideri inoltre che due browser diversi possono interpretare (cioè mostrare all'utente) la stessa pagina Web in modi differenti. HTTP non ha nulla a che fare con il modo in cui una pagina Web viene interpretata da un client. **Le specifiche HTTP**

definiscono solo il protocollo di comunicazione tra il programma client HTTP e il programma server HTTP.

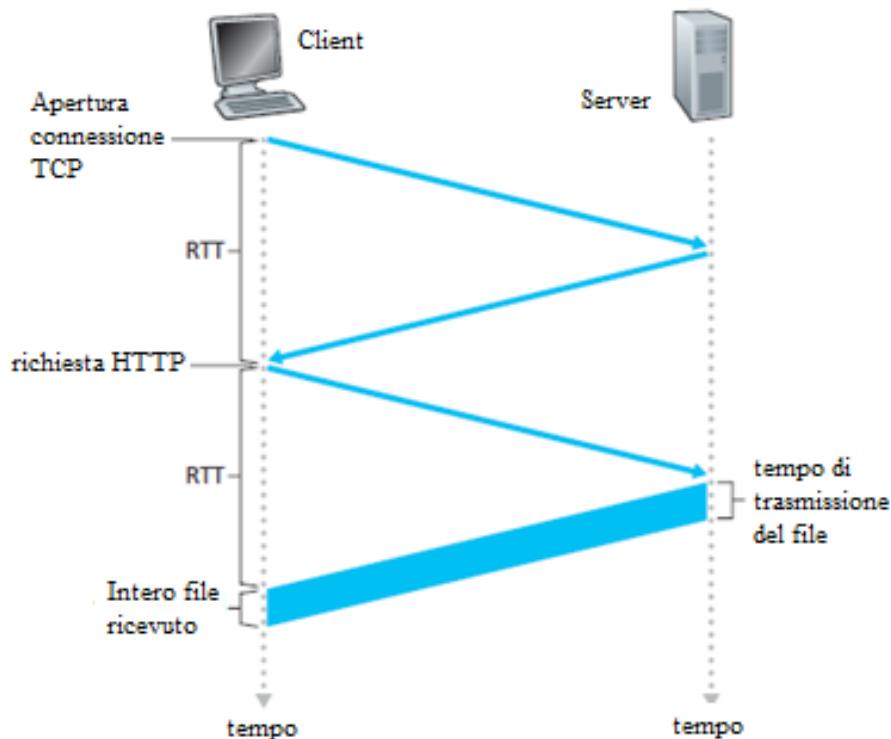
Protocollo stateless

È importante notare che il server invia un file al client che lo ha richiesto senza memorizzare le informazioni relative al client. Se un particolare client richiede lo stesso oggetto due volte a distanza di pochi secondi il server invierà nuovamente l'oggetto, non conservando alcuna memoria di quello che ha fatto in precedenza. **Poiché un server HTTP non mantiene le informazioni sui client, HTTP si dice che è un protocollo stateless.**

Round Trip Time (RTT)

Il **RTT** identifica **il tempo impiegato da una interazione di richiesta e risposta**, cioè il tempo impiegato dalla richiesta del client per giungere al server più il tempo impiegato dalla risposta del server per giungere al client.

Nel **RTT** vengono considerati i ritardi di propagazione del pacchetto lungo il canale, i ritardi per lo smistamento del pacchetto nei router intermedi e i ritardi di elaborazione del pacchetto. L'immagine seguente mostra cosa accade quando un utente fa clic su un collegamento ipertestuale.



Il browser invia la richiesta di "apertura connessione TCP" al server Web iniziando il **three-way handshake**:

1. il client invia un piccolo segmento TCP al server;
2. il server riconosce e risponde con un segmento TCP;
3. ed infine, il client invia la conferma al server.

Le prime due parti dell'handshake a tre vie consumano un RTT. Dopo aver completato le prime due parti dello handshake, il client invia il messaggio di richiesta HTTP combinato con la terza parte del three-way handshake sulla connessione TCP (**piggybacking**). Una volta che il messaggio di richiesta arriva al server, il server invia il file HTML sulla connessione TCP. Questa interazione di richiesta e risposta HTTP consuma un altro RTT. È quindi possibile stimare che **il tempo di risposta totale per richiedere un oggetto al server ed ottenerlo è di due RTT più il tempo di trasmissione dell'intero file HTML.**

Oltre allo overhead per stabilire e mantenere una connessione TCP per ogni oggetto di una pagina HTML, per ognuna di esse sarà inoltre necessario riservare i buffer TCP e le variabili TCP, sia sul client, sia sul server. Questo consuma risorse sul server Web che può ricevere richieste da numerosi altri client contemporaneamente. In secondo luogo ogni oggetto subisce un ritardo di consegna di due RTT: uno per stabilire la connessione TCP e uno per richiedere e ricevere un oggetto.

Connessioni Non-Persistenti e Persistenti

In molte applicazioni Internet, il client e il server comunicano per un periodo di tempo prolungato, con il client che effettua una serie di richieste e il server che risponde a ciascuna di esse. A seconda dell'applicazione e di come viene utilizzata, la serie di richieste può avvenire consecutivamente, periodicamente, ad intervalli regolari o a intermittenza. Quando questa interazione client-server si svolge **su TCP**, lo sviluppatore dell'applicazione deve **decidere** se ogni coppia "richiesta/risposta" deve viaggiare su una nuova connessione TCP, cioè le **connessioni sono non-persistenti**, oppure se tutte le richieste e le loro risposte corrispondenti devono viaggiare sulla stessa connessione TCP, secondo un approccio di **connessione persistente**.

Per comprendere questo problema di progettazione, conviene esaminare i vantaggi e gli svantaggi delle connessioni persistenti nel contesto specifico di **HTTP**, che è un **protocollo applicativo di rete che utilizza sia le connessioni persistenti sia le connessioni non persistenti**, a seconda della configurazione impostata (di default persistente).

HTTP con Connessioni Non-Persistenti

Come esempio di connessioni non persistenti si supponga di effettuare un trasferimento di una pagina Web da un server ad un client, costituita da un file HTML di base e 10 immagini in formato JPEG. Tutti gli 11 oggetti risiedono sullo stesso server. Se l'URL del file HTML di base è:

`http://www.scuola.edu/dipartimento/home.html`

I passi effettuati per il recupero degli 11 file sono i seguenti.

1. Il processo client HTTP chiede l'apertura di una connessione TCP con il server **www.scuola.edu** in ascolto sulla porta numero **80**, che è il numero di porta

predefinito per HTTP. Associato alla connessione TCP, ci sarà un socket sul client e un socket sul server.

2. Il client HTTP invia un messaggio di richiesta HTTP al server tramite il suo socket. Il messaggio di richiesta contiene il percorso **/scuola/home.html**.
3. Il processo server HTTP riceve il messaggio di richiesta attraverso il suo socket, recupera l'oggetto **/scuola/home.html** dal suo disco, incapsula l'oggetto in un messaggio di risposta HTTP, ed invia il messaggio di risposta al client tramite il suo socket.
4. Il processo server HTTP dice al TCP di chiudere la connessione che comunque attenderà la conferma di ricezione (acknowledgement) del messaggio da parte del client.
5. Il client HTTP riceve nel messaggio di risposta la pagina **home.html**, conferma la ricezione (acknowledgement) e rilascia la connessione TCP. Il client estrae il file dal messaggio di risposta, esamina il file HTML, e trova i riferimenti ai 10 oggetti JPEG.
6. I primi quattro passaggi vengono ripetuti per ciascuno dei 10 oggetti JPEG riferiti.

I passaggi descritti prima illustrano l'uso delle **connessioni non persistenti**, dove **ogni connessione TCP viene chiusa dopo che il server invia l'oggetto**, cioè la **connessione non persiste per altri oggetti**. Si noti che **ogni connessione TCP trasporta esattamente un messaggio di richiesta e un messaggio di risposta**. Quindi, in questo esempio, quando un utente richiede la pagina Web, si generano 11 connessioni TCP.

Nelle fasi sopra descritte, non si è precisato se il client ottiene le 10 immagini JPEG con 10 connessioni TCP seriali, o se alcuni dei file JPEG sono ottenuti tramite le connessioni TCP in parallelo; gli utenti possono configurare i browser moderni per controllare il grado di parallelismo. Di solito i browser aprono da 5 a 10 connessioni TCP in parallelo (variabile dall'utente), e ciascuna di queste connessioni gestisce una transazione richiesta-risposta.

L'esempio descritto evidenzia che la gestione **non persistente** delle connessioni **HTTP** utilizza **molto tempo per la gestione dello overhead per stabilire la connessione TCP** a causa dei numerosi **RTT** coinvolti. È evidente che le connessioni HTTP non persistenti sono quindi estremamente costose.

HTTP con Connessioni Persistenti

Con le connessioni persistenti, il server lascia la connessione TCP aperta dopo l'invio di una risposta. Le successive richieste e risposte tra lo stesso client e server possono essere inviate sulla stessa connessione.

In particolare, un'intera pagina Web (nell'esempio precedente, il file HTML di base e le 10 immagini) possono essere inviate attraverso una singola connessione TCP persistente, e le richieste per gli oggetti possono essere effettuate in successione, senza aspettare risposte alle richieste in sospeso (pipelining).

Tipicamente, **il server HTTP chiude la connessione quando non viene utilizzato per un certo tempo** (un intervallo di timeout configurabile).

La modalità predefinita di HTTP utilizza connessioni persistenti con pipelining, vista la maggiore efficienza di gestione delle connessioni TCP.

Formato dei messaggi HTTP

Le specifiche HTTP comprendono le definizioni dei formati dei messaggi HTTP. Ci sono **due tipi di messaggi HTTP**, i messaggi di **richiesta** e i messaggi di **risposta**.

Messaggio HTTP request

Un tipico **messaggio HTTP request** è il seguente:

```
GET /unadir/pagina.html HTTP/1.1
Host: www.scuola.edu
Connection: close
User-agent: Mozilla/5.0
Accept: text/html
Accept-language: fr
```

costituito da semplice codice ASCII.

L'esempio mostra un messaggio composto di cinque linee, ma un generico messaggio HTTP potrebbe contenere un numero maggiore o inferiore di righe rispetto a quelle mostrate.

Ogni riga è seguita da un ritorno a capo (**CR**: Carriage Return) e un avanzamento riga (**LF**: Line Feed). L'ultima riga è seguita da un ritorno a capo e un avanzamento riga supplementare.

La prima riga di un messaggio di richiesta HTTP è chiamata linea di richiesta; le righe successive sono chiamate le righe di intestazione (header).

La **linea di richiesta** è suddivisa in tre campi: il campo del **metodo** (**GET**), il campo **URL** (**/unadir/pagina.html**), e il campo **versione HTTP** (**HTTP/1.1**). Il campo metodo può assumere diversi valori, tra cui **GET**, **POST**, **HEAD**, **PUT** e **DELETE**.

Il metodo **GET** viene utilizzato quando il browser richiede un oggetto, che è specificato nel campo URL, nell'esempio il browser richiede l'oggetto **/unadir/pagina.html**, mentre il terzo campo specifica che il browser implementa la versione **HTTP/1.1**.

Dopo la riga di richiesta, nell'esempio, ci sono le righe di intestazione. La riga di intestazione **Host: www.scuola.edu** specifica l'host su cui risiede l'oggetto. Si potrebbe pensare che questa riga di intestazione non sia necessaria, in quanto vi è già una connessione TCP in atto verso l'host, ma le informazioni fornite dalla linea di intestazione host sono sfruttate dalla cache Web.

Includendo la riga di intestazione **Connection: close**, il browser sta dicendo al server che non vuole usare connessioni persistenti; vuole che il server chiuda la connessione dopo l'invio dell'oggetto richiesto; diversamente, per mantenere la connessione persistente si sarebbe dovuto includere lo header **Connection: keep-alive**.

La riga di intestazione **User-agent: Mozilla/5.0** specifica il tipo di browser che effettua la richiesta al server, nell'esempio Firefox. Questa linea di intestazione è utile perchè il server potrebbe inviare differenti versioni dello stesso oggetto ai diversi tipi di browser.

La riga di intestazione **Accept: text/html** specifica il formato dell'oggetto accettabile dal client come risposta dal server che, in questo caso, è un testo HTML.

Infine, la riga di intestazione **Accept-language: fr** indica che, se esiste sul server, l'utente preferisce ricevere una versione francese dell'oggetto, in caso contrario il server deve inviare la sua versione predefinita. L'intestazione **Accept-language:** è solo una delle intestazioni di negoziazione dei contenuti disponibili in HTTP.

Nell'immagine seguente viene fornito il formato generale di un messaggio di richiesta.



Dopo le righe di intestazione e un'altra coppia di caratteri ritorno a capo e avanzamento riga (CR-LF), c'è un "body" (corpo). Il body è vuoto se si specifica il metodo **GET**, **DELETE** o **HEAD** ma viene utilizzato se si specifica il metodo **POST** o **PUT**.

Un client HTTP usa spesso il metodo **POST** quando l'utente compila un form in quanto nel body verranno posti proprio i valori del form. Con un messaggio **POST** l'utente sta ancora chiedendo una pagina Web al server, ma i contenuti specifici della pagina Web dipendono da ciò che l'utente ha immesso nei campi del form.

In HTML è ammesso inviare i dati al server anche con il metodo **GET**, posti di seguito all'URL di richiesta. Ad esempio, se un modulo utilizza il metodo **GET** ed ha due campi i cui valori sono "scimmie" e "banane", l'URL avrà la forma: **www.unsito.com/cercaanimale?quale=scimmie&mangia=banane**.

Il metodo **HEAD** è simile al metodo **GET**. Quando un server riceve una richiesta con il metodo **HEAD**, risponde con un messaggio HTTP ma lascia vuoto il campo body, cioè

non invia l'oggetto richiesto. Gli sviluppatori di applicazioni utilizzano il metodo **HEAD** per il debug, recuperando solo le intestazioni della risposta HTTP.

Il metodo **PUT** è usato in combinazione con la pubblicazione sul web. Esso consente all'utente di caricare un oggetto in un percorso specifico (directory) su un server Web specifico. Il metodo **PUT** è utilizzato anche dalle applicazioni che necessitano di caricare oggetti sui server Web.

Il metodo **DELETE** consente a un utente o ad un'applicazione, di eliminare un oggetto su un server Web.

Messaggi HTTP Response

Il seguente è un esempio di **messaggio HTTP response**:

```
HTTP/1.1 200 OK
Connection: close
Date: Tue, 09 Aug 2017 15:44:04 GMT
Server: Apache/2.2.3 (CentOS)
Last-Modified: Tue, 15 Aug 2015 10:11:03 GMT
Content-Length: 6821
Content-Type: text/html
(... dati ...)
```

Questa risposta ha tre sezioni: una **riga di stato** (**HTTP/1.1 200 OK**), sei righe di **intestazione**, e poi il **corpo del messaggio** di risposta. **Il corpo del messaggio è l'oggetto richiesto**, rappresentato nell'esempio da (... dati ...).

La riga di stato ha tre campi: il **campo di versione del protocollo** (**HTTP/1.1**), un **codice di stato** (**200**) e il corrispondente **messaggio di stato** (**OK**). In questo esempio, la riga di stato indica che il server utilizza **HTTP/1.1** e che tutto è **OK**, cioè, il server ha trovato e sta inviando l'oggetto richiesto.

Il server utilizza la riga di intestazione **Connection: close** per dire al client che chiuderà la connessione TCP dopo l'invio del messaggio di risposta.

La linea di intestazione **Data: Tue, 09 Aug 2011 15:44:04 GMT** indica l'ora e la data in cui la risposta HTTP è stata creata e inviata dal server. Si noti che questo non è il momento in cui l'oggetto è stato creato o modificato l'ultima volta, ma è il momento in cui il server recupera l'oggetto dal suo file system, inserisce l'oggetto nel messaggio di risposta e invia il messaggio di risposta.

La riga di intestazione **Server: Apache/2.2.3 (CentOS)** indica che il messaggio è stato generato da un Web server Apache ed è analoga alla riga di intestazione **User-Agent:** nel messaggio di richiesta HTTP.

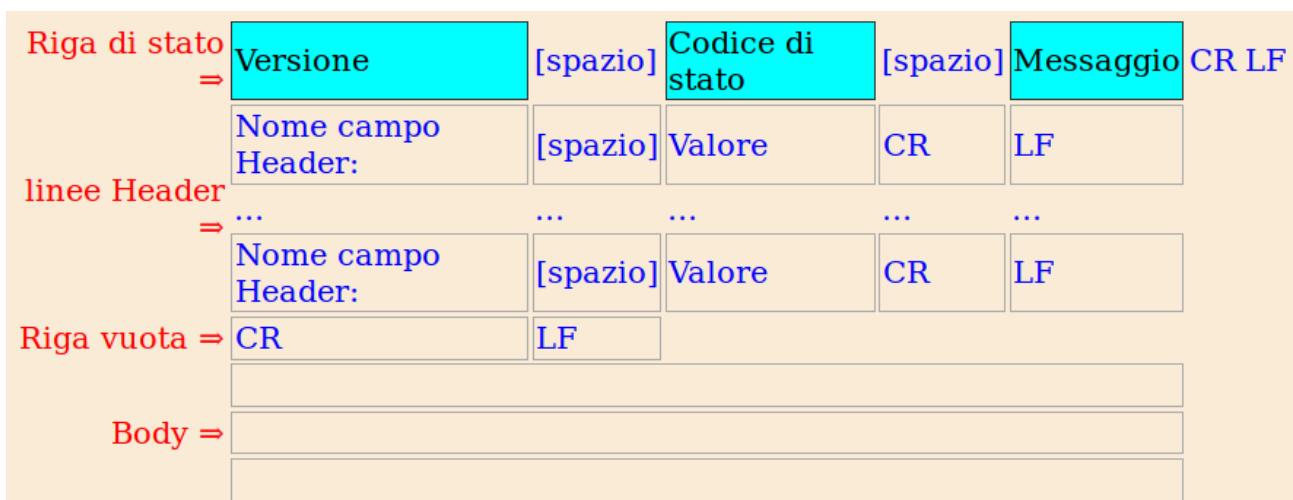
La riga di intestazione **Last-Modified: Tue, 15 Aug 2015 10:11:03 GMT** indica l'ora e la data in cui l'oggetto è stato creato o modificato l'ultima volta. L'intestazione

Last-Modified: è fondamentale per la memorizzazione nella cache, sia nel client locale sia nella cache di rete (nota anche come server proxy).

La riga di intestazione **Content-Length:** 6821 indica il numero di byte dell'oggetto inviato nel body.

La riga di intestazione **Content-Type:** `text/html` indica il formato dell'oggetto inviato nel body che, in questo caso, è un testo HTML. Si osservi che **il tipo dell'oggetto, e quindi l'applicazione che verrà avviata automaticamente per gestirlo, è specificato dal Content-Type: e non dall'estensione del file.**

Di seguito viene fornito il formato generale di un messaggio di risposta. Il codice di stato e la descrizione associata indicano l'esito della richiesta.



Alcuni codici di stato sono:

- **200 OK**: la richiesta si è risolta con successo e quindi nella risposta c'è l'informazione.
- **301 Moved Permanently**: l'oggetto richiesto è stato spostato in modo permanente; il nuovo URL è specificato nello header **Location**: del messaggio di risposta. Il client recupererà automaticamente il nuovo URL.
- **400 Bad Request**: questo è un codice di errore generico che indica che la richiesta non può essere compresa dal server.
- **404 Not Found**: il documento richiesto non esiste su questo server.
- **505 HTTP Version Not Supported**: la versione del protocollo HTTP richiesta non è supportata dal server.

Per vedere un vero e proprio messaggio di risposta HTTP si può provare ad avviare il programma Telnet sulla shell dei comandi, collegandosi ad un server Web digitando un messaggio di richiesta. Ad esempio, se si ha accesso a un prompt dei comandi digitare:

```
telnet cis.poly.edu 80
GET /~ross/ HTTP/1.1
Host: cis.poly.edu
```

Si prema due volte il tasto "Invio" dopo aver digitato l'ultima riga.

Si aprirà una connessione TCP sulla porta 80 dell'host **cis.poly.edu** e poi si invierà il messaggio di richiesta HTTP. Si dovrebbe vedere un messaggio di risposta che include il file HTML della home page del professor Ross (*il Prof. Ross non ha più la pagina presso l'università indicata, verrà probabilmente visualizzata una pagina di segnalazione 301 Moved permanently*). Se si preferisce vedere solo le linee del messaggio HTTP e non ricevere l'oggetto, sostituire **GET** con **HEAD**.

In questa sezione sono state descritte alcune righe di intestazione che possono trovarsi all'interno di una richiesta e di una risposta HTTP. Le specifiche HTTP definiscono molte altre righe di intestazione che possono essere inserite dai browser, dai server Web o dai server di cache della rete.

Un browser genererà linee di intestazione in funzione del tipo e della versione del browser (ad esempio, un browser **HTTP/1.0** non genererà alcuna linea di intestazione), della configurazione del browser dell'utente (ad esempio, la lingua preferita) e se il browser ha nella sua cache la versione non aggiornata dell'oggetto. I Server Web si comportano in modo simile, ma a seconda del prodotto, della versione e delle configurazioni, cambieranno le intestazioni incluse nei messaggi di risposta.

File Transfer Protocol: FTP

Libro vol.3 - Il livello delle applicazioni

STUDIARE: L. Lo Russo, E. Bianchi, Sistemi e Reti - Nuova Edizione OPENSCHOOL - Per l'articolazione INFORMATICA degli Istituti Tecnici settore Tecnologico, vol. 3, ed. Hoepli, 2017

- Il Web:HTTP e FTP pp.17-19

Funzionamento generale

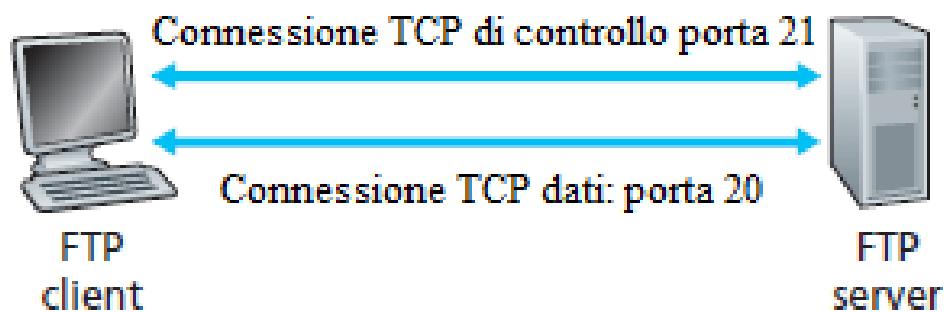
Il protocollo applicativo di rete **FTP** (*File Transfer Protocol*) è utilizzabile per il **trasferimento di file** da o verso un host remoto.

Un utente interagisce con FTP attraverso un **User-Agent FTP**, fornendo il **nome dell'host remoto**, un **nome di identificazione** e una **password**.

Il nome dell'host remoto è necessario affinché il processo FTP client nell'host locale apra una connessione TCP con il processo FTP server nell'host remoto.

Il nome di identificazione e la password saranno inviati attraverso la connessione TCP come parte di comandi FTP.

Una volta che il server avrà autorizzato l'utente, si potrà iniziare la copia di uno o più file memorizzati in uno dei due file system, verso il file system paritario.



HTTP e FTP sono due protocolli di trasferimento file che si appoggiano su TCP. Tuttavia, i due protocolli a livello di applicazione hanno alcune differenze importanti. La differenza più evidente è che **FTP utilizza due connessioni TCP in parallelo** per trasferire un file, **una connessione di controllo e una connessione dati**. La **connessione di controllo viene utilizzata per l'invio di informazioni di controllo tra i due host**, come l'identificativo utente, la password, i comandi per cambiare directory remota, e i comandi **PUT** e **GET** per trasferire i file. La **connessione dati viene utilizzata per inviare effettivamente un file**.

Poiché FTP utilizza una connessione di controllo separata, si dice che **FTP invia le informazioni di controllo fuori banda**. HTTP, come si ricorderà, manda pacchetti di richiesta e di risposta sulla stessa connessione TCP sulla quale trasferisce il file. Per questo motivo, si dice che **HTTP invia le informazioni di controllo in banda**.

Quando un utente avvia una sessione FTP con un host remoto, il lato client FTP prima apre una **connessione TCP di controllo** con il lato server sulla **porta numero 21 del server**. La parte client FTP invia l'identificativo dell'utente e la password su questa connessione di controllo. Il lato client FTP invia anche, tramite la connessione di controllo, i comandi per modificare la directory remota. Quando un client intende trasferire dei dati verso il server dovrà aprire un **canale** di tipo **attivo** selezionando una porta casuale, inviandone il valore al server tramite il canale comandi (*porta 21*) e attendendo che questo si connetta, facendo il binding alla **porta 20, dedicata al trasferimento dati da client a server**.

Quando invece si richiede un trasferimento di dati dal server al client, il **canale** dati è di tipo **passivo** e il server aprirà una porta solitamente casuale (superiore alla 1023); tramite il canale comandi invierà al client il numero della porta selezionata, attendendo che quest'ultimo si connetta.

FTP invia esattamente un file sulla connessione dati e quindi chiude la connessione. Se durante la stessa sessione l'utente volesse trasferire un altro file, FTP aprirà un'altra connessione dati. La **connessione di controllo resterà sempre aperta per tutta la durata della sessione**, ma la connessione per il trasferimento dei dati verrà creata per ogni file che dovrà essere trasferito. Da ciò si deduce che la **connessione di controllo è persistente**, mentre la **connessione dati non è persistente**.

Nel corso di una sessione, il server **FTP deve mantenere lo stato**, a differenza di HTTP che invece è stateless. In particolare, il server deve associare la connessione di controllo con un account utente specifico, e il server deve tenere traccia della directory corrente dell'utente mentre questo si sposta nel file system del server. Tenere traccia delle informazioni di stato per ogni sessione utente condiziona significativamente il numero totale di sessioni che FTP è in grado di mantenere contemporaneamente.

Comandi e Risposte FTP

Di seguito verranno illustrati alcuni dei comandi FTP più comuni, con le risposte relative all'esito del comando. I comandi, dal client al server, e le risposte, dal server al client, vengono inviate in chiaro sulla connessione di controllo in formato ASCII a 7 bit. Per delimitare i comandi, ognuno viene terminato con la coppia di caratteri ritorno a capo e avanzamento riga. Ogni comando è composto da caratteri ASCII, alcuni con argomenti opzionali. Alcuni dei comandi più comuni sono:

- **open addr**: usato per aprire una connessione remota.
- **quit**: usato per chiudere la sessione FTP corrente.
- **ls**: usato per chiedere al server di ottenere l'elenco dei file contenuti nella directory remota corrente. L'elenco dei file viene inviato su una (nuova e non-persistente) connessione dati.

- **get filename**: usato per ottenere (cioè scaricare) un file dalla directory corrente dello host remoto. Dopo questo comando l'host remoto apre una connessione e, su questa, invia il file.
- **put filename**: usato per memorizzare (upload) un file nella directory corrente dell'host remoto.

Vi è una corrispondenza uno a uno tra il comando che l'utente invia e il comando FTP inviato sulla connessione di controllo. Ciascun comando è seguito da una risposta inviata dal server al client. Le risposte sono numeri di tre cifre, con un messaggio opzionale dopo il numero. Questo è simile, nella struttura, al codice di stato con la frase nella riga di stato del messaggio di risposta HTTP.

Alcune possibili risposte con i loro messaggi sono le seguenti:

- 331 Username OK, password required
- 125 Data connection already open; transfer starting
- 425 Can't open data connection
- 452 Error writing file

Di seguito è riportato un esempio di sessione FTP a linea di comando:

```
mgm@umgm:~$ ftp
ftp> open 192.168.64.50 2221
Connected to 192.168.64.50.
220 Service ready for new user.
Name (192.168.64.50:mgm): utenteftp
331 User name okay, need password for utenteftp.
Password:
230 User logged in, proceed.
Remote system type is UNIX.
ftp> ls
200 Command PORT okay.
150 File status okay; about to open data connection.
drwx----- 3 user group          0 Aug 21  2015
.Ds2mfefxaxSR-fhfVO_OLvQBMkI=
drwx----- 3 user group          0 Jul 12 13:08
._kcTr_7DXOfhoqZuOwz6sJV7vjs=
drwx----- 3 user group          0 Oct 30  2016 .facebook_cache
drwx----- 3 user group          0 Jul 17 16:34 .mobvista700
drwx----- 3 user group          0 Mar 18  2015 .norton
drwx----- 3 user group          0 Mar 17  2015 .symantec_persisted
drwx----- 3 user group          0 Jan  2  1970 Alarms
drwx----- 3 user group          0 Sep 10  2015 Android
drwx----- 3 user group          0 Jun 26  2016 AnkiDroid
drwx----- 3 user group          0 Mar 14  2015 BDArenaConnector
drwx----- 3 user group          0 Mar 21  2015 DCIM
drwx----- 3 user group          0 Aug   6 21:28 Download
drwx----- 3 user group          0 Nov 12  2015 Fonts
drwx----- 3 user group          0 Oct 31  2015 Hyperionics
drwx----- 3 user group          0 Mar 25  2015 Movies
drwx----- 3 user group          0 Oct 22  2015 Music
drwx----- 3 user group          0 Jun 28  2016 Notifications
drwx----- 3 user group          0 Apr 11  2015 Pictures
```

```
drwx----- 3 user group          0 Jan 31 2016 Podcasts
drwx----- 3 user group          0 Jan  2 1970 Ringtones
drwx----- 3 user group          0 Dec  3 2016 Sonic Player
Recordings
drwx----- 3 user group          0 Oct  5 2015 SoundRecorder
drwx----- 3 user group          0 May 24 2016 Telegram
drwx----- 3 user group          0 May 28 2016 WhatsApp
drwx----- 3 user group          0 Aug  6 21:23 Zipper
drwx----- 3 user group          0 Jan 30 2016 aquerry
drwx----- 3 user group          0 Feb 15 2015 beam
drwx----- 3 user group          0 Oct 18 2015 com.facebook.katana
drwx----- 3 user group          0 Apr  3 2015 com.facebook.orca
drwx----- 3 user group          0 Apr 11 2016 data
drwx----- 3 user group          0 Jun 22 16:57 documents
drwx----- 3 user group          0 Feb 14 2015 media
drwx----- 3 user group          0 Feb 17 2015 storage
drwx----- 3 user group          0 Mar 17 2015
symantec_update_temp
drwx----- 3 user group          0 Apr 21 2016 viber
-rw----- 1 user group          33 Feb 25 2015 .bugsense
226 Closing data connection.
ftp> quit
221 Goodbye.
mgm@umgm:~$
```

Domain Name System: DNS

Libro vol.3 - Il livello delle applicazioni

STUDIARE: L. Lo Russo, E. Bianchi, Sistemi e Reti - Nuova Edizione OPENSCHOOL - Per l'articolazione INFORMATICA degli Istituti Tecnici settore Tecnologico, vol. 3, ed. Hoepli, 2017

- Email, DNS e Telnet pp.25-28

DNS - Il servizio Directory di Internet

Esistono molti modi per **identificare un host**, e uno di questi è **il nome** dell'host. Nomi host come `cnn.com`, `www.yahoo.com`, `frapec.edu` e `cis.poly.edu` sono mnemonici e pertanto sono più semplici da ricordare per gli esseri umani. Tuttavia **i nomi di host forniscono poche informazioni sulla posizione dell'host all'interno di Internet**. Un hostname come `www.euro.fr`, che termina con il codice del paese `fr`, dice che l'host è probabilmente in Francia, ma non fornisce altre informazioni. Inoltre, poiché i nomi degli host possono essere costituiti da caratteri alfanumerici di lunghezza variabile, sarebbero difficili da trattare dai router. Per queste ragioni, **gli host sono identificati dagli indirizzi IP**.

Un indirizzo IPv4 consiste di quattro byte ed ha una struttura gerarchica rigida. Un indirizzo IPv4 assomiglia a `121.7.106.83`, dove ogni punto separa uno dei byte espressi in notazione decimale da 0 a 255. L'indirizzo IP è gerarchico, perché quando un router lo analizza da sinistra a destra utilizzando la netmask, ottiene informazioni sempre più specifiche su dove si trova l'host in Internet, cioè la rete di appartenenza.

Servizi forniti dal DNS

Un **host** può essere **identificato** in due modi distinti, tramite un **nome** e attraverso il suo **indirizzo IP**.

Un **operatore umano** preferirà l'identificatore di host più mnemonico, cioè il **nome**, mentre i **router** preferiscono gli **indirizzi IP di lunghezza fissa e gerarchicamente strutturati**. Per conciliare queste preferenze, serve un **servizio di directory** che **traduca i nomi host in indirizzi IP**. Questo è il **compito principale del** sistema dei nomi di dominio di Internet, o **Domain Name System (DNS)**.

Il **DNS è un database distribuito** implementato in una gerarchia di server DNS, ma **è anche un protocollo del livello di applicazione** che consente agli host di interrogare il database distribuito per **risolvere il nome di un dominio, ottenendo il corrispondente indirizzo IP**. Il **protocollo DNS è eseguito su UDP e utilizza la porta 53**.

DNS è comunemente impiegato da altri protocolli del livello applicazione, tra cui HTTP (*HyperText Transfer Protocol*), SMTP (*Simple Mail Transfer Protocol*) e FTP (*File Transfer Protocol*), con il compito di tradurre i nomi degli host forniti dall'utente in indirizzi IP. Come esempio, si consideri cosa succede quando un browser (cioè, un client HTTP), in esecuzione sull'host di qualche utente, richiede la risorsa index.html presso il dominio www.scuola.edu/, usando l'URL www.scuola.edu/index.html. Affinché l'host dell'utente sia in grado di inviare un messaggio di richiesta HTTP al server Web, deve prima ottenere l'indirizzo IP di www.scuola.edu. Questa operazione di **risoluzione del nome di un dominio** viene svolta considerando i seguenti punti:

1. sulla macchina utente viene gestito il lato client dell'applicazione DNS;
2. il browser estrae il nome host, www.scuola.edu, dall'URL e passa il nome host al lato client dell'applicazione DNS;
3. il client DNS invia al server DNS una interrogazione contenente l'hostname www.scuola.edu;
4. il client DNS riceve una risposta contenente l'indirizzo IP corrispondente all'hostname;
5. una volta che il browser riceve l'indirizzo IP dal DNS, può aprire una connessione TCP con il processo server HTTP in ascolto sulla porta 80 a quell'indirizzo IP.

Da questo esempio si vede che il **DNS aggiunge un ulteriore ritardo per le applicazioni Internet** che lo utilizzano. Fortunatamente, come si vedrà di seguito, **l'indirizzo IP desiderato è spesso memorizzato nella cache in un server DNS "vicino"**, che aiuta a ridurre il traffico di rete e il ritardo medio di risposta del DNS.

DNS fornisce alcuni altri importanti servizi, oltre a tradurre nomi di host in indirizzi IP (risolvere il nome di un dominio):

- **Host aliasing:** un host con un nome complicato può avere uno o più nomi alternativi. Ad esempio, un hostname come relay1.west-coast.enterprise.com potrebbe avere, ad esempio, due alias come enterprise.com e www.enterprise.com. In questo caso, il nome del computer relay1.westcoast.enterprise.com si dice che è un **hostname canonico**. I nomi di host alternativi, quando presenti, sono in genere più mnemonici dei nomi host canonici. Il servizio DNS può essere richiamato da una applicazione per ottenere l'hostname canonico per un nome alternativo fornito, così come l'indirizzo IP dell'host.
- **Mail server aliasing:** per ovvie ragioni, è altamente auspicabile che gli indirizzi di posta elettronica siano mnemonici. Ad esempio, se Bob ha un account con Hotmail, l'indirizzo e-mail di Bob potrebbe essere semplice bob@hotmail.com. Tuttavia, il nome host del server di posta elettronica Hotmail è più complicato e meno mnemonico rispetto a hotmail.com (per esempio, il nome host canonico potrebbe essere qualcosa di simile a relay1.west-coast.hotmail.com). Il DNS può essere richiamato da una applicazione di posta elettronica per ottenere l'hostname canonico per un hostname alternativo fornito, così come l'indirizzo IP dell'host. Infatti, il record MX (vedi sotto) permette al server e-mail e ai server Web di una società di

avere hostname identici (alias); per esempio, il server Web e il server di posta di una società possono essere chiamati entrambi enterprise.com.

- **Load distribution:** il DNS è utilizzato anche per distribuire il carico tra i server replicati, come accade spesso per i server Web. I siti con un numero molto elevato di accessi, come ad esempio cnn.com, vengono replicati su più server, con ogni server in esecuzione su un sistema terminale diverso e ciascuno con un indirizzo IP diverso. **Per i server Web replicati un insieme di indirizzi IP viene associato ad un nome host canonico. Il database DNS contiene questa serie di indirizzi IP.** Quando i client interrogano il **DNS** per un nome corrispondente ad un insieme di indirizzi, il server risponde con l'intero insieme di indirizzi IP, ma **ruota l'ordine degli indirizzi all'interno di ciascuna risposta.** Poiché un client invia tipicamente il suo messaggio di richiesta HTTP all'indirizzo IP che viene nominato per primo nell'insieme, **la rotazione DNS distribuisce il traffico tra i server replicati.** La rotazione del DNS è utilizzata anche per la posta elettronica in modo che più server di posta elettronica possano avere lo stesso nome alternativo.

DNS: funzioni di rete critiche con il paradigma client-server

Come HTTP, FTP e SMTP, il protocollo **DNS è un protocollo del livello applicazione** perché viene eseguito per far comunicare sistemi terminali utilizzando il **paradigma client-server**, basandosi su un protocollo di trasporto end-to-end per il trasferimento di messaggi DNS tra i sistemi terminali comunicanti. Tuttavia, il ruolo del DNS è molto diverso dalle applicazioni Web (HTTP), dal trasferimento di file (FTP), e dalle e-mail (SMTP). A differenza di queste applicazioni, il **DNS non è un'applicazione con cui un utente interagisce direttamente.** Il **DNS fornisce una funzione Internet di base traducendo i nomi degli host nei relativi indirizzi IP**, per poterli utilizzare nelle applicazioni utente e altro software in Internet.

Gran parte della complessità dell'architettura Internet si trova ai "bordi" della rete. Il DNS, che implementa il processo di risoluzione del nome di un dominio (traduzione nome-indirizzo), utilizzando un'architettura client-server, si trova ai margini della rete confermando questa filosofia di design.

Panoramica del DNS⁴³

Quando una applicazione client, come un browser Web o un lettore di posta, in esecuzione su un computer di un utente necessita di **risolvere** un nome host in un indirizzo IP, deve necessariamente avviare il lato client del DNS, specificando il nome host che deve essere tradotto. Ad esempio, su molte macchine basate su UNIX l'applicazione client richiama la funzione `gethostbyname()`. Il DNS client presente sullo host dell'utente si assume quindi l'incarico di inviare un messaggio di interrogazione in rete, utilizzando la porta 53 e il protocollo di trasporto UDP.

⁴³ [How a DNS Server \(Domain Name System\) works.](#)

Dopo un ritardo, che varia da pochi millisecondi a qualche secondo, il DNS client sullo host dell'utente riceve un messaggio di risposta che fornisce la corrispondenza desiderata. Questa corrispondenza viene infine passata all'applicazione che ne necessitava e che aveva invocato il client DNS.

Dal punto di vista dell'applicazione chiamante sull'host dell'utente, il DNS è un *black-box* che fornisce un servizio di risoluzione utilizzando un gran numero di server DNS distribuiti in tutto il mondo, e avvalendosi di un protocollo del livello di applicazione che specifica come comunicano i server DNS e i client DNS sugli host che emettono l'interrogazione. L'applicazione chiamante ovviamente ignora completamente questo livello di complessità, utilizzando il servizio DNS all'occorrenza.

Affinché il servizio **DNS** sia in grado di svolgere effettivamente il suo compito di risoluzione **deve necessariamente basarsi su un database distribuito su più server.**

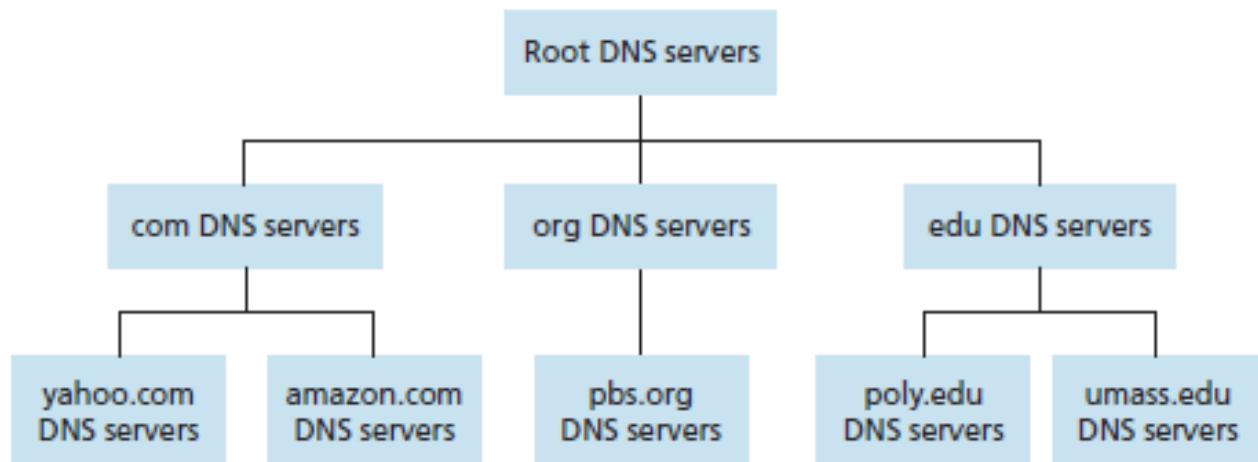
Un Database Gerarchico Distribuito

Per affrontare il problema dell'adattamento alle variazioni della rete il DNS utilizza un gran numero di server, organizzati in modo gerarchico e distribuito in tutto il mondo. In questo modo nessun server DNS ha tutte le corrispondenze per tutti gli host in Internet, distribuendo tra i diversi server DNS le associazioni nome-indirizzo.

In una prima approssimazione, i server DNS si suddividono in tre classi e ogni classe è preposta alla risoluzione di una porzione del nome di dominio, separate da un punto, considerando che ogni nome completo è terminato da un . che identifica proprio il dominio radice:

- **Server DNS root** (*Root DNS server*) - In Internet ci sono 13 server DNS root, etichettati da A a M, la maggior parte dei quali si trovano in Nord America. Un elenco degli attuali server DNS è disponibile tramite la [IANA](#) o tramite [Root Server Technical Operations Assn](#). Anche se ciascuno dei 13 server radice DNS viene immaginato come se fosse un unico server, ogni "server" è in realtà una rete di server replicati, per motivi di sicurezza e affidabilità. In totale, ci sono 247 server principali a partire dal 2011. Sono i server che ricevono per primi la query del client DNS e forniscono l'indirizzo IP del server DNS di primo livello da interrogare.
- **Server DNS di dominio di primo livello** (*TLD server - Top-Level Domain server*) - Questi server sono responsabili per i domini di primo livello come *.com*, *.org*, *.net*, *.edu*, *.gov* e tutti i domini di primo livello dei nomi dei paesi, quali *.uk*, *.fr*, *.ca*, *.jp*, *.it*. La società [Verisign Global Registry Services](#) mantiene i server TLD per il dominio *.com* di primo livello, e la società [Educause](#) mantiene i server TLD per il dominio di primo livello *.edu*. Presso il sito della [IANA](#) è possibile vedere un elenco di tutti i domini di primo livello. Questi server restituiscono l'indirizzo IP del server autorevole in grado di risolvere la query del client DNS.
- **Server DNS autorevole di dominio di secondo livello** (*Authoritative DNS server*) - Ogni organizzazione con dei server accessibili al pubblico (ad esempio un server Web o un server di posta) tramite Internet deve fornire i **record DNS**

accessibili pubblicamente che mappano i nomi di tali host in indirizzi IP. Alcuni **server DNS di fiducia di un'organizzazione ospitano questi record DNS**. Un'organizzazione può scegliere di eseguire il proprio server DNS di fiducia per tenere questi record o, in alternativa l'organizzazione può pagare per avere questi record archiviati in un server DNS di qualche fornitore di servizi. La maggior parte delle università e grandi aziende implementano e mantengono i propri server DNS autorevoli primario e secondario (quest'ultimo di backup). Questi server risolvono il dominio di secondo livello (*yahoo.com* del dominio di primo livello *.com*) e quindi forniscono la risposta definitiva di risoluzione del dominio al client DNS.



Per capire come queste tre classi di server interagiscono, si supponga che un client DNS voglia determinare l'indirizzo IP del nome host *www.amazon.com*. In prima approssimazione, si svolgeranno i seguenti eventi:

- il client DNS contatta uno dei DNS root server, che restituisce gli indirizzi IP dei server TLD per il dominio di primo livello *.com*;
- il client DNS contatta quindi uno di questi server TLD che restituirà l'indirizzo IP di un server autorevole per la risoluzione del dominio *amazon.com*;
- infine il client DNS contatta uno dei server autorevoli per *amazon.com* che restituirà l'indirizzo IP per il nome host www.amazon.com.

Il root DNS server, il TLD server e il server DNS autorevole appartengono tutti alla gerarchia di server DNS, ma esiste un altro importante tipo di server DNS denominato **server DNS locale**, che pur non appartenendo alla gerarchia dei server DNS, è comunque **fondamentale per l'architettura DNS**. Ogni ISP - come un'università, un dipartimento accademico, aziendale o un ISP residenziale dispone di un **server DNS locale**. Quando un host si connette a un ISP, questo fornisce gli indirizzi IP di uno o più dei suoi server DNS locali, tipicamente tramite DHCP. Il **server DNS locale di un host è tipicamente vicino allo host**. Ad esempio, per un ISP istituzionale il server DNS locale può essere sulla stessa LAN dell'host, mentre per un ISP residenziale sarà separato dall'host da qualche router.

Quando un host effettua una query DNS, la query viene inviata prima di tutto al server DNS locale, che ricopre anche il ruolo di proxy. Se il server DNS locale non è in grado

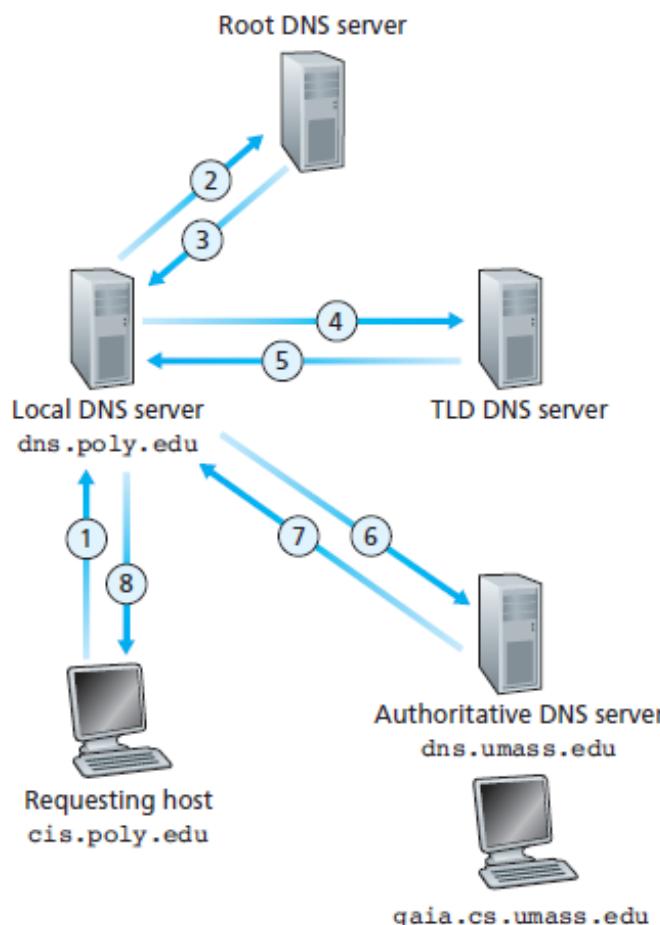
di risolvere il nome di dominio, non trovando un record nella sua cache per risoluzioni precedenti, allora inoltra la query alla gerarchia di server DNS.

Interrogazioni iterative e ricorsive

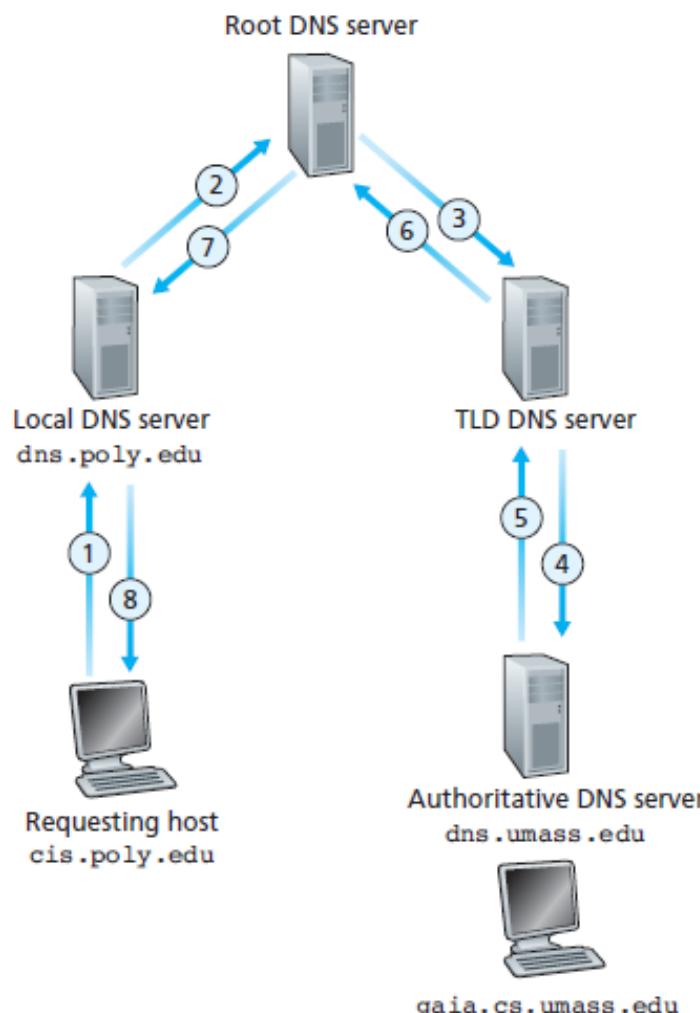
Le interrogazioni poste alla gerarchia dei server DNS possono essere risolte utilizzando due metodi:

- **iterativo**: lo host si occupa di interrogare direttamente ogni singolo livello della gerarchia dopo aver ricevuto l'indirizzo IP del livello da interrogare;
- **ricorsivo**: ogni server della gerarchia si occupa di portare avanti l'interrogazione per conto dello host.

Nella realtà **viene utilizzato un metodo ibrido**, utilizzando **il metodo ricorsivo tra lo host e il server DNS locale**, e avvalendosi del **metodo iterativo per tutti i server della gerarchia DNS**.

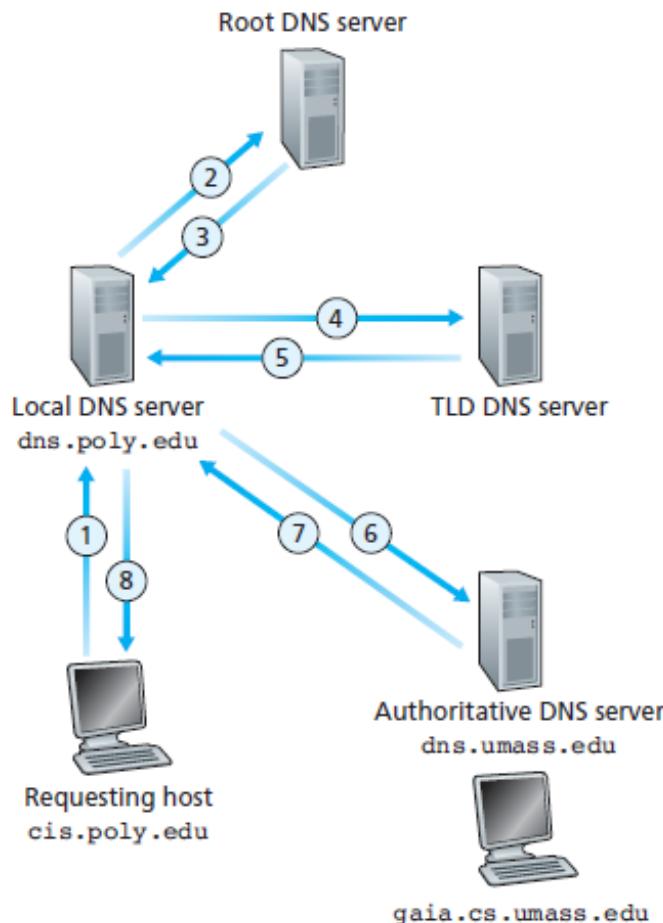


Nell'immagine seguente viene invece mostrato, per completezza, un esempio di interrogazione ricorsiva.



DNS Caching

Finora si è ignorata la presenza della **cache DNS**, una **caratteristica estremamente importante del sistema DNS**. In verità, il DNS sfrutta ampiamente il caching per migliorare le prestazioni in termini di ritardo, e allo scopo di ridurre il numero di messaggi DNS scambiati su Internet. L'idea alla base del caching DNS è molto semplice; in una catena di query, quando un server DNS riceve una risposta DNS contenente, ad esempio, una corrispondenza fra un nome host e un indirizzo IP, può memorizzare nella cache locale la corrispondenza. Per esempio, nell'esempio mostrato in figura, ogni volta che il server DNS locale dns.poly.edu riceve una risposta da qualche server DNS, può memorizzare nella cache qualsiasi informazione in essa contenuta.



Se una coppia nome dominio/indirizzo IP viene memorizzata nella cache di un server DNS locale, tutte le successive richieste per la risoluzione del medesimo dominio potranno essere soddisfatte direttamente dal server DNS locale, anche se non è autorevole per il nome di dominio. Chiaramente, dopo un certo tempo queste associazioni non autorevoli dovranno essere eliminate e sostituite in quanto le associazioni tra nomi host e indirizzi IP non sono permanenti. **Il caching delle associazioni nome dominio/indirizzi IP permette di velocizzare la risoluzione di un nome di dominio.**

Si consideri che un server DNS locale può anche memorizzare nella cache gli indirizzi IP dei server TLD, consentendo in tal modo al server DNS locale di saltare un livello nella sequenza delle interrogazioni.

Record DNS

I server DNS che insieme implementano il database DNS distribuito, memorizzano **record di risorse** (*RR - Resource Record*), compresi i record che forniscono la corrispondenza tra nome di dominio a indirizzo IP. Ogni messaggio di risposta DNS trasporta uno o più record di risorse.

Un **record di risorsa** è un tupla che contiene i seguenti campi:

(Nome, Valore, Tipo, TTL)

dove **TTL** è il **tempo di vita del record di risorse** e determina quando una risorsa deve essere rimossa da una cache. Negli esempi seguenti questo campo verrà tralasciato per chiarezza espositiva.

Il significato dei campi **Nome** e del campo **Valore** dipende dal campo **Tipo**. Di seguito vengono elencati alcuni [tipi di record DNS](#).

- Se **Tipo = A**, allora **Nome** è un nome host e **valore** è l'indirizzo IP di quello host. Così, un **record di Tipo A** fornisce la corrispondenza standard hostname-IP. A titolo di esempio, (`relay1.bar.foo.com`, `145.37.93.126`, `A`) è un record di tipo A.
- Se **Tipo = NS**, allora **Nome** è un dominio (ad esempio `foo.com`) e **valore** è l'hostname di un server DNS autorevole che sa come ottenere gli indirizzi IP per gli host nel dominio. Questo record è utilizzato per inoltrare le query DNS nella catena di query. Ad esempio, (`foo.com`, `dns.foo.com`, `NS`) è un **record di tipo NS**.
- Se **Tipo = CNAME**, allora il **Valore** è un **nome host canonico per il nome alternativo (alias) dello hostname**. Questo record può fornire agli host il nome canonico per un ottenere l'hostname. A titolo di esempio, (`foo.com`, `relay1.bar.foo.com`, `CNAME`) è un **record di tipo CNAME**.
- Se **Tipo = MX**, allora il **valore** è il **nome canonico di un server di posta che ha come alias Nome**. A titolo di esempio, (`foo.com`, `mail.bar.foo.com`, `MX`) è un **record di Tipo MX**. I Record MX consentono ai nomi host del server di posta elettronica di avere alias semplici. Si noti che utilizzando il record MX, una società può avere lo stesso nome alias per il server di posta e per uno dei suoi altri server (come ad esempio il proprio server Web). Per ottenere il nome canonico per il server di posta elettronica, un client DNS potrebbe interrogare per un record MX; per ottenere il nome canonico per l'altro server, il client DNS interroga per il record CNAME.

Utilizzando a linea di comando il comando `nslookup` è possibile inviare una query DNS a qualsiasi server DNS (root, TLD, autorevole o non autorevole). Dopo aver ricevuto il messaggio di risposta dal server DNS, `nslookup` visualizzerà i record inclusi nella risposta, in un formato leggibile. In ambiente Linux è disponibile anche il comando `dig`, più recente rispetto al comando `nslookup`.

Vulnerabilità DNS

Il DNS è un componente fondamentale dell'infrastruttura di Internet, con molti servizi importanti, tra cui il Web e la posta elettronica, incapaci di funzionare senza un servizio di risoluzione dei nomi di dominio. Per questa ragione il servizio DNS è spesso soggetto ad attacchi di diversa natura.

Il primo tipo di attacco che viene in mente è un **attacco DDoS (Distributed Denial of Service)** di **intasamento della banda contro i server DNS**. Ad esempio, un utente malintenzionato può tentare di inviare ad ogni server DNS principale una gran quantità di pacchetti, così tanti che la maggior parte delle query DNS legittime non otterrà mai risposta. Questo attacco DDoS contro server

radice DNS ha effettivamente avuto luogo il 21 Ottobre 2002⁴⁴. In questo attacco, gli aggressori inviarono ininterrottamente messaggi ping ICMP a ciascuno dei 13 root server DNS. Fortunatamente, questo attacco su larga scala ha causato danni minimi, avendo poco o nessun impatto sull'utilizzo della rete da parte degli utenti. Gli aggressori riuscirono a dirigere un flusso enorme di pacchetti ai server principali, ma molti dei root DNS server erano protetti da filtri di pacchetti, configurati per bloccare sempre tutti i messaggi ping ICMP diretti ai server principali. Questi server protetti furono così risparmiati e funzionarono normalmente. Inoltre, la maggior parte dei server DNS conteneva nella cache locale gli indirizzi IP dei server di dominio di primo livello, permettendo al processo di query di evitare spesso i root DNS server.

Un **attacco DDoS** potenzialmente più efficace contro il DNS sarebbe quello di **inviare un flusso di query DNS ai server di dominio di primo livello**, per esempio a tutti i server che gestiscono il dominio .com, in quanto risulta più difficile filtrare delle query legittime inviate al server DNS, e i server di primo livello non sono così facilmente agirabili come lo sono i server root. La gravità di un tale attacco sarebbe parzialmente mitigata dalla memorizzazione nella cache dei server DNS locali.

Il DNS potrebbe potenzialmente essere attaccato in altri modi. In un **attacco man-in-the-middle, l'attaccante intercetta le richieste da un host e restituisce le risposte false**.

In un attacco **DNS poisoning, l'attaccante invia risposte false a un server DNS, ingannando il server ad accettare i record falsi nella sua cache**. Uno di questi attacchi potrebbe essere utilizzato, ad esempio, per reindirizzare un utente ignaro al sito Web dell'utente malintenzionato. Questi attacchi, però, sono di difficile attuazione, in quanto richiedono di intercettare pacchetti o superare i firewall.

Un altro importante **attacco sfrutta l'infrastruttura DNS per lanciare un attacco DDoS contro un obiettivo diverso**, per esempio server di posta. In questo attacco, l'attaccante invia query DNS per molti server DNS autorevoli, mettendo in ogni query l'indirizzo sorgente contraffatto dell'host di destinazione. I server DNS quindi inviano le loro risposte direttamente all'host di destinazione. Se le query possono essere realizzate in modo tale che una risposta è molto più grande (in byte) di una query (cosiddetta amplificazione), allora l'attaccante può potenzialmente sopraffare la porta senza dover generare gran parte del proprio traffico. Tali attacchi DNS hanno avuto un successo limitato finora.

In sintesi il protocollo applicativo DNS si è dimostrato sorprendentemente robusto contro gli attacchi e, ad oggi non vi è stato un attacco che ha impedito con successo il servizio DNS. Ci sono stati attacchi efficaci, ma questi possono essere e sono affrontati con un'adeguata configurazione dei server DNS.

⁴⁴ William F. Slater III, [*The Internet Outage and Attacks of October 2002*](#), writing del 7 novembre 2002.

La posta elettronica

Libro vol.3 - Il livello delle applicazioni

STUDIARE: L. Lo Russo, E. Bianchi, Sistemi e Reti - Nuova Edizione OPENSCHOOL - Per l'articolazione INFORMATICA degli Istituti Tecnici settore Tecnologico, vol. 3, ed. Hoepli, 2017

- Il Web:HTTP e FTP pp.20-25 e pp.28-33

Introduzione

La posta elettronica è stata usata fin dalla nascita di Internet. È stata l'applicazione più popolare quando Internet era ancora agli albori, ed è diventata sempre più sofisticata e potente nel corso degli anni. Rimane una delle più importanti ed utilizzate applicazioni di Internet.

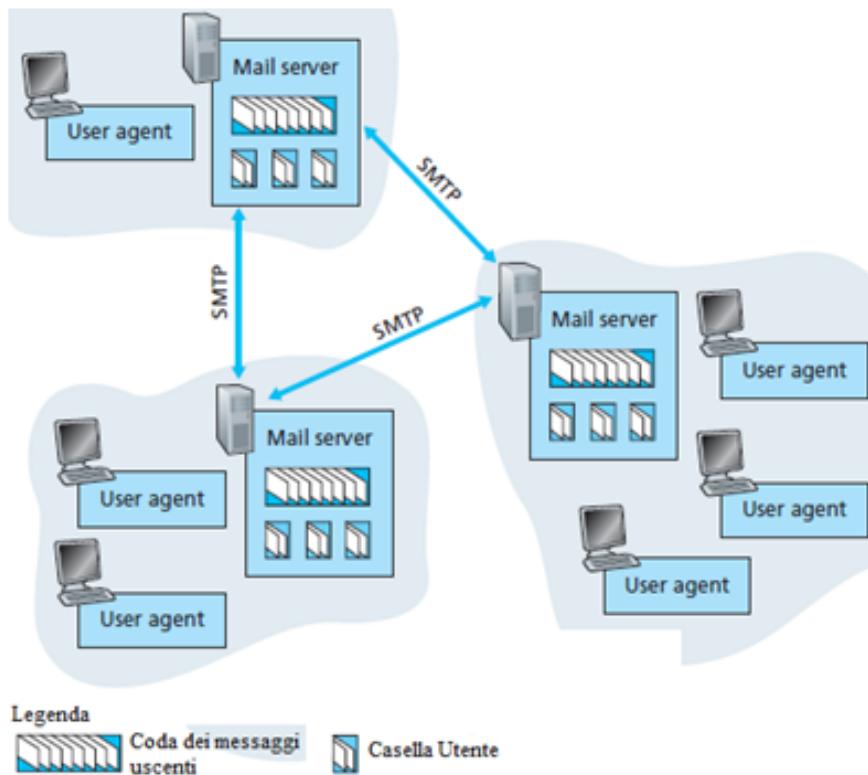
Come per la posta ordinaria, la posta elettronica è un mezzo di comunicazione asincrono - la gente invia i messaggi e li legge quando sono disponibili, senza doversi coordinare con gli orari delle altre persone. Rispetto alla posta ordinaria, la posta elettronica è veloce, facile da distribuire, e poco costosa.

La posta elettronica moderna ha molte caratteristiche in più rispetto alle prime versioni; un messaggio può avere gli allegati, i collegamenti ipertestuali, il testo in formato HTML, e le foto incorporate.

Caratteristiche principali

In questa sezione, vengono esaminati i protocolli del livello applicazione che sono usati dalla posta elettronica.

La Figura presenta una visione del sistema di posta usata su Internet. Ci sono tre componenti principali: gli **User-Agent**, i **server di posta**, e il **Simple Mail Transfer Protocol (SMTP)**. Il ruolo di ciascuno di questi componenti verrà descritto ipotizzando che un mittente, Alice, invia un messaggio di posta elettronica a un destinatario, Bob.



Gli **User-Agent** permetteranno agli utenti di leggere, rispondere, inoltrare, salvare e comporre messaggi. Mozilla Firefox, Microsoft Outlook e Apple Mail sono esempi di **User-Agent** per e-mail. Quando Alice ha terminato di comporre il suo messaggio, il suo **User-Agent** lo invia al server di posta che offre il servizio ad Alice; qui il messaggio viene inserito nella coda dei messaggi in uscita del server di posta. Quando Bob vuole leggere un messaggio, il suo **User-Agent** recupera il messaggio dalla casella di posta di Bob, nel server di posta che gli offre il servizio. Dato che Alice e Bob non devono essere contemporaneamente collegati per comunicare, la **e-mail è un servizio applicativo asincrono**.

I server di posta formano il nucleo del sistema di posta elettronica. Ogni destinatario, come Bob, ha una casella postale situata su un server di posta. La casella di posta di Bob gestisce e mantiene i messaggi che sono stati inviati a lui. Un messaggio tipico inizia il suo viaggio nello User-Agent del mittente, si reca al server di posta del mittente, e viaggia al server di posta del destinatario, dove viene depositato nella cassetta postale del destinatario.

Quando Bob vuole accedere ai messaggi nella sua casella di posta, il server di posta autentica Bob (con username e password).

Il Server di posta del mittente deve essere in grado di riconoscere gli errori che possono verificarsi nel server di posta del destinatario. Se il server del mittente non può consegnare la posta al server del destinatario, il server mittente conserva il messaggio in una coda di messaggi e tenterà il trasferimento successivamente. I nuovi tentativi sono spesso fatti ogni 30 minuti circa; se non vi è alcun successo dopo diversi giorni, il server mittente rimuove il messaggio e notifica la mancata consegna allo User-agent mittente con un messaggio di posta elettronica.

e-mail su Web

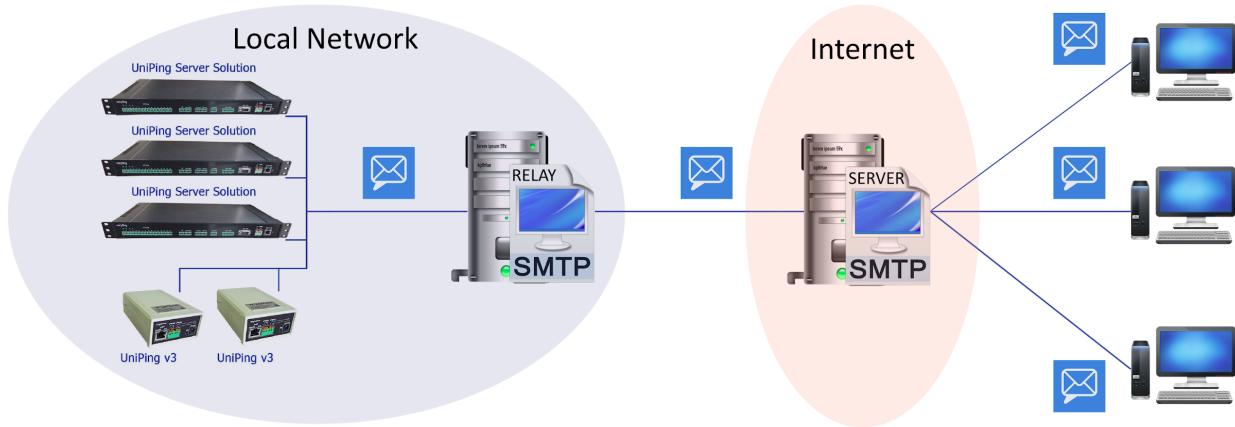
Nel dicembre 1995, solo pochi anni dopo che il Web fu inventato, Sabeer Bhatia e Jack Smith visitarono l'Internet venture capitalist Draper Fisher Jurvetson e proposero lo sviluppo di un sistema di e-mail gratuito basato sul web. L'idea era di dare un account di posta elettronica gratuita a chiunque lo volesse, e rendere gli account accessibili dal web. In cambio del 15 per cento della società, Draper Fisher Jurvetson finanziarono Bhatia e Smith, che costituirono una società denominata Hotmail.

Con tre persone a tempo pieno e 14 persone part-time che hanno lavorato per le stock option, sono stati in grado di sviluppare e lanciare il servizio nel luglio 1996. Meno di un mese dopo il lancio, avevano 100.000 abbonati. Nel dicembre 1997, meno di 18 mesi dopo il lancio del servizio, Hotmail ha oltre 12 milioni di abbonati ed è stata acquisita da Microsoft per 400 milioni di dollari. Il successo di Hotmail è attribuito al suo "vantaggio della prima mossa" e alla intrinseca diffusione di avvisi pubblicitari, anche involontaria, della posta elettronica.

La posta elettronica sul Web continua a prosperare, diventando più sofisticata e potente ogni anno. Uno dei servizi più popolari oggi è gmail di Google, che offre spazio di archiviazione gratuito, rilevamento e filtraggio dello spam, un'avanzata protezione da virus, la crittografia (utilizzando SSL), il recupero della posta dai servizi di posta elettronica di terze parti, e un'interfaccia orientata alla ricerca. Anche la Messaggistica asincrona all'interno delle reti sociali, come Facebook, è diventata popolare negli ultimi anni.

SMTP

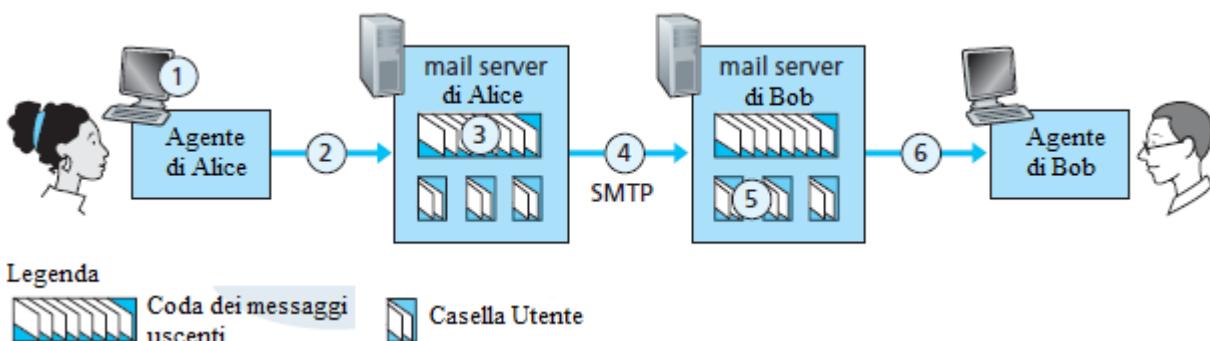
SMTP, definito nello [RFC 5321](#), è il principale protocollo del livello applicazione per la posta elettronica. È un **protocollo applicativo orientato alla connessione** e **utilizza** il servizio di trasferimento dati affidabile di **TCP** per trasferire la posta dal server di posta del mittente al server di posta del destinatario. SMTP ha un lato client, che viene eseguito sul server di posta del mittente, e un lato server, che viene eseguito sul server di posta del destinatario. Entrambi le componenti di SMTP funzionano su un server di posta. Quando un server di posta invia la posta ad altri server di posta, agisce come un client SMTP. Quando un server di posta riceve la posta da altri server di posta, agisce come un server SMTP.



Anche se SMTP ha numerose qualità, è una tecnologia vecchia, che possiede alcune caratteristiche arcaiche. Ad esempio, nelle intestazioni e nel corpo di tutti i messaggi di posta elettronica si usa la codifica ASCII a 7 bit. Questa limitazione aveva un senso nei primi anni '80, quando la capacità di trasmissione era scarsa e nessuno spediva allegati di grandi dimensioni come i file audio, video, ecc. Ma oggi, nell'era multimediale, la restrizione dei codici a 7 bit è inammissibile. Non conviene che i dati multimediali binari vengano codificati in ASCII prima di essere inviati tramite SMTP, per poi essere riconvertiti nuovamente in binario una volta raggiunto il ricevitore.

Per illustrare il funzionamento di base di SMTP, si esami un caso comune. Si supponga che Alice voglia inviare a Bob un messaggio di testo.

1. Alice richiama il suo User-Agent di posta elettronica, fornisce l'indirizzo e-mail di Bob (per esempio, bob@scuola.edu), compone un messaggio, e incarica l'User-Agent di inviare il messaggio.
2. Lo User-Agent di Alice invia il messaggio al suo server di posta, dove viene collocato in una coda di messaggi.
3. Il lato client di SMTP in esecuzione sul server di posta di Alice vede il messaggio nella coda di messaggi e apre una connessione TCP con il server SMTP in esecuzione sul server di posta di Bob.
4. Dopo l'handshaking iniziale, il client SMTP invia il messaggio di Alice sulla connessione TCP.
5. Al server di posta di Bob, il lato server SMTP riceve il messaggio. Il server di posta di Bob poi inserisce il messaggio nella casella di posta di Bob.
6. Quando ne ha la possibilità, Bob richiama il suo User-Agent per leggere il messaggio.



È importante osservare che SMTP normalmente non usa server di posta intermedi per conservare la posta, anche quando i due server di posta sono situati alle estremità opposte del mondo. Se il server di Alice è a Hong Kong e il server di Bob è a St. Louis, la connessione TCP è una connessione diretta tra i server di Hong Kong e di St. Louis. In particolare, se il server di posta di Bob è spento, il messaggio rimane nel server di posta di Alice, in attesa di un nuovo tentativo, il messaggio non viene inserito in alcun mail server intermedio.

Ma in che modo SMTP trasferisce un messaggio da un server di posta mittente ad un server di posta destinatario? Il protocollo SMTP ha molte analogie con i protocolli utilizzati nell'interazione diretta tra le persone.

In primo luogo, il **client SMTP** (in esecuzione su un server di posta mittente) deve stabilire una **connessione TCP** sulla **porta 25** del **server SMTP** (in esecuzione sulla macchina server di posta ricevente). Se il server è inattivo, il client riproverà più tardi. Una volta stabilita la connessione, i processi applicativi SMTP del server e del client eseguono una fase di handshake, proprio come le persone che si presentano prima di scambiarsi informazioni: il client e il server SMTP si presentano prima di trasferire le informazioni. Durante questa fase di handshaking a livello applicativo (SMTP), il client SMTP indica l'indirizzo e-mail del mittente (la persona che ha generato il messaggio) e l'indirizzo e-mail del destinatario. Una volta che il client e il server SMTP si sono sincronizzati, il client invia il messaggio.

SMTP può contare sul servizio di trasferimento dati affidabile e senza errori di TCP per consegnare il messaggio al server. Il client quindi ripete questo processo sulla stessa connessione TCP se ha altri messaggi da inviare al server; in caso contrario, chiede al TCP di chiudere la connessione.

Di seguito vedremo nel dettaglio la fase di trascrizione dei messaggi scambiati tra un client SMTP (c) e un server SMTP (s). L'hostname del client è `crepes.fr` e il nome host del server è `hamburger.edu`. Nel seguente esempio, le righe di testo ASCII precedute da `c:` sono esattamente le linee che il client invia nel suo socket TCP, e le linee di testo ASCII precedute da `s:` sono esattamente le linee che il server invia nel suo socket TCP. Il seguente dialogo inizia appena viene stabilita la connessione TCP.

```
S: 220 hamburger.edu
C: HELO crepes.fr
S: 250 Hello crepes.fr, pleased to meet you
C: MAIL FROM: <alice@crepes.fr>
S: 250 alice@crepes.fr ... Sender ok
C: RCPT TO: <bob@hamburger.edu>
S: 250 bob@hamburger.edu ... Recipient ok
C: DATA
S: 354 Enter mail, end with "." on a line by itself
C: From: alice@crepes.fr
C: To: bob@hamburger.edu
C: Subject: Considerazioni sul significato della nostra esistenza.
C:
```

```
C: La risposta ad ogni domanda è 42.  
C: .  
S: 250 Message accepted for delivery  
C: QUIT  
S: 221 hamburger.edu closing connection
```

Nell'esempio precedente il client invia un messaggio ("Ti piace il ketchup? Che ne dici dei sottaceti?") Dal server di posta `crepes.fr` al server di posta `hamburger.edu`. Nell'ambito del dialogo, il client ha emesso cinque comandi: HELO (abbreviazione di HELLO), MAIL FROM, RCPT TO, DATA, e QUIT. Questi comandi sono auto esplicativi. Il client invia anche una linea costituita da un unico punto, o meglio, ogni messaggio si conclude con CRLF.CRLF, dove CR e LF sono rispettivamente i codici per il ritorno a capo e avanzamento riga.

Il server risponde ad ogni comando ricevuto con un codice e alcune opzionali spiegazioni in inglese. Si ricordi che **SMTP usa connessioni persistenti**: se il server di posta di **invio** ha **diversi messaggi da inviare allo stesso server di posta ricevente**, può **inviare tutti i messaggi sulla stessa connessione TCP**. Per ogni messaggio il client inizia il processo con una nuova MAIL FROM: <alice@crepes.fr>, indica la fine del messaggio con un punto (CRLF.CRLF), e invia QUIT solo dopo che tutti i messaggi sono stati inviati.

Utilizzare `telnet` per svolgere un dialogo diretto con un server SMTP. Per svolgere questo esercizio scrivere `telnet servername 25` dove `servername` è il nome di un server di posta locale. Quando si esegue questa operazione, si sta semplicemente stabilendo una connessione TCP tra l'host locale e il server di posta. Dopo aver digitato questa riga, si dovrebbe ricevere immediatamente la risposta 220 dal server. Quindi inviare i comandi SMTP: HELO, MAIL FROM, RCPT TO, DATA, CRLF.CRLF, e QUIT al momento opportuno. Si suggerisce di documentarsi sulla possibilità di costruire, in un linguaggio di programmazione di propria conoscenza, un semplice User-Agent che implementa il lato client di SMTP, con il quale si potrà inviare un messaggio di posta elettronica a un destinatario arbitrario tramite un server di posta locale.

Formati dei messaggi di posta

Quando Alice scrive una lettera ordinaria a Bob, può includere tutti i tipi di informazioni ausiliarie sulla busta della lettera, come l'indirizzo di Bob, il suo indirizzo di ritorno. Analogamente, quando un messaggio di posta elettronica viene inviato da una persona all'altra, un'intestazione contenente informazioni ausiliarie precede il corpo del messaggio stesso. Queste informazioni ausiliarie sono contenute in una serie di righe di intestazione, definite in [RFC 5322](#). Le linee di intestazione e il corpo del messaggio sono separate da una riga vuota (cioè da CRLF). [RFC 5322](#) specifica il formato esatto per le linee di intestazione di posta così come il loro significato. Come con HTTP, ciascuna riga di intestazione contiene il testo in chiaro, costituito da una parola chiave seguita da due punti, e da un valore. Alcune delle parole chiave sono obbligatorie e altre sono facoltative. Ogni intestazione deve avere un campo `From:` e una riga header `To:`. Lo header può contenere anche una riga di `Subject:` così come altre linee di intestazione opzionali. È importante notare che queste righe di intestazione sono diverse dai comandi SMTP già visti (anche se contengono alcune

parole comuni come "from" e "to"). I comandi in quella sezione erano parte del protocollo di sincronizzazione SMTP; le linee di intestazione esaminate in questa sezione fanno parte del messaggio di posta stesso (questa differenza ha delle implicazioni importanti in termini di sicurezza, quali?).

Una tipica intestazione del messaggio, e relativo messaggio, potrebbe essere la seguente:

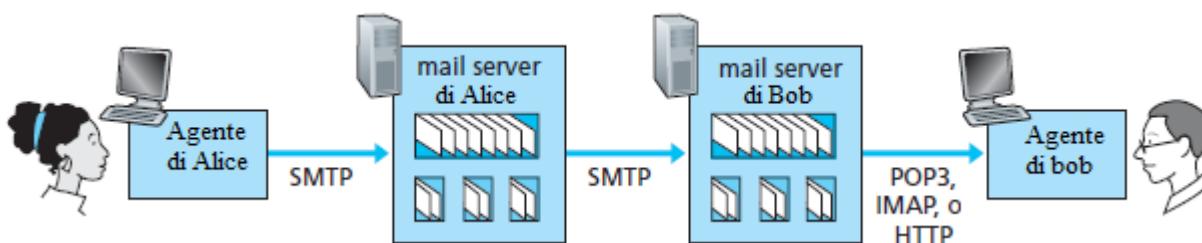
```
From: alice@crepes.fr  
To: bob@hamburger.edu  
Subject: Considerazioni sul significato della nostra esistenza.
```

La risposta ad ogni domanda è 42.

L'intestazione del messaggio è seguita da una riga vuota e poi dal corpo del messaggio (in ASCII). È possibile utilizzare `telnet` per inviare un messaggio a un server di posta che contiene alcune righe di intestazione. Per fare questo, scrivere `telnet servername 25`, come discusso nella sezione precedente.

Mail Access Protocol

Una volta che SMTP recapita il messaggio dal server di posta di Alice al server di posta di Bob, il messaggio viene inserito nella casella di posta di Bob. Nel corso di questa discussione è stato tacitamente assunto che Bob legge la sua posta elettronica autenticandosi sul server, e quindi usando un lettore di posta in esecuzione sul suo dispositivo locale, un programma denominato **User-Agent**, con il quale accede alla sua casella di posta memorizzata su un server di posta condiviso e sempre attivo. Questo server di posta è condiviso con altri utenti e viene in genere gestito dallo ISP dell'utente (ad esempio, università o società).



Un messaggio di posta elettronica, quando viene inviato da Alice a Bob, deve essere depositato nel server di posta di Bob utilizzando SMTP, che è stato progettato per trasferire le e-mail dallo host mittente al server di posta del destinatario. Tuttavia lo User-Agent del mittente non dialoga direttamente con il server di posta del destinatario. Invece, come mostrato in figura, l'User-Agent di Alice utilizza SMTP per depositare il messaggio di posta elettronica nel suo server di posta, quindi il server di posta di Alice utilizza il protocollo SMTP (come un client SMTP) per inoltrare il messaggio di posta elettronica al server di posta di Bob.

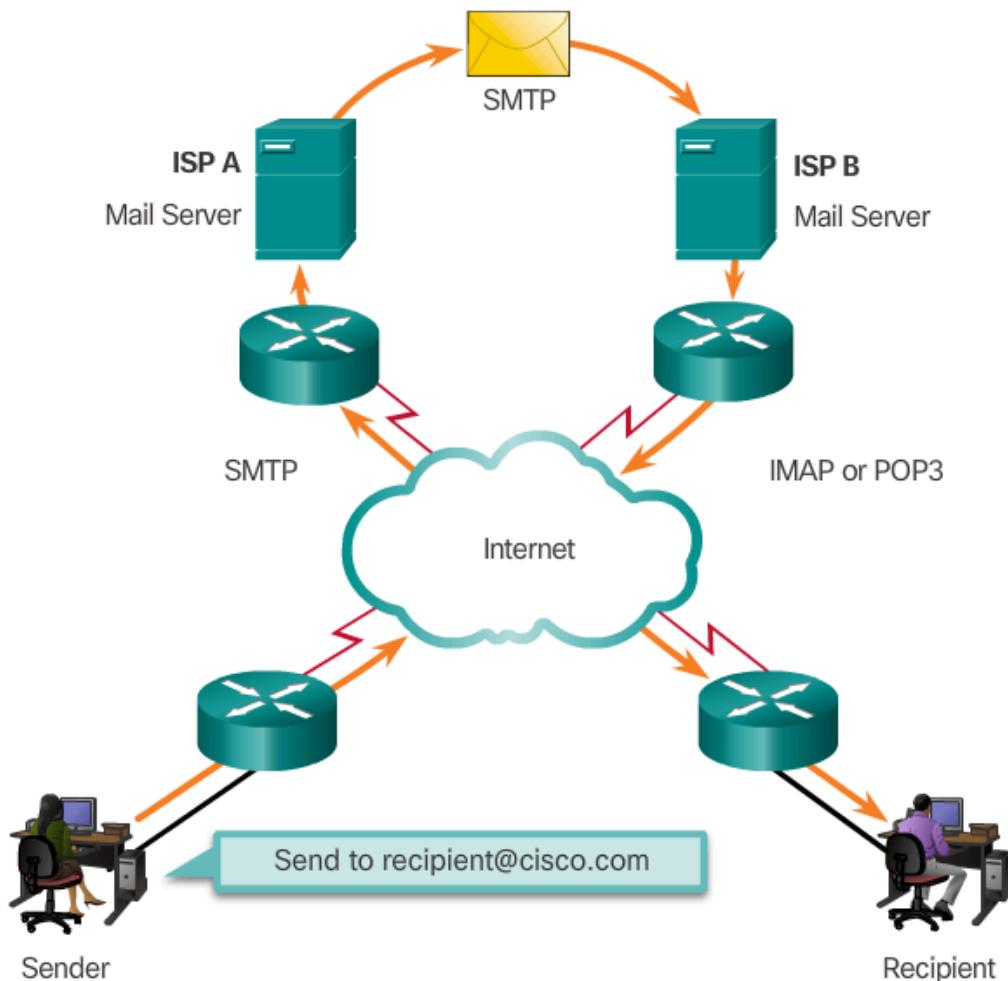
La ragione principale per cui questa procedura avviene in due fasi risiede soprattutto sul fatto che senza l'inoltro attraverso il server di posta di Alice, l'User-Agent di Alice

non potrebbe ricevere l'eventuale notifica che il server di posta di destinazione è irraggiungibile. Avendo Alice depositato prima l'e-mail nel proprio server di posta, il server di posta di Alice può ripetutamente provare a inviare il messaggio al server di posta di Bob, ad esempio ogni 30 minuti, fino a quando il server di posta di Bob diventa operativo.

Un'altra ragione della necessità di separare in fasi distinte il **prelievo**, **l'invio** e la **ricezione** dei messaggi di posta risiede nel fatto che lo User-agent di Bob non può utilizzare il protocollo SMTP per ottenere i messaggi perché ottenere i messaggi è un'operazione di prelievo, mentre **SMTP è un protocollo di inserimento**. Il problema viene risolto con l'introduzione di un **apposito protocollo di accesso alla posta che trasferisce i messaggi dal server di posta** di Bob **al suo dispositivo locale**. Attualmente ci sono una serie di protocolli di accesso alla posta, tra cui Post Office Protocol - versione 3 (**POP3**), Internet Mail Access Protocol (**IMAP**) e **HTTP**.

L'elenco seguente fornisce un sintetico riepilogo delle funzioni principali dei protocolli utilizzati per gestire la posta elettronica:

- **SMTP** è utilizzato per **trasferire la posta** dal server di posta del mittente al server di posta del destinatario;
- **SMTP** è utilizzato anche per **trasferire la posta** dall'User-Agent del mittente al server di posta del mittente;
- **POP3 o IMAP** sono dei protocolli di **accesso alla posta**, utilizzati per trasferire la posta dal server del destinatario all'User-Agent del destinatario.



POP3

POP3 è un **protocollo di accesso alla posta** estremamente semplice, definito in [RFC 1939](#). Poiché il protocollo è molto semplice, la sua funzionalità è piuttosto limitata. **POP3** inizia quando l'User-Agent (client) apre una **connessione TCP** con il server di posta (server) sulla **porta 110**. Con la connessione TCP stabilita POP3 attraversa tre distinti fasi per effettuare il recupero della posta: **l'autorizzazione**, la **transazione** e **l'aggiornamento**.

Durante la **fase di autorizzazione** l'User-Agent invia al server di posta il nome utente e la password (in chiaro) per autenticare l'utente.

Durante la **fase di transazione** l'User-Agent recupera i messaggi e può anche contrassegnare i messaggi per l'eliminazione, rimuovere i segni di cancellazione e ottenere statistiche di posta.

La terza **fase, l'aggiornamento**, si verifica dopo che il client ha emesso il comando quit, che termina la sessione POP3; in questo momento, il **server di posta elimina i messaggi che sono stati contrassegnati per la cancellazione**.

In una transazione POP3 lo User-Agent invia comandi e il server risponde ad ogni comando con un messaggio. Ci sono due possibili risposte:

- +OK (a volte seguita dai dati da server a client) utilizzata dal server per indicare che il comando precedente è stato correttamente interpretato;
- -ERR, utilizzata dal server per indicare che è avvenuto qualche cosa di sbagliato nel comando precedente.

La **fase di autorizzazione** ha due comandi principali: USER <username> e PASS <password>. Per illustrare questi due comandi, si consiglia di avviare una sessione telnet direttamente in un server POP3, utilizzando la porta 110, e scrivendo questi comandi, supponendo che mailServer sia il nome del server di posta. Si vedrà qualcosa di simile:

```
C: telnet mailServer 110
S: +OK POP3 server ready
C: user bob
S: +OK
C: pass hungry
S: +OK user successfully logged on
```

Se si sbaglia l'ortografia di un comando, il server POP3 risponderà con il messaggio -ERR.

Nella **fase di transazione** uno User-Agent che utilizza POP3 è usualmente configurato per scaricare e cancellare le e-mail dal server. POP3 può essere anche configurato per scaricare e conservare le e-mail, ma essendo un protocollo molto semplice, manca dei meccanismi necessari per rendere l'operazione realmente efficace; in questo caso viene usato il protocollo **IMAP**.

Uno User-Agent POP3 emetterà i comandi `list`, `retr` e `dele` per scaricare e cancellare le e-mail dal server di posta. Nel dialogo mostrato di seguito C: (client) è lo User-Agent e S: (server) è il server di posta.

```
C: list
S: 1 498
S: 2 912
S: .
C: retr 1
S: (blah blah ...
S: .....
S: .....blah)
S: .
C: dele 1
C: retr 2
S: (blah blah ...
S: .....
S: .....blah)
S: .
C: dele 2
C: quit
S: +OK POP3 server signing off
```

Il programma utente prima chiede al server di posta di elencare le dimensioni di ciascuno dei messaggi memorizzati, poi recupera il messaggio desiderato, eliminandolo subito dopo dal server. Si noti che dopo la fase di autorizzazione, l'User-Agent impiega solo quattro comandi: `list`, `retr`, `dele`, e `quit`. La sintassi per questi comandi è definita nello [RFC 1939](#). Dopo aver elaborato il comando `quit`, il server POP3 entra nella fase di aggiornamento e rimuove i messaggi 1 e 2 dalla casella postale.

La tipica modalità di funzionamento di POP3 pone il problema di partizionare i messaggi di posta nei diversi dispositivi usati dall'utente: se Bob legge un messaggio nuovo sul suo PC in ufficio, non sarà in grado di rileggere il messaggio dal suo portatile a casa.

Durante una sessione POP3 il server mantiene alcune informazioni sullo stato; in particolare, tiene traccia dei messaggi che gli utenti hanno contrassegnato "da eliminare". Tuttavia, il server POP3 non trasporta informazioni sullo stato tra le sessioni POP3. Questa mancanza di informazioni sullo stato tra le sessioni semplifica notevolmente l'implementazione di un server POP3 non rendendolo però un protocollo efficace per la consultazione della posta da diversi dispositivi.

IMAP

Con il protocollo **IMAP**, definito nello [RFC 3501](#), si sono superate molte delle limitazioni di POP3, a fronte però di una maggiore complessità del protocollo.

Un server **IMAP associa ogni messaggio con una cartella**; quando un messaggio arriva al server viene automaticamente associato alla cartella "Posta in arrivo" del destinatario. Il destinatario può spostare il messaggio in una nuova cartella, leggere il messaggio, eliminare il messaggio, e così via. Il protocollo IMAP fornisce comandi per permettere agli utenti di creare cartelle e spostare i messaggi da una cartella ad un'altra. IMAP fornisce anche i comandi che consentono agli utenti di cercare nelle cartelle remote i messaggi che corrispondono a criteri specifici.

Si noti che, a differenza di POP3, un **server IMAP mantiene informazioni sullo stato dell'utente tra le sessioni IMAP**, come ad esempio i nomi delle cartelle e quali messaggi sono associati ad esse, **permettendo all'utente di consultare i messaggi ricevuti utilizzando diversi dispositivi**.

Il protocollo IMAP fornisce inoltre dei comandi mirati all'ottenimento di singoli componenti dei messaggi. Ad esempio un User-Agent può ottenere solo l'intestazione di un messaggio o solo una parte di un messaggio MIME. Questa funzione è utile quando vi è una connessione a bassa larghezza di banda (ad esempio, un collegamento modem a bassa velocità) tra l'User-Agent e il relativo server di posta. Con una connessione a bassa larghezza di banda, l'utente può decidere di non scaricare tutti i messaggi nella sua casella di posta, evitando i messaggi lunghi che potrebbero contenere, ad esempio, una clip audio o video.

E-Mail nel Browser

Sempre più utenti utilizzano la posta elettronica attraverso i browser web. Hotmail ha introdotto un accesso basato sul Web a metà degli anni '90. Ora si sono diffusi servizi di **e-mail accessibili via Web** forniti da Google, Yahoo!, così come quasi tutte le principali università e società. Con questo servizio, **l'User-Agent è un comune browser e l'utente comunica con la sua casella e-mail remota via HTTP**. Quando un destinatario, come Bob, vuole accedere a un messaggio nella sua casella di posta, il messaggio di posta elettronica viene inviato dal server di posta di Bob al browser di Bob utilizzando il protocollo HTTP anziché il protocollo POP3 o IMAP. Quando un mittente, come Alice, vuole inviare un messaggio di posta elettronica, questo viene inviato dal proprio browser al suo server di posta su HTTP anziché su SMTP. Alice, però, continua a inviare messaggi e a ricevere messaggi da altri server di posta tramite SMTP.

D - Sicurezza informatica e crittografia



"On the Internet, nobody knows you're a dog."

Credit: Peter Steiner. © The New Yorker magazine

By Source, Fair use, <https://en.wikipedia.org/w/index.php?curid=13627120>

Prefazione

I seguenti appunti sono in parte basati sul lavoro svolto dalla Prof.ssa Sophia Danesino, mentore, amica, appassionata di conoscenza e insegnante fuori dal comune. Sul suo [canale YouTube](#) è possibile trovare i video su molti degli argomenti trattati.

Per l'arricchimento e la stesura finale degli appunti sono risultate fondamentali le lezioni del [Prof. Steven Gordon](#) a cui va la mia profonda gratitudine per la pubblicazione delle sue lectures sul suo [canale YouTube](#).

Sicurezza in rete - 1^a parte

La sicurezza delle reti

Libro vol.3 - La sicurezza nei sistemi informativi

STUDIARE: *L. Lo Russo, E. Bianchi, Sistemi e Reti - Nuova Edizione OPENSCHOOL - Per l'articolazione INFORMATICA degli Istituti Tecnici settore Tecnologico, vol. 3, ed. Hoepli, 2017.*

- UdA4 - La sicurezza delle reti
 - Lezione 1 - La sicurezza dei sistemi informativi.
 - La sicurezza dei dati pp.166-168
 - Sicurezza di un sistema informatico pp.169-170
 - Valutazione dei rischi pp.170-172
 - Principali tipologie di minacce pp.172-173 - [SYN attack](#)
 - Sicurezza nei sistemi informativi distribuiti pp.173-175
- [S/MIME](#) - [Wikipedia](#)

Crittografia simmetrica

Tecniche crittografiche per la protezione dei dati

Libro vol.3 - La crittografia simmetrica

STUDIARE: L. Lo Russo, E. Bianchi, Sistemi e Reti - Nuova Edizione OPENSCHOOL - Per l'articolazione INFORMATICA degli Istituti Tecnici settore Tecnologico, vol. 3, ed. Hoepli, 2017.

- UdA3 - Tecniche crittografiche per la protezione dei dati
 - Lezione 1 - La crittografia simmetrica
 - La sicurezza nelle reti pp.96-97
 - Crittografia pp.97-98
 - Crittoanalisi pp.98-99
 - Cifrari e chiavi pp.99-100
 - Il cifrario DES pp.100-102
- CLIL Listening - [Introduction to Cryptography](#) del Prof. Steven Gordon (*il video è utile per comprendere meglio quanto spiegato dal libro, con l'aggiunta di alcuni approfondimenti, ed è parte integrante del materiale di studio*)
- UdA3 - Tecniche crittografiche per la protezione dei dati
 - Lezione 1 - La crittografia simmetrica
 - 3-DES pp.102-103 (*è necessario comprenderne la ragione dell'uso dell'algoritmo e spiegarlo in termini generali, non serve il dettaglio*)
 - IDEA p.103 (*è necessario comprenderne la ragione dell'uso dell'algoritmo e spiegarlo in termini generali, non serve il dettaglio*)
 - AES pp.103-105 (*è necessario comprenderne la ragione dell'uso dell'algoritmo e spiegarlo in termini generali, non serve il dettaglio*)
 - Limiti degli algoritmi simmetrici p.106
- CLIL Listening - [Symmetric Key Encryption and Brute Force Attacks](#) del Prof. Steven Gordon (*il video è utile per comprendere meglio quanto spiegato dal libro e presenta degli interessanti esempi di brute force attack su DES, 3-DES e AES, ed è parte integrante del materiale di studio*)

CLIL - Cryptography Introduction

[Introduction - Applied Cryptography](#)

[Introduction Solution - Applied Cryptography](#)

CLIL Speaking Activity - Introduction to Cryptography

1. Which is the meaning of the word "cryptography"?
2. Which is the meaning of the word "cryptology"?
3. Which are the most strange answers you watched on the second video?
4. Why did the teacher decide not typing his password on video?
5. What does Google.com use to ensure security?
6. Did you find something that you do not know in these video?

13 - CLIL - Homework

Study all the topics your teacher spoke about and watch the following videos.

1. [Introduction to internet security](#)
2. [Who are Alice and Bob?](#)
3. [What is end-to-end encryption?](#)
4. [What is a plaintext? What is a ciphertext?](#)

CLIL - Substitution ciphers

CLIL - Breaking a substitution cipher: brute force attack

Breaking the scheme is *straightforward*. Since there are only a limited number of possible shifts (25 in English⁴⁵: if you shift by 26 you are back to where you started), they can each be tested in turn in a **brute force attack**. One way to do this is to write out a snippet of the ciphertext in a table of all possible shifts. The example given is for the ciphertext "EXXEGOEXSRGI":

Decryption shift	Candidate plaintext
0	exxegoexsrgi
1	dwwdfndwrqfh
2	cvvcemcvqpeg
3	buubdlbupodf
4	attackatonce
5	zsszbjzsnmbd
6	yrryaiyrm lac
...	
23	haahjravujl
24	gzzgiqqzutik
25	fyypfytshj

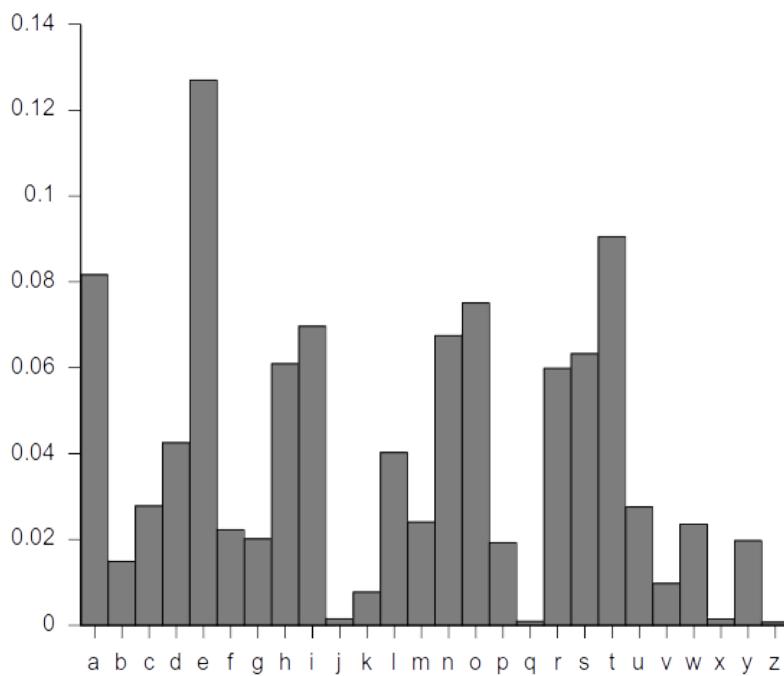
But if it is used a match between english alphabet and a randomly disposition of the same alphabet, the possible keys become **26!**

This is due to the fact that there are **26** possible keys for the letter **a**, **25** remaining keys for the letter **b**, **24** remaining keys for the letter **c**, and so on.

⁴⁵ The modern *English alphabet* consists of 26 letters.

CLIL Reading Activity - Frequency analysis

Frequency analysis⁴⁶ is the study of letters or groups of letters contained in a ciphertext in an attempt to partially reveal the message. The English language (as well as most other languages) have certain letters and groups of letters appearing in varying frequencies.



This is a chart of the **frequency distribution of letters in the English alphabet**. As you can see, the letter 'e' is the most common, followed by 't' and 'a', with 'j', 'q', 'x', and 'z' being very uncommon.

Knowing the usual frequencies of letters in English communication, if the encryption method does not effectively mask these frequencies it is possible to statistically determine parts of the plaintext from looking at the ciphertext alone. Let's look at an example based on a plaintext encrypted with the Caesar Cipher – a cipher that provides no protection from frequency analysis.

wkh sdvvzrug lv vhyhq grqw whoo dqbrqh

Let's get the letter frequencies (how often each letter appears) of this ciphertext.

h = 5
v = 4
q = 3
r = 3
g = 3
d = 2
b = 1
k = 1

⁴⁶ <http://learncryptography.com/frequency-analysis/>

I = 1
s = 1
y = 1

Analysing the letter frequencies, it can be tried h = e, and since it is known the cipher used is the Caesar cipher, it can be used a shift of -3, revealing the true message.

14 - CLIL Listening, Speaking Activity - Caesar and Vigenère cipher

Vocabulary

Flaw, narrow sth down, consistent, blow

Watch the videos and answer to the questions below:

1. [The Caesar cipher | Journey into cryptography | Computer Science | Khan Academy](#)
2. [Vigenère Cipher](#)

Questions:

1. What is frequency analysis?
2. Which is the main difference between a monoalphabetic and a polyalphabetic cipher?
3. What is a brute force attack?
4. Using monoalphabetic substitution cipher how many checks does a brute-force attack in the worst case?
5. Vocabulary - Which is the meaning of the following words or phrases?

have a crush on =

to bud =

at all cost =

to shift the alphabet BY a certain numbers of letters =

to shift a letter BY one/two/.. place/s TO another letter =

to crack the code =

it occurs almost 13% of the time =

to thwart =

to rip =

to rip through sth =

to plague =

as a brief aside =

to match up =

to end up with =

to meet up =

it's worth it to ... =

interwoven =

to crop up =

weird =

prying =

15 - Esercizi

1. Uno dei primi esempi famosi di crittografia si trova nel De Bello Gallico di Cesare. L'autore racconta del riuscito invio di un messaggio a Cicerone, assediato e sul punto di arrendersi. Cesare usò una cifratura detta per trasposizione, che consisteva in un alfabeto in chiaro (quello ordinario) e un alfabeto cifrante ottenuto sostituendo ogni lettera dell'alfabeto ordinario con una lettera che lo rimpiazza nel crittogramma. Basandosi su questa tecnica, indicare qual è il messaggio originale di **LNAONR FHVDUH**. Indicare quanti tentativi si devono fare al massimo per tornare al messaggio originale?
2. Utilizzando il cfrario di Cesare (chiave 3), a cosa corrisponde questo messaggio? DOHD NDFWD HVZ. Mostrare i passaggi per ottenere il testo in chiaro.
3. Quante sono le possibili chiavi nel cfrario di Cesare?
4. Encrypt and decrypt the following message "HELLO FROM ITALY" using a [polyalphabetic cipher](#) (use the key "CLIL"). Write also all the steps to obtain the ciphertext.
5. Utilizzando la [tecnica a traslazione polialfabetica](#) e la chiave data, trovare la parte mancante mostrando i passaggi effettuati:
 - a. Key: COUNTON
Plain text: VIGNENERECHIPER
Encryption:
 - b. Key: NETWORK
Plain text:
Encryption: FIXUCLDBQHNFFG
 - c. Key: CIPHER
Plain text:
Encryption: DZJAI WQZRL EKVIRR

È possibile usare il seguente schema per effettuare immediatamente la conversione:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

6. Dato questo testo [cifrato con il quadro di Vigenere](#) **MEGBSMXFUQHIUUEOS** usando la parola chiave **VERME**, ricostruire il messaggio originale mostrando i passaggi effettuati.

CLIL - Transposition cipher

In cryptography, a transposition cipher is a method of encryption by which the positions held by units of plaintext (which are commonly characters or groups of characters) are shifted according to a regular system, so that the ciphertext constitutes a permutation of the plaintext. That is, the order of the units is changed (the plaintext is reordered). Mathematically a bijective function is used on the characters' positions to encrypt and an inverse function to decrypt.⁴⁷

16 - CLIL Listening Activity - Transposition cipher

Watch this video and then make the following exercises.

[Cryptography: Transposition Cipher](#)

1. Decipher the message TTNAAPMTSUOAODWCOIXKNLYPETZ using a transposition cipher with the key: 3 4 2 1 5 6 7
TTNAAPMTSUOAODWCOIXKNLYPETZ
Write the steps to obtain the plaintext.
2. Encrypt the message MEET ME NEAR THE CLOCK TOWER AT TWELVE MIDNIGHT TONIGHT using the key LEMON.
Write the steps to obtain the ciphertext.

⁴⁷ https://en.wikipedia.org/wiki/Transposition_cipher

17 - Laboratorio - Cifrario monoalfabetico

Implementare un algoritmo di crittografia basato su una classe Java che codifichi una stringa di caratteri in ingresso in una stringa criptata illeggibile.

Il sistema deve funzionare attraverso una chiave numerica che prende tutte le occorrenze di ogni lettera della stringa e le sostituisce con la lettera che sta in k (chiave numerica) posizioni più a destra in ordine alfabetico. La chiave k deve essere pari al numero del vostro computer.

Si usi come riferimento la codifica ASCII dei caratteri per effettuare i calcoli necessari.

Si dovrà produrre:

1. un'analisi sintetica dei problemi logici affrontati indicando come si sono risolti;
2. il diagramma UML delle classi;
3. l'implementazione dell'algoritmo.

CLIL Listening Activity - Symmetric Cryptosystems

[Symmetric Cryptosystems - Applied Cryptography](#)

CLIL Listening Activity - Correctness

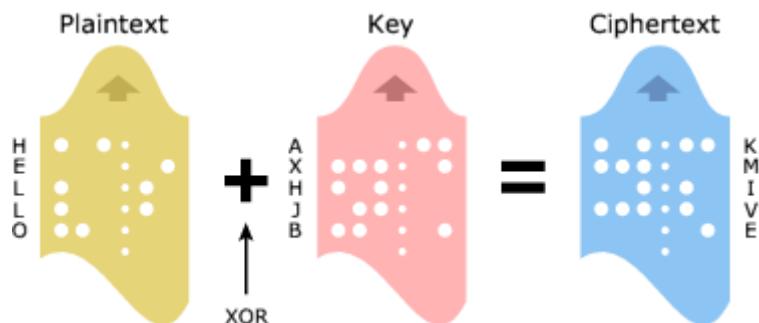
[Correctness And Security - Applied Cryptography](#)

[Correctness And Security Solution - Applied Cryptography](#)

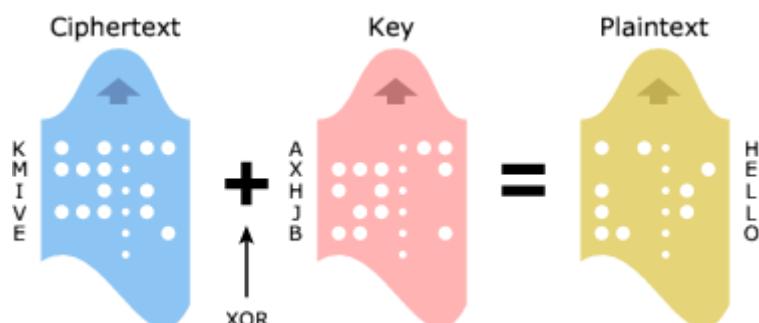
CLIL - XOR Cipher and One Time Pad encryption

CLIL - XOR Cipher

In cryptography, the simple **XOR cipher** is an encryption algorithm that operates according to the principles:



where \oplus denotes the exclusive disjunction (XOR) operation. With this logic, **a string of text can be encrypted by applying the bitwise XOR operator to every character using a given key**. To decrypt the output, merely reapplying the XOR function with the key will remove the cipher.



For example, the string "Wiki" in 8-bit ASCII is

01010111 01101001 01101011 01101001

and it can be encrypted with the repeating key 11110011 as follows:

$$\begin{array}{r}
 01010111 01101001 01101011 01101001 \\
 \oplus 11110011 11110011 11110011 11110011 \\
 \hline
 = 10100100 10011010 10011000 10011010
 \end{array}$$

And conversely, for decryption:

$$\begin{array}{r}
 10100100 10011010 10011000 10011010 \\
 \oplus 11110011 11110011 11110011 11110011 \\
 \hline
 = 01010111 01101001 01101011 01101001
 \end{array}$$

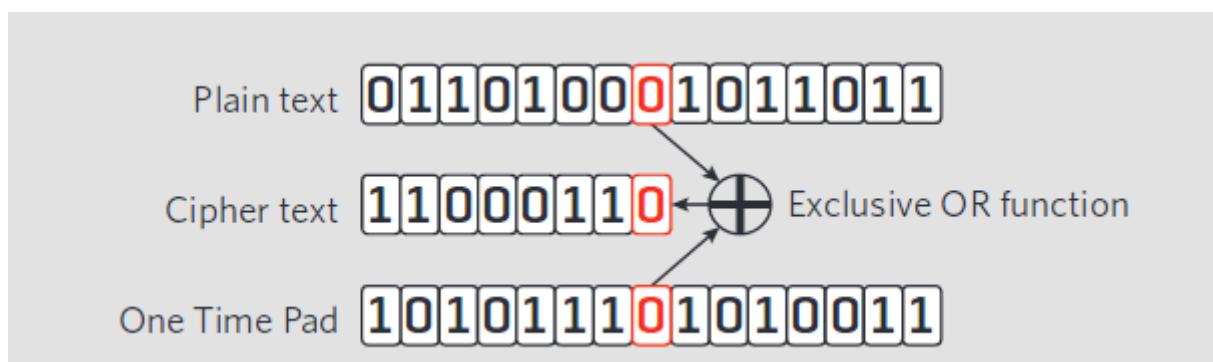
Its primary merit is that it is simple to implement and XOR operation is **computationally inexpensive**.

CLIL - One Time Pad encryption method



If the key is random and is at least as long as the message, the XOR cipher is much more secure than using a key repetition within a message.

With a key that is truly random, the result is a one-time pad (also known as the Vernam⁴⁸ cipher) **which is unbreakable even in theory**.



It is possible to **prove** that a stream cipher encryption scheme⁴⁹ is **unbreakable** if the following precondition are met:

1. **the key must be as long as the plain text;**
2. **the key must be truly random;**
3. **the key must only be used once.**

One Time Pad keys are used in pairs. The keys are distributed securely prior to encryption. One copy of the key is kept by the sender and one by the recipient.

⁴⁸ Gilbert Vernam invented and patented his cipher in 1917 while working at AT&T.

⁴⁹ https://en.wikipedia.org/wiki/Stream_cipher

1. To encrypt plain text data, the sender uses a key string equally long as the plain text. The key is used by mixing (XOR-ing) bit by bit, always adding one bit of the key with one bit of the plain text to create one bit of ciphertext.
2. This ciphertext is then sent to the recipient.
3. At the recipient's end, the encoded message is mixed (XOR-ed) with the duplicate copy of the One Time Key and the plain text is restored.
4. Both sender's and recipient's keys are automatically destroyed after use, so that erroneous re-application of the same key is impossible.



The **most critical aspect** of the Vernam cipher is the **randomness of the pad sequence**. An **event sequence** can be said to be **truly random if it is impossible to predict the next event in the sequence even if the entire state of the generating process up to that point is known**. Any deterministic process, such as running software on a computer, can never produce truly random numbers.

18 - CLIL - Homework

Study all the topics your teacher spoke about and watch the following videos.

1. [What is symmetric encryption?](#)
2. [What is a substitution cipher?](#)
3. [What is a cryptographically secure hash function?](#)

Esercizi sulla crittografia simmetrica

Nei seguenti esercizi si farà riferimento alla seguente notazione:

- $C = E(K_{XY}, P)$ con la quale si identifica un messaggio cifrato C scambiato tra due utenti X e Y , ottenuto applicando sul testo in chiaro P un algoritmo crittografico simmetrico $E()$ che usa la chiave simmetrica K_{XY} condivisa tra i due utenti X e Y .
- $P = D(K_{XY}, C)$ con la quale si identifica un messaggio in chiaro P scambiato tra due utenti X e Y , ottenuto applicando al testo cifrato C un algoritmo de-crittografico $D()$ che usa la chiave simmetrica K_{XY} condivisa tra i due utenti X e Y .

Esercizi

1. Due utenti X ed Y decidono di comunicare in modo confidenziale utilizzando la crittografia simmetrica e si scambiano il seguente messaggio cifrato:
 $C = E(K_{XY}, P)$
Supporre che l'utente Z intercetti il testo cifrato C e tenti di eseguire la seguente operazione:
 $D(K_{XZ}, C)$
Si spieghi cosa può assumere l'utente Z da questo tentativo in base a quanto appreso finora sulla crittografia simmetrica.
2. Si supponga che l'utente X invii il messaggio cifrato $C = (K_{XY}, P)$. L'utente Y riceverà il messaggio cifrato C_r . Si spieghi cosa potrà desumere l'utente Y nei due seguenti casi:
 - a. $D(K_{XY}, C_r) = P$
 - b. $D(K_{XY}, C_r) \neq P$
3. Con le conoscenze acquisite finora, spiegare se la crittografia simmetrica può essere usata per l'autenticazione.

Cenni di teoria dei numeri [da concludere]

Teorema di Eulero

La funzione toziente di Eulero

In matematica, la **funzione φ di Eulero** o semplicemente **funzione di Eulero** o **toziente**, è una funzione che, per ogni intero positivo n , è definita come il **numero degli interi compresi tra 1 e n e minori di n che sono coprimi con n** . Ad esempio, $\varphi(8) = 4$ poiché i numeri coprimi di 8 sono quattro: 1, 3, 5, 7.

Valgono le seguenti proprietà:

- $\varphi(1) = 1$
- Dato il numero primo p , $\varphi(p) = p - 1$
- Dati due valori a e b coprimi e $n = a * b$, allora
$$\varphi(n) = \varphi(a) * \varphi(b)$$
- Dati due numeri primi diversi p e q e calcolato $n = p * q$, allora
$$\varphi(n) = (p - 1) * (q - 1)$$

Esempi sulla funzione toziente di Eulero

- Dati due valori a e b coprimi⁵⁰ e $n = a * b$, allora
$$\varphi(n) = \varphi(a) * \varphi(b)$$

$a = 5$, $b = 6$ coprimi perché $MCD(5, 6) = 1$

$$n = 5 * 6 = 30$$

$$\varphi(30) = \varphi(5 * 6)$$

ma dato che 5 è un numero primo sappiamo che $\varphi(5) = 5 - 1 = 4$, invece, dato che 6 non è un numero primo bisogna calcolare il risultato di $\varphi(6) = 2$, dato dai valori $\{1, 5\}$, quindi

$$\varphi(30) = \varphi(5 * 6) = \varphi(5) * \varphi(6) = 4 * 2 = 8$$

- Dati due numeri primi diversi p e q e calcolato $n = p * q$, allora
$$\varphi(n) = (p - 1) * (q - 1)$$

$p = 7$, $q = 11$ numeri primi diversi fra loro

⁵⁰ $MDC(a, b) = 1$

$$n = 7 * 11 = 77$$

$$\phi(77) = \phi(7) * \phi(11) = (7 - 1) * (11 - 1) = 6 * 10 = 60$$

Queste tecniche di scomposizione vengono usate anche dai computer per velocizzare i calcoli. Si osservi quindi che dato un valore **n**, se questo è fattorizzabile come prodotto di numeri primi, il calcolo della funzione di Eulero diventa banale.

Si provi ora a calcolare $\phi(143)$. Per poterlo fare è necessario fattorizzare il valore 143, ma senza alcuna informazione ulteriore non è banale fattorizzare in numeri primi 143, nonostante 143 sia un numero piccolo.

Teorema di Eulero

1. Per ogni **a** e **n coprimi** vale la seguente relazione

$$a^{\phi(n)} \equiv 1 \pmod{n}$$

cioè $a^{\phi(n)}$ e 1 sono congruenti $\Rightarrow a^{\phi(n)} \pmod{n} = 1 \pmod{n} \Rightarrow a^{\phi(n)} \pmod{n} = 1$

2. Dati due valori interi e positivi **a** e **n** vale la seguente relazione:

$$a^{\phi(n)+1} \equiv a \pmod{n}$$

cioè $a^{\phi(n)+1}$ e a sono congruenti $\Rightarrow a^{\phi(n)+1} \pmod{n} = a \pmod{n} \Rightarrow a^{\phi(n)} * a \pmod{n} = a \pmod{n} \Rightarrow a^{\phi(n)} \pmod{n} * a \pmod{n} = a \pmod{n} \Rightarrow a^{\phi(n)} \pmod{n} = 1$ oppure $a^{\phi(n)+1} \pmod{n} = a$

Esempi sul teorema di Eulero

2. Dati due valori interi e positivi **a** e **n** vale la seguente relazione:

$$a^{\phi(n)+1} \equiv a \pmod{n}$$

$$a = 97, n = 143$$

$$\phi(143) = \phi(11) * \phi(13) = (11 - 1) * (13 - 1) = 10 * 12 = 120$$

quindi

$$a^{\phi(n)+1} \pmod{n} = a \Rightarrow 97^{121} \pmod{143} = 97$$

Teorema di Fermat

1. Dato un numero primo **p** e un numero intero positivo **a** **non divisibile per p**, allora

$$a^{p-1} \equiv 1 \pmod{p}$$

2. Dato un numero primo p e un numero intero positivo a , allora

$$a^p \equiv a \pmod{p}$$

$$\text{oppure } a^p \pmod{p} = a$$

Ad esempio, dato $a = 3$ e $p = 5 \Rightarrow 3^5 \pmod{5} = 3$.

L'utilità del teorema di Fermat risiede nella facilità e velocità dei calcoli svolti con un elaboratore.

Crittografia asimmetrica

Tecniche crittografiche per la protezione dei dati

Libro vol.3 - La crittografia asimmetrica

STUDIARE: L. Lo Russo, E. Bianchi, Sistemi e Reti - Nuova Edizione OPENSCHOOL - Per l'articolazione INFORMATICA degli Istituti Tecnici settore Tecnologico, vol. 3, ed. Hoepli, 2017.

- UdA3 - Tecniche crittografiche per la protezione dei dati
 - Lezione 2 - La crittografia asimmetrica
 - Generalità pp.108,110-113 (*saltare l'esempio di pp.108-109*)
- CLIL Listening - [Assumptions of Encryption and Authentication](#) (*dal minuto 48:43*) del [Prof. Steven Gordon](#) (*il video è utile perché fornisce i concetti generali della crittografia asimmetrica, ed è parte integrante del materiale di studio*)
- UdA3 - Tecniche crittografiche per la protezione dei dati
 - Lezione 2 - La crittografia asimmetrica
 - RSA pp.113-117 (*l'esempio a pp.114-115 del calcolo del MCD usando il metodo di Euclide può essere tralasciato*)
 - **FACOLTATIVO:** chi volesse approfondire il funzionamento dell'algoritmo RSA e i principi matematici che lo fanno funzionare potrà farlo vedendo i seguenti video del [Prof. Steven Gordon](#):
 - [Basics of Primes and Modular Arithmetic](#)
 - [Eulers Theorem, Fermats Theorem and Discrete Logarithms](#)
 - [Public Key Crypto with RSA](#)
 - [RSA Summary and Examples](#)
- Studiare sugli [appunti del docente RSA](#) pp.89-96
- UdA3 - Tecniche crittografiche per la protezione dei dati
 - Lezione 2 - La crittografia asimmetrica
 - Crittografia ibrida pp.117-119
- CLIL Listening - [Public Key Crypto and Digital Signatures](#) del [Prof. Steven Gordon](#) (*il video è utile perché fornisce molteplici esempi di uso della crittografia simmetrica e asimmetrica, un esempio di uso della crittografia asimmetrica per lo scambio delle chiavi simmetriche, chiarisce il funzionamento delle funzioni hash e introduce la firma digitale; inoltre il video è parte integrante del materiale di studio*)

Introduzione alla crittografia asimmetrica

Il **punto debole della crittografia simmetrica** è lo **scambio della chiave**, infatti se questa viene intercettata la crittografia diventa inutile.

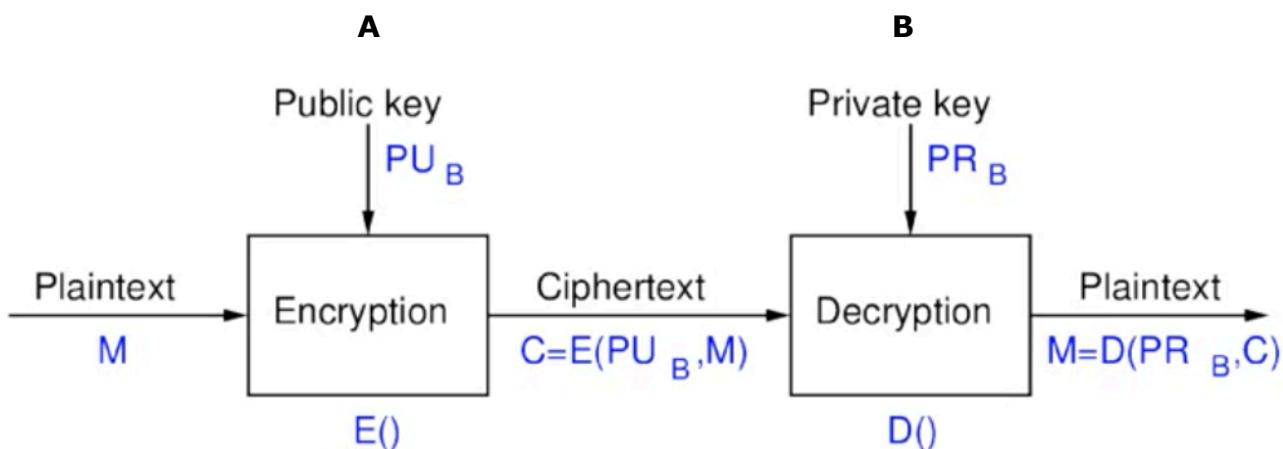
Una svolta nella storia della crittografia venne con la **crittografia asimmetrica**, conosciuta anche come crittografia a coppia di chiavi, crittografia a chiave pubblica/privata o anche solo **crittografia a chiave pubblica**, che prevede l'uso di una **coppia di chiavi associate ad ogni utente**:

1. la **chiave pubblica** è **pubblicamente distribuita** e viene usata per **cifrare** per **garantire segretezza**, mentre è usata per **decifrare** per **garantire autenticazione**;
2. la **chiave privata** che deve rimanere **segreta** ed è usata per **decifrare** per **garantire segretezza**, mentre è usata per **cifrare** per **garantire autenticazione**.

In questo modo si **evitano i problemi** relativi allo **scambio** dell'unica **chiave** utile alla cifratura/decifratura, presenti invece nella **crittografia simmetrica**. Da ciò si capisce che la **crittografia a chiave pubblica** fu una vera rivoluzione nella storia della crittografia.

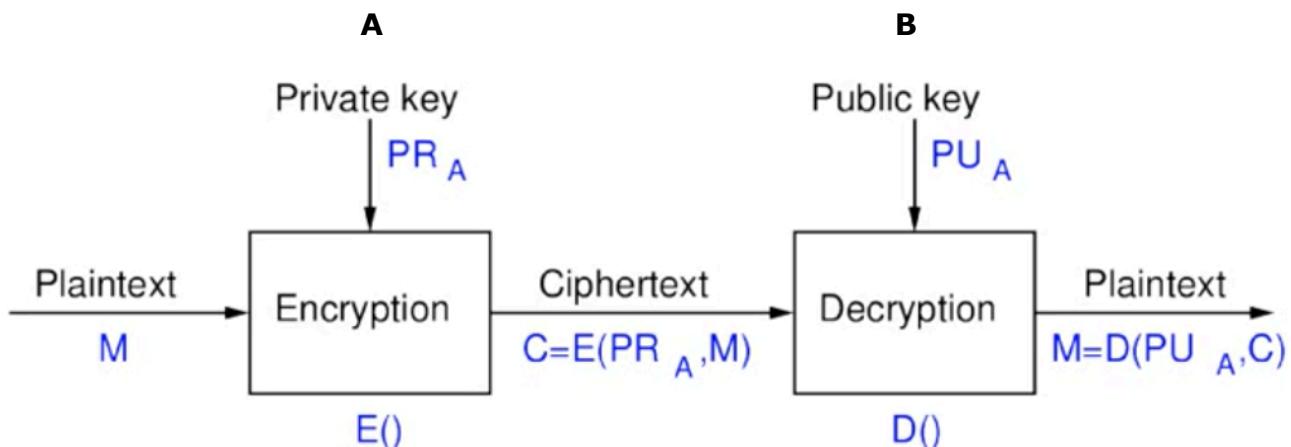
Il seguente schema mostra come vengono usate la chiave pubblica **PU_B** e la chiave privata **PR_B** per **garantire la segretezza**:

- si usa la chiave pubblica **PU_B** per criptare;
- si usa la chiave privata **PR_B** per decriptare;
- solo chi è in possesso della chiave privata **PR_B** può decriptare il messaggio **C**.



Il seguente schema mostra invece come vengono usate la chiave pubblica **PU_B** e la chiave privata **PR_B** per **garantire l'autenticazione**:

- si usa la chiave privata **PR_B** per criptare;
- si usa la chiave pubblica **PU_B** per decriptare;
- solo chi è in possesso della chiave privata **PR_B** può criptare il messaggio **C**.



Nei prossimi paragrafi si analizzerà prima come garantire la segretezza, successivamente verrà illustrato come garantire l'autenticità.

Il principale **svantaggio degli algoritmi a crittografia asimmetrica** sta nella **complessità dei calcoli che rendono poco efficiente la loro implementazione** soprattutto con l'aumentare della lunghezza della chiave: gli algoritmi simmetrici e quelli asimmetrici necessitano di chiavi di lunghezze differenti per raggiungere il medesimo grado di sicurezza teorica con lunghezze "sfavorevoli" per gli algoritmi asimmetrici, come si può vedere dalla seguente tabella:

Simmetrica	Asimmetrica
56	384
64	512
80	768
112	1792
128	2304

Il metodo Diffie-Hellman e lo scambio delle chiavi

Lo **scambio di chiavi Diffie-Hellman** (*Diffie-Hellman key exchange*) è un **protocollo crittografico** che **consente a due entità di stabilire una chiave condivisa e segreta utilizzando un canale di comunicazione insicuro** (pubblico) **senza la necessità che le due parti si siano scambiate informazioni o si siano incontrate in precedenza**. La chiave ottenuta mediante questo protocollo, denominata anche **chiave di sessione**, può essere successivamente impiegata per cifrare le comunicazioni successive tramite uno schema di crittografia simmetrica.

Il protocollo per lo scambio di chiave Diffie-Hellman fu sviluppato da Diffie ed Hellman nel 1976.



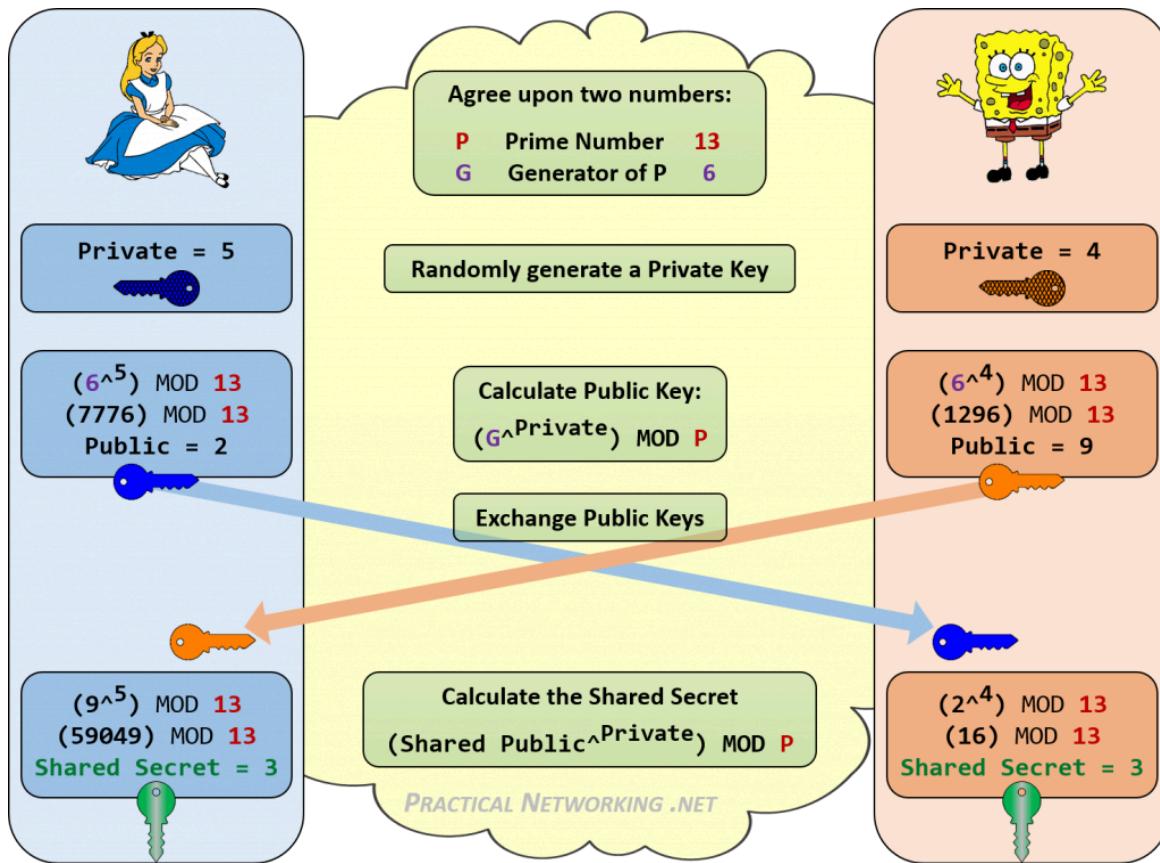
Il protocollo ha due **parametri di sistema p e g** che sono entrambi **pubblici**. Il parametro **p è un numero primo** e il parametro **g** (generalmente chiamato **generatore**) è un intero minore di p ⁵¹.

Supponiamo che Bob e Alice si vogliano accordare su una chiave segreta condivisa usando il protocollo di Diffie-Hellman:

- Alice genera un valore positivo casuale **a** ($< p$) che solo lei conosce e Bob fa altrettanto generando **b** ($< p$);
- entrambi ricaveranno i loro numeri pubblici, da scambiarsi successivamente, nel seguente modo, usando il parametro **p** , il parametro **g** e i loro numeri privati **a** e **b** :
 - la **chiave pubblica** di Alice è determinato da **$A = g^a \text{ mod } p$** ;
 - la **chiave pubblica** di Bob è determinato da **$B = g^b \text{ mod } p$** ;
- Alice e Bob si scambiano i loro valori pubblici **A** e **B** ;

⁵¹ **g** è una radice primitiva (o generatore) di **p** . Una radice primitiva (o generatore) **g** di un numero primo **p** è un numero tale che per ogni **$n \in [1, p-1]$** , **$(g^n \text{ mod } p)$ deve generare un valore diverso per ogni n** .

- Alice calcola la **chiave segreta condivisa K** con il calcolo
 $B^a \text{ mod } p = (g^b)^a \text{ mod } p = K;$
- Bob calcola la **chiave segreta condivisa K** con il calcolo
 $A^b \text{ mod } p = (g^a)^b \text{ mod } p = K.$



In questo modo Alice e Bob hanno generato una **chiave segreta condivisa K senza che si siano scambiati informazioni segrete** o si siano incontrati per effettuare lo scambio delle chiavi. Inoltre entrambi possono **distribuire la propria chiave pubblica a chiunque** voglia scambiare con loro messaggi criptati.

Le due chiavi, pubblica e privata, **sono fra loro correlate ma è estremamente difficile risalire alla seconda conoscendo la prima**. Questa difficoltà è di natura matematica, o meglio, è legata ad un problema matematico⁵² particolarmente difficile da risolvere quando viene scelto un numero primo **p molto grande**. Se non venisse scelto un numero primo sufficiente grande, l'algoritmo risulterebbe facilmente violabile.

Analogamente i valori segreti di **a** e **b**, scelti casualmente, devono anch'essi essere molto grandi e minori di **p**.

⁵² Il problema del logaritmo discreto

Esempio sull'uso di Diffie-Hellman

Supponiamo che due utenti **Alice** e **Bob** vogliano comunicare fra loro per **generare una chiave comune** da usare per future comunicazioni cifrate. Per rendere i calcoli eseguibili facilmente si useranno valori di **p**, **a** e **b** piccoli rispetto alla dimensione che dovrebbero realmente avere.

Alice	Bob
$p = 353$ (numero primo)	
$g = 3$ (generatore $< p$)	
$a = 97$ (scelto a caso $e < p$)	
$A = g^a \text{ mod } p$ $= 3^{97} \text{ mod } 353$ $= 40$ (chiave pubblica di Alice)	

→**p=353, g=3, A=40**→

Alice invia a Bob un messaggio contenente $p = 353$, $g = 3$ e la sua chiave pubblica $A = 40$. Il messaggio è inviato usando un canale non sicuro e quindi potrebbe essere intercettato da un attaccante. Bob ricevuto il messaggio effettuerà i medesimi passaggi di Alice per determinare la sua chiave pubblica.

	$b = 233$ (scelto a caso $e < p$)
	$B = g^b \text{ mod } p$ $= 3^{233} \text{ mod } 353$ $= 248$ (chiave pubblica di Bob)

←**B = 248**←

Bob invierà un messaggio ad Alice contenente la sua chiave pubblica $B = 248$ usando un canale non sicuro che potrebbe essere intercettato da un attaccante.

$K_{\text{Alice}} = B^a \text{ mod } p$ $= 248^{97} \text{ mod } 353$ $= \boxed{\mathbf{160}}$	$K_{\text{Bob}} = A^b \text{ mod } p$ $= 40^{233} \text{ mod } 353$ $= \boxed{\mathbf{160}}$
--	--

19 - Esercizi sull'uso di Diffie-Hellman

Si svolgano i seguenti esercizi usando la tecnica mostrata in precedenza

1.

Alice	Bob
$p = 19$ (numero primo)	
$g = 10$ (generatore $< p$)	
$a = \dots$ (scelto a caso e $< p$)	
$A = g^a \bmod p$ $= \dots$ $= \dots$ (chiave pubblica di Alice)	

→**p=19, g=10, A=...**→

Alice invia a Bob un messaggio contenente $p = 19$, $g = 10$ e la sua chiave pubblica $A = \dots$. Il messaggio è inviato usando un canale non sicuro e quindi potrebbe essere intercettato da un attaccante. Bob ricevuto il messaggio effettuerà i medesimi passaggi di Alice per determinare la sua chiave pubblica.

	$b = \dots$ (scelto a caso e $< p$)
	$B = g^b \bmod p$ $= \dots$ $= \dots$ (chiave pubblica di Bob)

←**B = ...**←

Bob invierà un messaggio ad Alice contenente la sua chiave pubblica $B = \dots$ usando un canale non sicuro che potrebbe essere intercettato da un attaccante.

$K_{\text{Alice}} = B^a \bmod p$ $= \dots$ $= \dots$	$K_{\text{Bob}} = A^b \bmod p$ $= \dots$ $= \dots$
--	--

2. Generare la chiave di sessione condivisa da Alice e Bob usando Diffie-Hellman nel caso vengano scelti $p = 31$ e $g = 3$.
3. Generare la chiave di sessione condivisa da Alice e Bob usando Diffie-Hellman nel caso vengano scelti $p = 19$ e $g = 2$.
4. Generare la chiave di sessione condivisa da Alice e Bob usando Diffie-Hellman nel caso vengano scelti $p = 47$ e $g = 5$.

5. Generare la chiave di sessione condivisa da Alice e Bob usando Diffie-Hellman nel caso vengano scelti $p = 11$ e $g = 2$.
6. Generare la chiave di sessione condivisa da Alice e Bob usando Diffie-Hellman nel caso vengano scelti $p = 23$ e $g = 5$.
7. Generare la chiave di sessione condivisa da Alice e Bob usando Diffie-Hellman nel caso vengano scelti $p = 1193$ e $g = 3$.
8. Generare la chiave di sessione condivisa da Alice e Bob usando Diffie-Hellman nel caso vengano scelti $p = 191$ e $g = 2$.

Determinazione delle chiavi identiche in Diffie-Hellman

Partendo dalle equazioni per la determinazione delle chiavi pubbliche

$$A = g^a \bmod p$$

$$B = g^b \bmod p$$

e le equazioni per la determinazione delle chiavi private

$$K_{Alice} = B^a \bmod p$$

$$K_{Bob} = A^b \bmod p$$

avremo⁵³

$$K_{Alice} = B^a \bmod p = (g^b \bmod p)^a \bmod p = ((g^b)^a \bmod p) \bmod p = (g^b)^a \bmod p = g^{b*a} \bmod p$$

e analogamente

$$K_{Bob} = A^b \bmod p = (g^a \bmod p)^b \bmod p = ((g^a)^b \bmod p) \bmod p = (g^a)^b \bmod p = g^{a*b} \bmod p$$

che per la proprietà commutativa del prodotto posto all'esponente, risultano essere lo stesso numero.

⁵³ Il prodotto modulo m è definito come $(a \bmod m)*(b \bmod m) = (a * b) \bmod m$. Inoltre determinare il modulo di un modulo non cambia il valore finale e quindi $(a \bmod m) \bmod m = a \bmod m$, ad es. $10 \bmod 8 = 2$, ma $2 \bmod 8 = 2$, quindi possiamo scrivere che $(10 \bmod 8) \bmod 8 = 10 \bmod 8 = 2$. È possibile dire che 10 e 2 sono *congruenti modulo 8* perché $(10 \bmod 8) = (2 \bmod 8)$, e si scrive $10 \equiv 2 \pmod{8}$ o in generale se $(a \bmod n) = (b \bmod n)$ allora $a \equiv b \pmod{n}$.

Robustezza dell'algoritmo Diffie-Hellman

L'**attaccante Eve** che volesse intromettersi nella comunicazione tra Alice e Bob per riuscire a scoprire qual è la chiave di sessione K_{Alice} , dovrebbe cercare di risolvere la seguente equazione

$$K_{Alice} = B^a \bmod p$$

ad esempio $K_{Alice} = 248^a \bmod 353$

che è un'equazione in due incognite non risolvibile.

Tentando un altro approccio, invece di cercare di scoprire direttamente K_{Alice} , Eve potrebbe tentare di scoprire **a** nel seguente modo

$$A = g^a \bmod p$$

ad esempio $248 = 3^a \bmod 353$

che sicuramente è un'equazione risolvibile avendo una sola incognita.

L'inverso per determinare l'esponente modulo **n** è il **logaritmo discreto**, che quando applicato a valori di **p** molto grandi non è risolvibile in tempi brevi.

Eve potrebbe tentare con un attacco **brute-force**, usando una tabella in cui vengano riportate tutte le potenze di **g** modulo **p**, ma una tale tabella dovrebbe avere **p - 1** righe, ma se **p** viene scelto sufficiente grande, ad esempio lungo 2048 cifre, la tabella dovrebbe avere un numero di righe superiore agli atomi dell'universo. Attualmente sarebbe impossibile anche solo pensare di memorizzare una tabella di queste dimensioni. Quindi se **p** viene scelto sufficientemente grande, Eve non ha attualmente modo di recuperare la chiave condivisa da Alice e Bob.

La gestione di valori così grandi potrebbe indurre a pensare che anche Alice e Bob possano avere problemi a trattarli per calcolare delle potenze che produrrebbero numeri ancora più grandi. Il fatto che i calcoli fatti da Alice e Bob siano svolti tutti modulo **p**, permette di usare il **metodo dei quadrati ripetuti** che consente di suddividere il calcolo della potenza modulo **p** in tanti calcoli separati costituiti di potenze più piccole, riducendone complessivamente la complessità.

Esempio di applicazione del metodo dei quadrati ripetuti [leggere]

Vediamone un esempio usando numeri relativamente piccoli, ma che comunque creerebbero dei problemi con una normale calcolatrice.

Supponiamo di voler effettuare il seguente calcolo

$$7^{29} \pmod{17}$$

la potenza 29 può essere suddivisa in una somma di potenze

$$29 = 16 + 8 + 4 + 1 = 2^4 + 2^3 + 2^2 + 2^0$$

che corrisponde alla rappresentazione in binario del numero $(29)_{10} = (11101)_2$.

Applichiamo ora il metodo dei quadrati ripetuti della base 7 modulo 17 e utilizzando le proprietà del modulo:

$$7^1 \pmod{17} = 7$$

$$\begin{aligned} 7^2 \pmod{17} &= 7^1 * 7^1 \pmod{17} = \\ ((7^1 \pmod{17}) * (7^1 \pmod{17})) \pmod{17} &= \\ 7 * 7 \pmod{17} &= 49 \pmod{17} = 15 \end{aligned}$$

$$\begin{aligned} 7^4 \pmod{17} &= 7^2 * 7^2 \pmod{17} = \\ ((7^2 \pmod{17}) * (7^2 \pmod{17})) \pmod{17} &= \\ 15 * 15 \pmod{17} &= 4 \end{aligned}$$

$$\begin{aligned} 7^8 \pmod{17} &= 7^4 * 7^4 \pmod{17} = \\ ((7^4 \pmod{17}) * (7^4 \pmod{17})) \pmod{17} &= \\ 4 * 4 \pmod{17} &= 16 \end{aligned}$$

$$\begin{aligned} 7^{16} \pmod{17} &= 7^8 * 7^8 \pmod{17} = \\ ((7^8 \pmod{17}) * (7^8 \pmod{17})) \pmod{17} &= \\ 16 * 16 \pmod{17} &= 1 \end{aligned}$$

Quindi

$$\begin{aligned} 7^{29} \pmod{17} &= 7^{16} * 7^8 * 7^4 * 7^1 \pmod{17} = \\ 1 * 16 * 4 * 7 \pmod{17} &= 448 \pmod{17} = 6. \end{aligned}$$

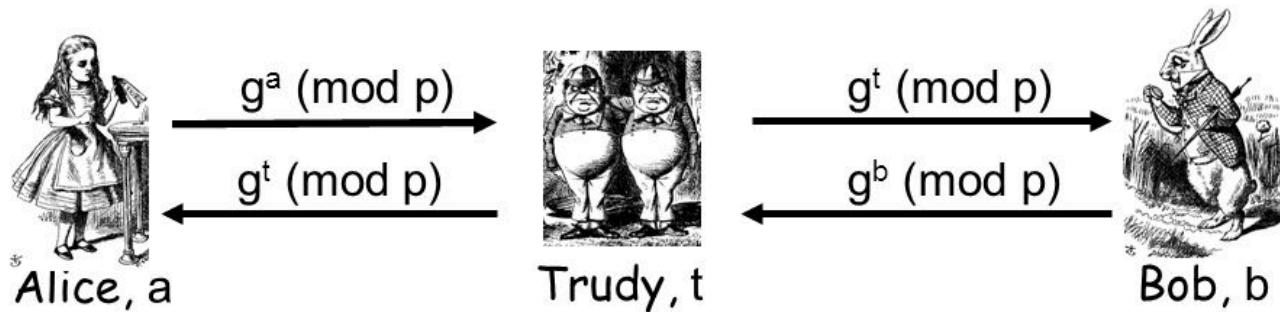
In questo procedimento i numeri non supereranno mai il valore $16^2 = 256$, tranne all'ultimo passo del processo. In ogni caso, se un numero fosse troppo grande sarebbe possibile ridurlo applicandogli il mod 17 prima di effettuare la moltiplicazione finale. Quindi con il **metodo dei quadrati ripetuti**, Alice e Bob possono comunque calcolare i valori di cui necessitano.

Diffie-Hellman e attacco Man-in-the-middle

Questo tipo di attacco permette ad un **attaccante** (Trudy) di **accedere ai messaggi** che Alice e Bob si scambiano **senza dover risolvere il problema del logaritmo discreto**. Infatti l'**attaccante non cerca di scoprire la chiave segreta K** condivisa da Alice e Bob, ma bensì **si intromette nella comunicazione** condividendo una chiave segreta diversa con Alice e Bob e spacciandosi per loro durante la comunicazione.

Si supponga che quando Alice invia a Bob il messaggio per comunicargli i valori di **p**, **g** e **A**, Trudy intercetti il messaggio inviando a Bob una sua chiave pubblica **T'**. Analogamente Trudy invierà ad Alice la propria chiave pubblica **T''**.

In questo modo Trudy intercetterà tutti i messaggi che Alice e Bob si scambieranno, decriptando e ri-criptando i messaggi dopo averne fatto ciò che desidera.



- Trudy shares secret $g^{at} \pmod{p}$ with Alice
- Trudy shares secret $g^{bt} \pmod{p}$ with Bob
- Alice and Bob don't know Trudy exists!

Per **evitare questo** tipo di **attacco** è necessario affiancare il protocollo Diffie-Hellman con un **metodo di firma digitale dei messaggi**.

Esempio di attacco Man-in-the-middle su Diffie-Hellman

Alice

$$p = 19, g = 3, a = 10 \Rightarrow A = 3^{10} \bmod 19 = 16$$

Alice invia a Bob i valori $p = 19, g = 3$ e $A = 16$

Alice: $p=19, g=3, \Rightarrow$ **Trudy** \Rightarrow **Bob**
A=16

Trudy

Il messaggio viene intercettato da Trudy che sceglierà $t_{\text{Trudy-Bob}} = 2$ come numero segreto casuale minore di $p \Rightarrow T' = 3^2 \bmod 19 = 9$ che sarà inviato a Bob insieme a $p = 19$ e $g = 3$.

Alice: $p=19, g=3, \Rightarrow$ **Trudy:** $p=19, g=3, T'=9 \Rightarrow$ **Bob**
A=16

$$t_{\text{Trudy-Bob}} = 2$$

$$T' = 9$$

Bob

Bob sceglierà il proprio valore $b = 11$ e calcolerà la chiave pubblica $B = 3^{11} \bmod 19 = 10$ e lo invierà a Trudy pensando che sia Alice. Nel frattempo calcolerà la chiave segreta condivisa usando i parametri che gli sono stati inviati da Trudy, pensando che siano quelli di Alice $K_{\text{Bob}} = 9^{11} \bmod 19 = 5$.

Alice: $p=19, g=3, \Rightarrow$ **Trudy:** $p=19, g=3, T'=9 \Rightarrow$ **Bob**
A=16

$$t_{\text{Trudy-Bob}} = 2$$

$$T' = 9$$

Trudy \Leftarrow **Bob:** $B=10$

$$K_{\text{Bob}} = 5$$

Trudy

Trudy ricevendo la chiave pubblica di Bob calcolerà la chiave segreta per scambiare i messaggi con Bob, $K_{\text{Trudy-Bob}} = 10^2 \bmod 19 = 5$ che è la stessa di Bob.

Trudy dovrà ora scegliere un numero segreto minore di p da usare con Alice, $t_{\text{Trudy-Alice}} = 7$, calcolando così la chiave pubblica da inviare ad Alice, $T'' = 3^7 \pmod{19} = 2$.

Infine Trudy calcolerà la chiave segreta da usare con Alice, $K_{\text{Trudy-Alice}} = 16^7 \pmod{19} = 17$.

Alice: $p=19$, $g=3$, \Rightarrow **Trudy:** $p=19$, $g=3$, $T'=9$ \Rightarrow **Bob**

$$t_{\text{Trudy-Bob}} = 2$$

$$T' = 9$$

Trudy <===== **Bob:** $B=10$

$$K_{\text{Trudy-Bob}} = 5 \quad K_{\text{Bob}} = 5$$

$$t_{\text{Trudy-Alice}} = 7$$

$$T'' = 2$$

$$K_{\text{Trudy-Alice}} = 17$$

Alice

Alice ricevendo la chiave pubblica da Trudy, pensando che sia Bob, calcolerà la propria chiave segreta $K_{\text{Alice}} = 2^{10} \pmod{19} = 17$ che è la stessa di Trudy.

In questo modo Alice e Bob pensando di comunicare direttamente in modo cifrato, finiranno con il comunicare con Trudy che farà ciò che vorrà con i messaggi scambiati.

Alice: $p=19$, $g=3$, \Rightarrow **Trudy:** $p=19$, $g=3$, $T'=9$ \Rightarrow **Bob**

$$t_{\text{Trudy-Bob}} = 2$$

$$T' = 9$$

Alice <===== **Trudy** <===== **Bob:** $B=10$

$$K_{\text{Alice}} = 17 \quad K_{\text{Trudy-Bob}} = 5 \quad K_{\text{Bob}} = 5$$

$$t_{\text{Trudy-Alice}} = 7$$

$$T'' = 2$$

$$K_{\text{Trudy-Alice}} = 17$$

Quando avverrà lo scambio di messaggi criptati usando le chiavi di sessioni lo scenario che si presenterà sarà il seguente:

Alice: $c = E(17, m \Rightarrow Trudy:c' = E(5, m) \Rightarrow Bob$
)

$m = D(17, c) \qquad \qquad \qquad m = D(5, c')$

Alice $\leq Trudy:c''' = E(17, m' \leq Bob:c'' = E(5, m')$
)

$m' = D(17, c''') \qquad \qquad \qquad m' = D(5, c'')$

CLIL - Homework

Study all the topics your teacher spoke about and watch the following videos.

1. [What is a trapdoor function?](#)
2. [What is the basis for most modern cryptography?](#)
3. [What is asymmetric encryption?](#)
4. [How do two parties exchange keys to communicate securely?](#)
5. [What is a brute force attack?](#)

RSA

In crittografia la sigla **RSA** indica un algoritmo di crittografia asimmetrica, inventato nel 1977 da Ronald Rivest, Adi Shamir e Leonard Adleman utilizzabile per cifrare o firmare informazioni.



Il sistema di crittografia si basa sull'esistenza di due chiavi distinte, che vengono usate per cifrare e decifrare. Se la prima chiave viene usata per la cifratura, la seconda deve necessariamente essere utilizzata per la decifratura e viceversa. La questione fondamentale è che **nonostante le due chiavi siano fra loro dipendenti, non è computazionalmente possibile risalire dall'una all'altra**, in modo che se anche si è a conoscenza di una delle due chiavi, non si possa risalire all'altra, garantendo in questo modo l'integrità della crittografia.

RSA è basato sulla **elevata complessità computazionale della fattorizzazione in numeri primi**⁵⁴, oltre che del calcolo **del logaritmo discreto**⁵⁵ e della determinazione **della funzione di Eulero**⁵⁶; quando si lavora con numeri molto grandi, dell'ordine di centinaia di cifre, la risoluzione di uno qualsiasi di questi problemi diventa improponibile.

⁵⁴ Il **teorema fondamentale dell'aritmetica** afferma che: *ogni numero naturale maggiore di 1 o è un numero primo o si può esprimere come prodotto di numeri primi. Tale rappresentazione è unica, se si prescinde dall'ordine in cui compaiono i fattori.*

⁵⁵ Il **logaritmo discreto** è l'inverso per la determinazione dell'esponente di una potenza, modulo n.

⁵⁶ La **funzione di Eulero $\phi(n)$** , detta **toziente**, è una funzione che, per ogni intero positivo n, è **definita come il numero degli interi compresi tra 1 e n e minori di n che sono coprimi con n**.

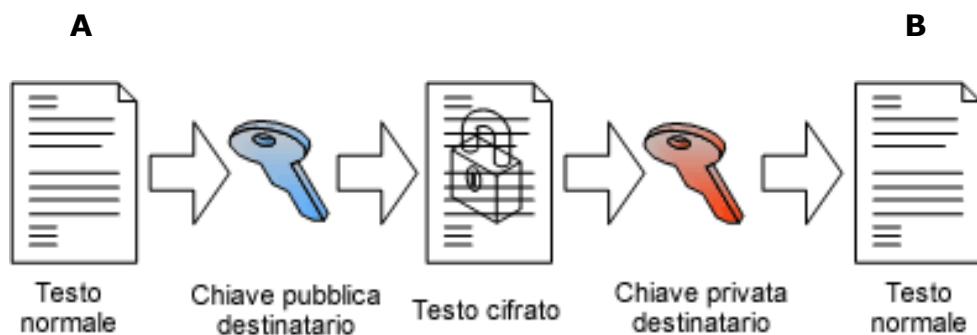
In breve dati due numeri primi **p** e **q** molto grandi, è **facile calcolare**

$$n = p * q$$

ma è **computazionalmente troppo lungo trovare** i due fattori **p** e **q**, noto **n**.

Concetto di base di RSA

Per semplificare il funzionamento immaginiamo che **A** debba spedire un messaggio segreto a **B**.



- B** determina due chiavi, una privata e una pubblica. Le **chiavi** sono delle **coppie di numeri**.
- B** invia la propria chiave pubblica ad **A**. Chiunque può vedere questa chiave.
- A** usa questa chiave per cifrare il messaggio (**garanzia di confidenzialità**).
- A** manda il messaggio cifrato a **B**, chiunque può vederlo, ma non decifrarlo.
- B** riceve il messaggio e, utilizzando la chiave privata che solo lui conosce, lo decifra.

Immaginiamo che la **chiave pubblica** sia **PU** = (e, n) costituita dalla coppia di numeri **e** e **n**, e la **chiave privata** sia **PR** = (d, n) costituita dalla coppia di numeri **d** e **n**⁵⁷. Per **cifrare** un messaggio **m** che deve essere minore di **n** (**m < n**) si utilizzerà la **chiave pubblica** **PU** = (e, n) e l'algoritmo

$$\text{Cifratura: } c = m^e \bmod n$$

mentre per **decifrare** un messaggio cifrato **c** si utilizzerà la **chiave privata** **PR** = (d, n) e l'algoritmo

$$\text{Decifrazione: } m = c^d \bmod n$$

Si noti che l'unico elemento veramente segreto nelle due chiavi **PU** e **PR** è il valore **d**.

⁵⁷ Più avanti sarà spiegato come determinare i valori **e** e **d**.

Esempio di criptazione e decrittazione usando RSA

Il messaggio viene rappresentato come un valore intero, e ciò è fattibile perché un messaggio è una sequenza di bit di cui si può trovare l'equivalente decimale.

Messaggio: ' m ' = $(10010001)_2 = (145)_{10}$

quindi crittografare un messaggio equivale a cifrare il suo corrispondente intero decimale.

Vediamo un esempio di crittografia e decrittografia usando **RSA**, e supponiamo che si debba crittografare il messaggio $m = 7$, usando i valori $n = 55$ ed $e = 3$ che costituiranno la chiave pubblica $PU = (e, n) = (3, 55)$

$$c = m^e \bmod n = 7^3 \bmod 55 = 13$$

Per decrittografare il messaggio cifrato **13** usiamo la chiave privata $PR = (d, n) = (27, 55)$

$$m = c^d \bmod n = 13^{27} \bmod 55 = 7$$

Quindi utilizzando la chiave pubblica e quella privata che sono create usando una correlazione matematica, è possibile cifrare e decifrare un messaggio.

La generazione delle chiavi

1. Si scelgono a caso due numeri primi diversi p e q abbastanza **grandi** da garantire la sicurezza dell'algoritmo (sono consigliati almeno 4096 bit); **p e q devono rimanere privati;**
2. si calcola il loro prodotto $n = p * q^{58}$, chiamato modulo (dato che tutta l'aritmetica seguente è modulo n);
3. si calcola il prodotto $z = (p-1) * (q-1)^{59}$
4. si determina la **chiave pubblica PU = (e, n)** scegliendo un numero e tale per cui :
 - a. $1 < e < n$
 - b. e deve essere **coprimo con z**⁶⁰
5. si determina la **chiave privata PR = (d, n)** scegliendo un numero d tale per cui $(e*d) \bmod z = 1^{61}$

Da questi valori si determina la **chiave pubblica PU = (e, n)** e la **chiave privata PR = (d, n)**. Dopo la determinazione delle due chiavi i due numeri primi p e q possono anche essere distrutti, ma spesso vengono mantenuti insieme alla chiave privata.

La forza dell'algoritmo sta nel fatto che per calcolare d da e o viceversa, non basta la conoscenza di n , ma serve il numero $z = (p-1) * (q-1)$ e che il suo calcolo richiede tempi molto elevati; infatti fattorizzare in numeri primi, o determinare il toziente di un numero, sono operazioni che richiedono molto tempo se si scelgono dei valori numerici costituiti da molte cifre. Solo conoscendo la fattorizzazione di n è possibile trovare il valore delle chiavi. Infatti conoscendo la fattorizzazione di n è possibile calcolare $z = (p-1) * (q-1)$ e calcolare $d = e^{-1} \bmod z$ usando l'algoritmo esteso di Euclide.

Inoltre la sicurezza di **RSA** si basa sul fatto che la funzione di cifratura $c = m^e \bmod n$ è una funzione "**one-way**" che è computazionalmente difficile da invertire per un nemico che volesse decifrare un messaggio.

⁵⁸ La fattorizzazione di n è segreta e solo chi sceglie i due numeri primi, p e q la conosce. Inoltre se i numeri primi p e q vengono scelti molto grandi, diventa molto difficile risalire ad essi conoscendo solo n .

⁵⁹ Il valore di $z = (p-1)*(q-1)$ rappresenta la **funzione di Eulero $\phi(n)$** , o **toziente**, che **rappresenta il numero degli interi compresi tra 1 e n e minori di n che sono coprimi con n**.

⁶⁰ Due numeri e e z si dicono **coprimi** se non hanno fattori in comune, cioè se $MCD(e, z) = 1$.

⁶¹ **$(e*d) \bmod z = 1$** implica che $d \equiv e^{-1} \bmod z$. Per il **teorema cinese del resto**, un modo semplice per determinare d è il seguente $d = (z * k + 1)/e$ dove k è un intero positivo.

Esempio di generazione delle chiavi di RSA

Vediamo un esempio di generazione delle due chiavi dell'algoritmo RSA. Per esigenze didattiche verranno presi come numeri primi **p** e **q** due valori piccoli, ma si ricordi che nella realtà i due numeri primi dovranno essere costituiti da un numero molto elevato di cifre per garantire la sicurezza dell'algoritmo.

1. **p = 5, q = 11**
2. **n = p * q = 5 * 11 = 55**
z = (p-1)*(q-1) = 4 * 10 = 40
3. Trovare **1 < e < 55** tale che **MCD(e, 40) = 1**
 40 = 5 * 2³
 per e = 2 => MCD(2, 40) = 2 => NO
 per **e = 3 => MCD(3, 40) = 1 => SI**⁶²
4. Trovare **d** tale che **(3 * d) mod 40 = 1**
 per d = 2 => (2*3) mod 40 = 6 => NO
 per d = 3 => (3*3) mod 40 = 9 => NO

 per **d = 27 => (3*27) mod 40 = 81 mod 40 = 1 => SI**

Chiave pubblica = (3, 55)

Chiave privata = (27, 55)

La determinazione del valore **d** richiede molto tempo, ma questo valore può essere determinato più velocemente usando il **teorema Cinese del resto** che permette di determinarlo usando la seguente formula

$$d = (z * k + 1)/e$$

con **k intero positivo**

e **d** tale che **(e*d) mod z = 1**

quindi procedendo per passi successivi si ottiene:

- **k = 1 => d = (40 * 1 + 1)/3 => d = 13**, e controllando che il valore vada bene si dovrà avere $(e * d) \text{ mod } z = 1 \Rightarrow (3 * 13) \text{ mod } 40 = 39 \Rightarrow \text{NO}$
- **k = 2 => d = (40 * 2 + 1)/3 => d = 27**, e controllando che il valore vada bene si dovrà avere $(e * d) \text{ mod } z = 1 \Rightarrow (3 * 27) \text{ mod } 40 = 1 \Rightarrow \text{SI}$

⁶² Si potrebbe continuare a cercare un qualsiasi altro valore minore di n = 55 che sia coprimo di z = 40.

Esempio sull'uso di RSA

Generazione delle chiavi

- Si scelgano dei due numeri primi diversi: $p = 17$, $q = 11$
- Si moltiplichino i due numeri primi: $n = p * q = 17 * 11 = 187$
- Si calcoli la funzione toziente⁶³ di $n = 187$:

$$z = (p - 1) * (q - 1) = 16 * 10 = 160$$
- Si scelga un valore di e tale che $\text{MCD}(e, z) = 1$ cioè siano coprimi e $1 < e < n$:

$$2^{\text{NO}}, 3^{\text{SI}}, 4^{\text{NO}}, 5^{\text{NO}}, 6^{\text{NO}}, 7^{\text{SI}}, 8^{\text{NO}}, 9^{\text{SI}}, 10^{\text{NO}}, 11^{\text{SI}}, \dots$$

Può essere scelto il valore $e = 7$
- Si determini un valore di d tale che $(e * d) \bmod z = 1$ o per il teorema Cinese del resto $d = (z * k + 1)/e$, con k intero positivo:

$$k = 1 \Rightarrow d = (160 * 1 + 1)/7 \Rightarrow d = 23 \Rightarrow (7 * 23) \bmod 160 = 1 \Rightarrow \text{Sì}$$
- Quindi $\text{PU} = (e, n) \Rightarrow \text{PU} = (7, 187)$, $\text{PR} = (d, n) \Rightarrow \text{PR} = (23, 187)$

I valori che devono assolutamente rimanere **segreti** sono **p**, **q**, **d** e **z**, mentre sono **pubblici** i valori **n**, **e** e gli **algoritmi** usati per criptare e decriptare.

Si supponga di dover effettuare una **comunicazione criptata** tra Alice e Bob:

Alice	Bob
<i>Generazione delle chiavi</i>	
$\text{PU}_A = (7, 187)$ $\text{PR}_A = (23, 187)$ <i>Alice invia a Bob la sua chiave pubblica</i> $\text{PU}_B = (5, 299)$	$\text{PU}_B = (5, 299)$ $\text{PR}_B = (53, 299)$ <i>Bob invia ad Alice la sua chiave pubblica</i> $\text{PU}_A = (7, 187)$
<i>Alice vuole inviare il messaggio m = 15 criptato a Bob. La dimensione del messaggio m deve essere più piccola di n = 299.</i>	
$c = E(\text{PU}_B, m)$ $c = m^e \bmod n$ $c = 15^5 \bmod 299$ $c = 214$	
<i>Bob riceve il messaggio criptato c = 214</i>	
	$c = 214$ $m' = D(\text{PR}_B, c)$ $m' = c^d \bmod n$ $m' = 214^{53} \bmod 299$ $m' = 15$
<i>Bob ha decriptato correttamente il messaggio m = 15</i>	

⁶³ $\phi(n) = \phi(p) * \phi(q)$, dato che p e q sono primi $\Rightarrow \phi(n) = (p - 1) * (q - 1)$. Per brevità nei libri di testo non universitari viene semplicemente indicato $z = (p - 1) * (q - 1)$.

Se Alice volesse inviare a Bob un messaggio, ad esempio *ciao*, si dovrebbero convertire i singoli caratteri in codice ASCII:

c	01100011
i	01101001
a	01100001
o	01101111

Il **messaggio in chiaro** sarà: 01100011011010010110000101101111

La sequenza di bit viene suddivisa in blocchi di g bit in modo tale che g sia il più grande numero tale che $2^g < n$, cioè $2^g < 299$, quindi $g = 8$ ($2^8 = 256$).

Dividiamo il messaggio in blocchi 8 bit: 01100011 01101001 01100001 01101111

Cifratura del messaggio

Blocco	In decimale	Blocco cifrato in decimale $P_U_B=(5,299)$	Blocco cifrato in binario
01100011	99	$99^5 \bmod 299 = 86$	01010110
01101001	105	$105^5 \bmod 299 = 27$	00011011
01100001	97	$97^5 \bmod 299 = 158$	10011110
01101111	111	$111^5 \bmod 299 = 11$	00001011

Il **messaggio cifrato** è: 01010110000110111001111000001011

Decifratura del messaggio

Blocco cifrato in binario	In decimale	Blocco decifrato in decimale $P_R_B=(53,299)$	Blocco decifrato in binario
01010110	86	$86^{53} \bmod 299 = 99$	01100011
00011011	27	$27^{53} \bmod 299 = 105$	01101001
10011110	158	$158^{53} \bmod 299 = 97$	01100001
00001011	11	$11^{53} \bmod 299 = 111$	01101111

Il messaggio decifrato da Bob corrisponde esattamente a quello inviato da Alice.

Esercizi sull'uso di RSA

1. Generare la chiave pubblica e segreta dati i numeri primi $p = 13$ e $q = 23$ e criptare e decriptare un messaggio a vostra scelta.
2. Date le chiavi pubbliche e private $PU = (7, 33)$ e $PR = (3, 33)$, si cifri e si decifri il messaggio $m = 15$
3. Generare la chiave pubblica e segreta dati i numeri primi $p = 7$ e $q = 17$ e criptare e decriptare un messaggio a vostra scelta.
4. Date le chiavi pubbliche e private $PU = (5, 65)$ e $PR = (29, 65)$, si cifri e si decifri il messaggio $m = 7$
5. Generare la chiave pubblica e segreta dati i numeri primi $p = 61$ e $q = 53$ e criptare e decriptare un messaggio a vostra scelta.
6. Date le chiavi pubbliche e private $PU = (17, 3233)$ e $PR = (2753, 3233)$, si cifri e si decifri il messaggio $m = 123$
7. Generare la chiave pubblica e segreta dati i numeri primi $p = 47$ e $q = 71$ e criptare e decriptare un messaggio a vostra scelta.
8. Date le chiavi pubbliche e private $PU = (79, 3337)$ e $PR = (1019, 3337)$, si cifri e si decifri il messaggio $m = 688$

Esercizi sulla crittografia asimmetrica

Nei seguenti esercizi si farà riferimento alla seguente notazione:

- $C = E(PU_x, P)$ con la quale si identifica un messaggio cifrato C ottenuto applicando sul testo in chiaro P un algoritmo crittografico a chiave asimmetrica $E()$ che usa la chiave pubblica PU_x dell'utente X .
- $C = E(PR_x, P)$ con la quale si identifica un messaggio cifrato C ottenuto applicando sul testo in chiaro P un algoritmo crittografico a chiave asimmetrica $E()$ che usa la chiave privata PR_x dell'utente X .

Esercizi

1. Si supponga che l'utente Y riceva il messaggio cifrato $C = E(PU_x, P)$. L'utente Y può decriptare C per ottenere P ? Si motivi la risposta.
2. Si supponga che l'utente X riceva il messaggio cifrato $C = E(PU_x, P)$. L'utente X può decriptare C per ottenere P ? Si motivi la risposta.
3. Si supponga che l'utente Y riceva il messaggio cifrato $C = E(PR_x, P)$. L'utente Y è in grado di conoscere il testo in chiaro P ? E se il messaggio crittografato C fosse ricevuto dall'utente Z , potrebbe ottenere il testo in chiaro P ? Cos'altro può essere dedotto dal ricevimento di C ? Motivare tutte le risposte.
4. Si supponga che l'utente Y riceva il seguente messaggio crittografato:
 $C = E(PU_y, E(PR_x, P))$
Si spieghi se Y potrà decrittografare C per ottenere P e cos'altro potrà dedurre Y dal ricevimento di questo messaggio.
Si supponga inoltre che un utente malevolo Z intercetti il messaggio crittografato C , si spieghi se l'utente Z è in grado di ottenere il messaggio in chiaro P .
5. Con le conoscenze acquisite finora, spiegare se la crittografia asimmetrica può essere usata per l'autenticazione.

Autenticazione e affidabilità

Tecniche crittografiche per la protezione dei dati

Libro vol.3 - Certificati e firma digitale

STUDIARE: L. Lo Russo, E. Bianchi, Sistemi e Reti - Nuova Edizione OPENSCHOOL - Per l'articolazione INFORMATICA degli Istituti Tecnici settore Tecnologico, vol. 3, ed. Hoepli, 2017.

- Uda3 - Tecniche crittografiche per la protezione dei dati
 - Lezione 3 - Certificati e firma digitale
 - I sistemi di autenticazione pp.120-122
 - Firme digitali⁶⁴ pp.123-125
- CLIL Listening - [Public Key Crypto and Digital Signatures](#) (*dal minuto 48:32*) del [Prof. Steven Gordon](#) (*il video è utile perché chiarisce il funzionamento delle funzioni hash e introduce la firma digitale, ed è parte integrante del materiale di studio*)
- Uda3 - Tecniche crittografiche per la protezione dei dati
 - Lezione 3 - Certificati e firma digitale
 - I certificati digitali pp.125-128
 - Riferimenti normativi pp.128-129

⁶⁴ Per approfondimenti sulle funzioni hash, MD5 e SHA si veda [Funzioni di hash](#) - [Wikiversità](#)

Codice di autenticazione dei messaggi (crittografia simmetrica)

L'autenticazione dei messaggi utilizzando un **Message Authentication Code (MAC)** prevede l'aggiunta al messaggio di un piccolo blocco di dati **T** di lunghezza fissa, derivato dal messaggio stesso **M**, utilizzando una chiave **K** conosciuta solo dalle parti comunicanti (crittografia simmetrica). Il messaggio **M** può avere qualsiasi lunghezza, mentre il blocco di dati **T** di lunghezza fissa è determinato da una funzione hash che usa una chiave **K**, indicata con l'acronimo **MAC**. La funzione **MAC** è simile a quelle usate per criptare l'intero messaggio, ma non è necessariamente reversibile, e risulta comunque più semplice da progettare rispetto ad una funzione di criptazione.

$$T = \text{MAC}(K, M)$$

- **M** = messaggio in input
- **MAC** = funzione MAC
- **K** = chiave segreta lunga *k* bit, condivisa fra le parti comunicanti
- **T** = codice di autenticazione del messaggio (*Message Authentication Code - MAC*) lungo *n* bit (detto anche *tag*)

La funzione **MAC** deve essere tale per cui sia molto semplice determinare $T = \text{MAC}(K, M)$, ma sia invece molto difficile determinare **M** dati **T** e **K**, cioè sia difficile trovare un messaggio alternativo in grado di replicare il medesimo codice di autenticazione **T** o derivare la chiave **K** usata dalle parti comunicanti.

Quindi un attaccante che modifichi il messaggio originale **M** fallirà se non riuscirà a riprodurre esattamente **T**, non conoscendo la chiave **K** usata. Inoltre, la funzione **MAC** deve essere costruita in modo tale che sia estremamente difficile trovare un messaggio **M'** che fornisca il medesimo **T** determinato usando **M**, anche se non impossibile. Infatti la funzione **MAC** deve soddisfare il più possibile i seguenti requisiti:

1. $T = \text{MAC}(K, M) \text{ diverso } T' = \text{MAC}(K, M')$

cioè la funzione **MAC** calcolata su due diversi messaggi **M** e **M'**, usando la stessa chiave **K**, deve produrre due tag **T** e **T'** diversi;

2. $T = \text{MAC}(K, M) \text{ diverso } T' = \text{MAC}(K', M)$

cioè la funzione **MAC** calcolata sullo stesso messaggio **M** usando due chiavi **K** e **K'** diverse deve produrre due tag **T** e **T'** diversi.

Purtroppo il **requisito 1 non è sempre vero** proprio perché la funzione **MAC** determina un tag di dimensione piccola e di lunghezza fissa, derivato da messaggi di dimensione variabile; quindi può succedere che due messaggi

producano il medesimo *tag*. **La bontà di una funzione MAC deriva anche dalla difficoltà del verificarsi di questa situazione.**

I tentativi di **attacco** sono di tipo **brute force** sui due elementi di dimensione fissa, la chiave **K**, lunga k bit, e il *tag* **T**, lungo n bit. Un attacco **brute force** richiede rispettivamente uno sforzo pari a 2^k per la chiave **K** e 2^n per il *tag* **T**. Normalmente viene scelto l'attacco che minimizza lo sforzo, quindi la dimensione della chiave **K** e del *tag* **T** sono fondamentali per rendere impraticabile questo tipo di attacco.

Altri tipi di **attacchi** si basano **sulle caratteristiche dello specifico algoritmo** che implementa la funzione **MAC**, anche se generalmente **questi algoritmi vengono considerati sicuri**.

Infine si consideri che se venisse **richiesto** sia la **confidenzialità**, sia l'**autenticazione** del messaggio, il procedimento da seguire prevederebbe l'**invio** di

$$E(K, M \parallel T)$$

cioè prima si calcola il *tag* **T** = **MAC**(**K**, **M**) e lo si concatena al messaggio **M** (**M**||**T**), poi si cripta tutto usando la chiave **K** (**E**(**K**, **M**||**T**)).

Se invece bastasse **solo l'autenticazione** del messaggio **senza** richiedere anche la **confidenzialità**, il procedimento da seguire prevederebbe l'**invio** di

$$M \parallel E(K, T)$$

cioè si calcola il *tag* **T** del messaggio **M** (**T** = **MAC**(**K**, **M**)), lo si cripta (**E**(**K**, **T**)) e lo si concatena al messaggio **M** per l'invio (**M**||**E**(**K**, **T**)).

Chiaramente l'ultimo procedimento risulta computazionalmente meno oneroso del primo in quanto non è richiesta la criptazione dell'intero messaggio.

L'**autenticazione dei messaggi** utilizzando un **Message Authentication Code** (MAC) permette a due soggetti comunicanti di verificare la provenienza dei messaggi reciprocamente, ma **non permette ad un terzo soggetto di individuare in modo univoco chi ha originato il messaggio** perché solo i due soggetti comunicanti condividono la medesima chiave **K** (crittografia simmetrica).

Per poter **garantire l'univocità della generazione di un messaggio da parte di un singolo soggetto**, e quindi raggiungere in modo inequivocabile e verificabile da chiunque anche gli obiettivi della **non ripudiabilità** e dell'**autenticazione**, deve essere usata la **crittografia asimmetrica applicata alla firma digitale**.

La firma digitale (crittografia asimmetrica)

Nella **firma digitale** viene **usata una funzione hash per generare un blocco di dati di lunghezza fissa a partire dal messaggio M**, (**H(M)**), che viene successivamente **criptato utilizzando la chiave privata** del mittente, **generando**

la firma digitale ($S = E(PR_A, H(M))$), da attaccare al messaggio per la spedizione ($M \parallel S$).

Il destinatario **verificherà l'autenticità del messaggio decriptando** inizialmente **la firma digitale usando la chiave pubblica del mittente** ($h = D(PU_A, S)$), **ricalcolando l'hash del messaggio** ricevuto $H(M)$, e **confrontando** infine i due valori; se i **due valori** h e $H(M)$ risultano **identici** la **firma digitale** risulta **verificata**.

Le funzioni hash

La **funzione hash** usata per generare il blocco di dati di lunghezza fissa, da criptare con la chiave privata del mittente ed ottenere la firma digitale, **dove essere tale per cui $H(M)$ sia diverso da $H(M')$** , cioè la funzione *hash* calcolata su due diversi messaggi M e M' deve produrre due valori diversi. Purtroppo **questo requisito non è sempre vero** proprio perché la funzione *hash* **determina un valore di dimensione piccola e di lunghezza fissa, derivato da messaggi di dimensione variabile**, quindi si potrebbero avere delle **collisioni**, cioè la funzione *hash* potrebbe generare il medesimo valore a fronte di due messaggi diversi.

La **possibilità di trovare delle collisioni** usando la **funzione hash dipende dalla lunghezza del valore $H(M)$** , quindi per rendere difficoltoso l'uso di questa vulnerabilità dell'algoritmo *hash* da usare in un eventuale attacco, l'*hash* di un messaggio $H(M)$ **dove essere sufficientemente lungo da richiedere troppo tempo per trovare una collisione** utilizzando un messaggio M' diverso da M (es. SHA-256, SHA-512, anche indicate come SHA- 2^{65} , vengono considerate sicure).

In generale una **funzione hash**, definita come $y = H(x)$, è una funzione che possiede più controimmagini (pre-immagine), cioè **è una funzione non iniettiva**, e ciò comporta che:

- se H prende in input messaggi M lunghi b bit, l'insieme di tutti i possibili messaggi sarà costituito da 2^b elementi;
- se H produce un codice *hash* lungo sempre n bit, con $n < b$, l'insieme di tutti i possibili codici *hash* sarà costituito da 2^n elementi;
- quindi se i valori della funzione *hash* H risultano uniformemente distribuiti, ogni codice *hash* avrà in media 2^{b-n} controimmagini.

Ad esempio se supponiamo di avere dei messaggi lunghi 1000 bit e una funzione *hash* che genera codici lunghi 20 bit, l'insieme M di tutti i possibili messaggi è costituita da 2^{1000} elementi, mentre l'insieme H di tutti i possibili codici *hash* è costituito da 2^{20} elementi. Di conseguenza si avranno in media $2^{1000-20} = 2^{980}$ collisioni. Risulta quindi chiaro perché è importante la lunghezza del codice prodotto dalla funzione *hash*.

⁶⁵ Per approfondimenti su SHA-2 si veda [Secure Hash Algorithm - Wikipedia](#)

Una funzione *hash* H per essere considerata sicura, deve rispettare i seguenti requisiti⁶⁶, e quelli relativi alla sicurezza possono essere perseguiti totalmente o parzialmente a seconda dello scopo per cui viene usata la funzione *hash*.

Proprietà pratiche di implementazione:

- **input di dimensione variabile:** H deve essere applicabile a messaggi di qualsiasi dimensione;
- **output di dimensione fissa:** H deve produrre in output un codice di lunghezza fissa (e relativamente piccola);
- **efficiente:** H deve essere relativamente semplice da implementare e da applicare, cioè dato un messaggio x , deve essere semplice calcolare $y = H(x)$;
- **output pseudo-casuale:** l'output di H deve generare valori molto diversi anche a fronte di piccole variazioni nei messaggi.

Proprietà di sicurezza:

- **resistenza alla pre-immagine:** per ogni valore di *hash* y dato, deve essere computazionalmente difficile risalire ad un messaggio x tale per cui $H(x) = y$ (*funzione unidirezionale*), diversamente la funzione *hash* sarebbe vulnerabile agli attacchi alla pre-immagine;
- **resistenza alla seconda pre-immagine:** per ogni messaggio x deve essere computazionalmente difficile trovare un messaggio z con $x \neq z$, tale per cui $H(x) = H(z)$ (*proprietà debole della resistenza alle collisioni*), diversamente la funzione *hash* sarebbe vulnerabile agli attacchi alla seconda pre-immagine;
- **resistenza alla collisione:** deve essere computazionalmente difficile trovare una qualsiasi coppia di messaggi (x, z) con $x \neq z$, tale per cui $H(x) = H(z)$ (*proprietà forte della resistenza alle collisioni*). Questa proprietà implica la proprietà debole alla resistenza alle collisioni, ma non implica la resistenza alla pre-immagine.

Per un attaccante sarebbe più semplice provare a violare la *proprietà forte della resistenza alla collisione*⁶⁷ piuttosto che la *proprietà debole della resistenza alla collisione*⁶⁸, perché la probabilità di trovare una qualsiasi coppia di messaggi (x, z) i cui *hash* collidano è più elevata della probabilità di trovare un messaggio z il cui *hash* collida con quello di un messaggio x dato⁶⁹.

⁶⁶ Si veda [Funzione crittografica di hash - Wikipedia](#)

⁶⁷ Lo sforzo richiesto per violare la *proprietà forte della resistenza alla collisione* è proporzionale a $2^{m/2}$, dove m è la lunghezza in bit del blocco *hash*.

⁶⁸ Lo sforzo richiesto per violare la *proprietà debole della resistenza alla collisione* è proporzionale a 2^m , dove m è la lunghezza in bit del blocco *hash*.

⁶⁹ Queste probabilità vengono determinate tramite il **Paradosso del Compleanno** di cui potete trovare un'ottima spiegazione nel video della [Lezione 20](#) (anno 2014) del Prof. Steven Gordon, oltre che su innumerevoli siti su Internet.

Queste probabilità sono direttamente correlate alla lunghezza del blocco *hash*, quindi, anche se il blocco *hash* di lunghezza fissata non deve essere troppo lungo, tale lunghezza deve però essere tale da rendere computazionalmente difficile violare la *proprietà forte della resistenza alla collisione*, e quindi in un attacco a forza bruta deve abbassare la probabilità di trovare una qualsiasi coppia di messaggi (\mathbf{x}, \mathbf{z}) i cui *hash* collidano.

Si può fornire una motivazione per ognuna delle tre proprietà di sicurezza delle funzioni hash considerando il meccanismo di generazione della firma digitale, in cui la firma è apposta al valore hash $\mathbf{H}(\mathbf{m})$.

- \mathbf{H} dovrebbe essere una funzione hash che rispetti la *proprietà debole della resistenza alla collisione* altrimenti, un nemico \mathbf{C} potrebbe ricavare dalla firma l'impronta $\mathbf{H}(\mathbf{m})$, cercare un messaggio \mathbf{m}' tale che $\mathbf{H}(\mathbf{m}) = \mathbf{H}(\mathbf{m}')$, e quindi dimostrare che il mittente ha firmato \mathbf{m}' invece di \mathbf{m} .
- \mathbf{H} dovrebbe essere una funzione hash che rispetti la *proprietà forte della resistenza alla collisione* altrimenti, un nemico \mathbf{C} che possa scegliere il messaggio che il mittente deve firmare, ha solo bisogno di cercare una coppia $(\mathbf{m}, \mathbf{m}')$ che generi una collisione per modificare quanto è stato firmato, modificando ad esempio i termini di un contratto.
- \mathbf{H} dovrebbe essere una *funzione unidirezionale* e questo è dimostrabile proprio in uno schema di firma digitale a chiave pubblica. Considerando RSA, dove l'utente \mathbf{A} possiede una chiave pubblica (\mathbf{e}, \mathbf{n}) , un attaccante \mathbf{C} potrebbe scegliere un valore casuale \mathbf{f} , calcolare $\mathbf{h} = \mathbf{f}^{\mathbf{e}} \bmod \mathbf{n}$, e se \mathbf{C} può cercare una preimmagine \mathbf{m} tale che $\mathbf{H}(\mathbf{m}) = \mathbf{h}$, può dimostrare che \mathbf{f} è una firma del mittente sul messaggio \mathbf{m} .

Lo scopo di una funzione di hash è di creare un'impronta (*fingerprint*) del messaggio, ma non è una firma e quindi non garantisce autenticazione. Nel calcolo di un hash non si inserisce nessuna informazione segreta, per cui chiunque può generare l'hash corretto di qualunque messaggio. Inoltre una funzione di hash non è un algoritmo di cifratura, infatti calcolare l'hash di un messaggio non equivale a cifrarlo: il calcolo di un hash non include nessuna chiave, e soprattutto è un'operazione non invertibile.

Gli algoritmi di hash sono largamente utilizzati nei seguenti ambiti:

1. **Controllo degli errori:** in questo caso la funzione di hash viene utilizzata come un checksum, per identificare eventuali errori di trasmissione dei dati. Le proprietà aggiuntive degli hash non vengono in questo caso sfruttate.
2. **Integrità dei dati:** la funzione di hash viene utilizzata per garantire che un messaggio non sia stato modificato da un eventuale attaccante. Utilizzato ad esempio negli IDS (*Intrusion Detection System*) per controllare l'integrità dei file critici di un sistema operativo (*tripwire/fastsum*).
3. **Memorizzazione di password e autenticazione:** nella maggior parte dei casi, quando un sistema informatico deve memorizzare una password, questa non viene salvata in chiaro per motivi di sicurezza, né viene cifrata per evitare

che sia possibile risalire alla password originale. Si memorizza in questi casi l'hash della password. Ad esempio, sotto Windows, Linux e MacOS le password degli utenti sono salvate sotto forma di hash. Quando un utente scrive la propria password per accedere al sistema, ne viene calcolato l'hash e confrontato con quello memorizzato nel sistema.

4. **Firme digitali:** si è già detto che la crittografia a chiave pubblica è computazionalmente molto onerosa, per cui solitamente viene usata solo per trasmettere la chiave per usare la crittografia simmetrica Per le firme digitali esiste un problema di prestazioni analogo, infatti firmare l'intero documento è computazionalmente pesante. La soluzione trovata è stata quella di firmare solo un hash del documento (ha anche il vantaggio di lasciare il documento in chiaro, leggibile anche da chi non dispone degli strumenti per verificare la firma digitale).
5. **Nell'ambito dell'informatica forense per validare digitalmente i dati acquisiti, tipicamente le copie forensi.** La recente legislazione impone infatti una **catena di custodia** che permetta di preservare i reperti informatici da eventuali modifiche successive all'acquisizione. Tramite i codici hash è possibile in ogni momento verificare che quanto repertato sia rimasto immutato nel tempo. Se i codici hash corrispondono, entrambe le parti in un procedimento giudiziario hanno la certezza di poter lavorare sulla stessa versione dei reperti, garantendo quindi una uniformità di analisi.

CLIL - Homework

Study all the topics your teacher spoke about and watch the following videos.

1. [What is a zero-knowledge proof?](#)
2. [What is a digital certificate?](#)
3. [What is access control?](#)

I certificati digitali e Autorità di Certificazione

Nella tecnologia di crittografia a chiave pubblica il **recupero della chiave pubblica pone un problema di sicurezza**: essendo distribuita pubblicamente è particolarmente critico garantirne l'autenticità, ossia **assicurare che una certa chiave pubblica appartenga effettivamente all'interlocutore** con cui si instaura la comunicazione. Se infatti, una terza parte sostituisse la chiave pubblica del destinatario con la propria, il contenuto dei messaggi cifrati sarebbe disvelato e le firme digitali potrebbero essere falsificate.

La distribuzione delle chiavi pubbliche è pertanto il problema cruciale della tecnologia a chiave pubblica. **Il problema della distribuzione delle chiavi pubbliche viene risolto tramite l'impiego dei certificati elettronici. Un certificato è un documento elettronico che associa una chiave pubblica ad una persona fisica/ente.**

L'utilizzo dei **certificati elettronici** presuppone l'esistenza di una **Autorità di Certificazione** (*Certification Authority* o CA) che li emetta e li gestisca, oltre che firmare il certificato stesso validandolo.

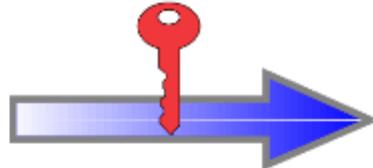
I **compiti di una CA** sono:

- rilascio e pubblicazione del certificato firmato con la propria chiave privata;
- manutenzione del registro delle chiavi pubbliche;
- revoca o sospensione dei certificati in caso di istanza dell'interessato o in caso di abusi, falsificazioni, ecc.;
- aggiornamento della lista pubblica dei certificati sospesi o revocati (*Certificate Revocation List*).

Documento con identità e chiave pubblica di Mario Rossi



Chiave Privata del soggetto certificatore



Certificato di Mario Rossi

Nome: Mario
Cognome: Rossi
Indirizzo: via

Chiave Pubblica di Mario Rossi

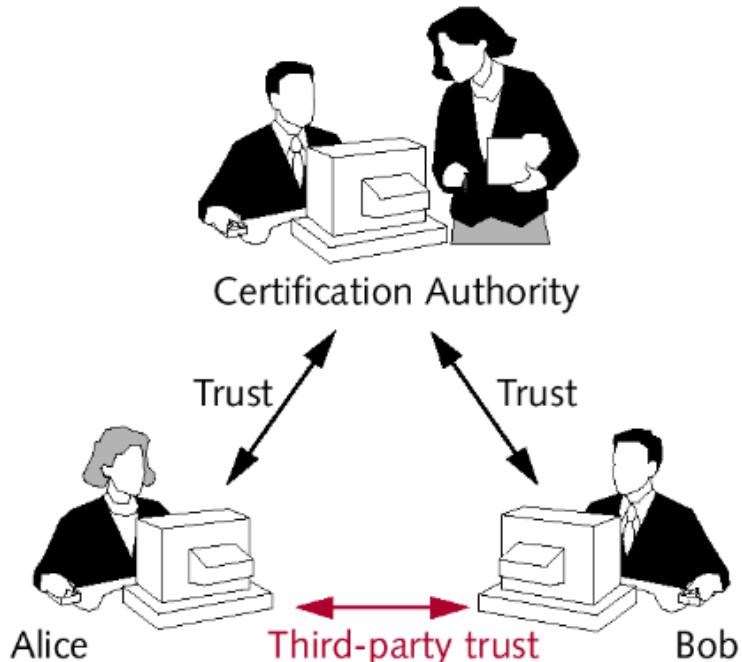
Firma del soggetto certificatore

Documento firmato dal soggetto certificatore

Infrastruttura a chiave pubblica (PKI)

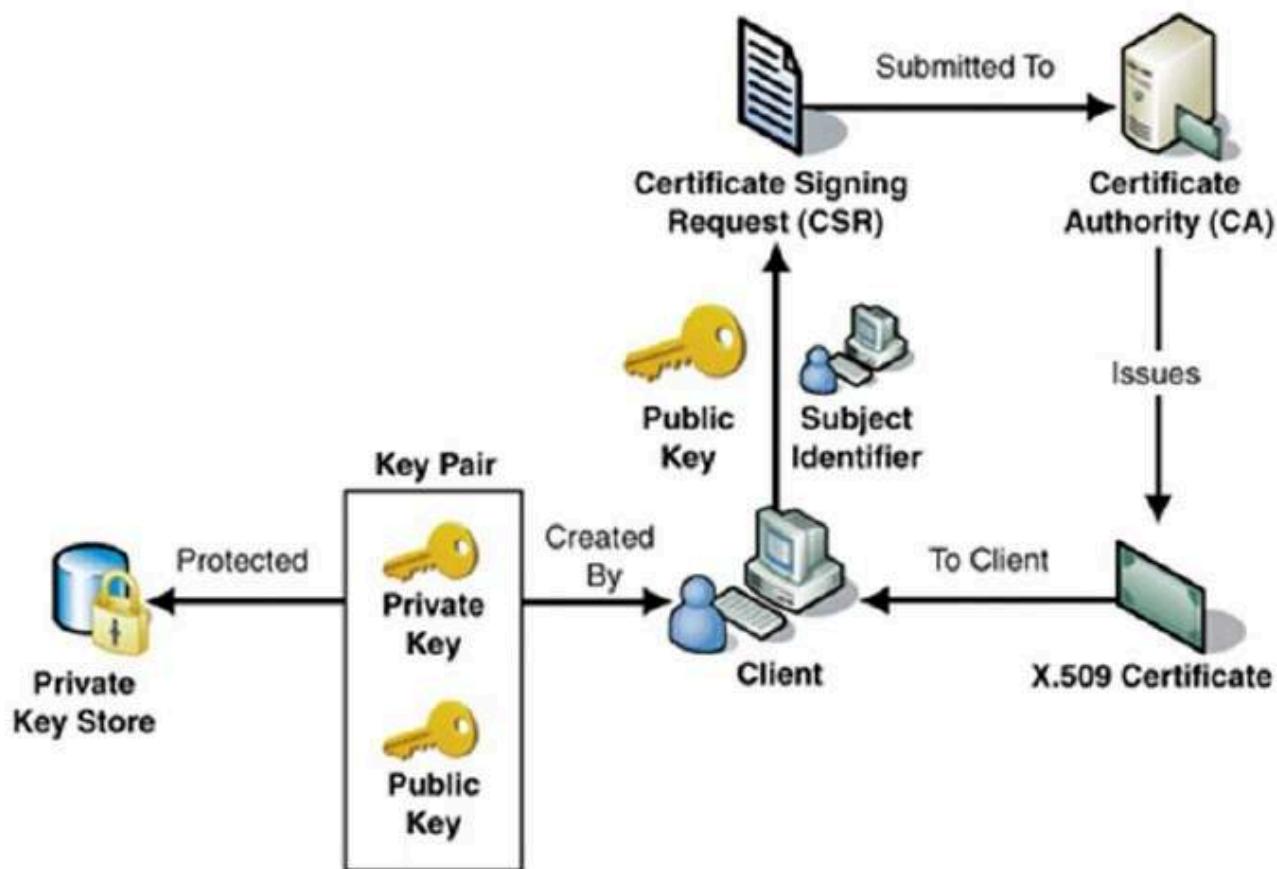
Le **Infrastrutture a chiave pubblica** (*Public Key Infrastructure*) forniscono il supporto necessario affinché la tecnologia di crittografia a chiave pubblica sia utilizzabile su larga scala.

Una infrastruttura a chiave pubblica introduce il concetto di **third-party trust**, ossia di quella situazione che si verifica quando **due generiche entità si fidano implicitamente l'una dell'altra**, senza che abbiano precedentemente stabilito una personale relazione di fiducia, perché **entrambe le entità condividono una relazione di fiducia con una terza parte comune**.



Third-party trust è un requisito fondamentale per qualsiasi **implementazione** su larga scala che utilizzi la **crittografia a chiave pubblica**, e in una *PKI* viene realizzata attraverso l'**Autorità di Certificazione**.

In Italia le Autorità di Certificazione sono, ad esempio, Infocamere, Poste italiane, Actalis S.p.A., ecc.. L'elenco aggiornato e completo dei certificatori qualificati è possibile trovarlo sul sito dell'[AgID](#) (Agenzia per l'Italia Digitale).



Il modello di una *PKI* in realtà prevede due enti: l'**autorità di registrazione** e quella di certificazione.

RA (*Registration Authority*): Autorità di Registrazione

L'accertamento dell'identità dell'utente che richiede un certificato elettronico deve precedere l'effettiva emissione del certificato. Questa verifica è indispensabile perché con l'emissione di un certificato elettronico si rende pubblicamente valida l'associazione tra una certa chiave pubblica e una certa entità.

Una volta attestata la validità dell'identità dell'utente attraverso una serie di procedure definite nell'ambito di una precisa politica di sicurezza (ad esempio, il controllo della carta di identità), l'**Autorità di Registrazione (RA)** ha il compito di abilitare l'utente come appartenente ad uno specifico **dominio di fiducia**.

La funzionalità di **Autorità di Registrazione** può essere espletata dall'Autorità di Certificazione stessa oppure delegata ad altre entità.

CA (*Certification Authority*): Autorità di Certificazione

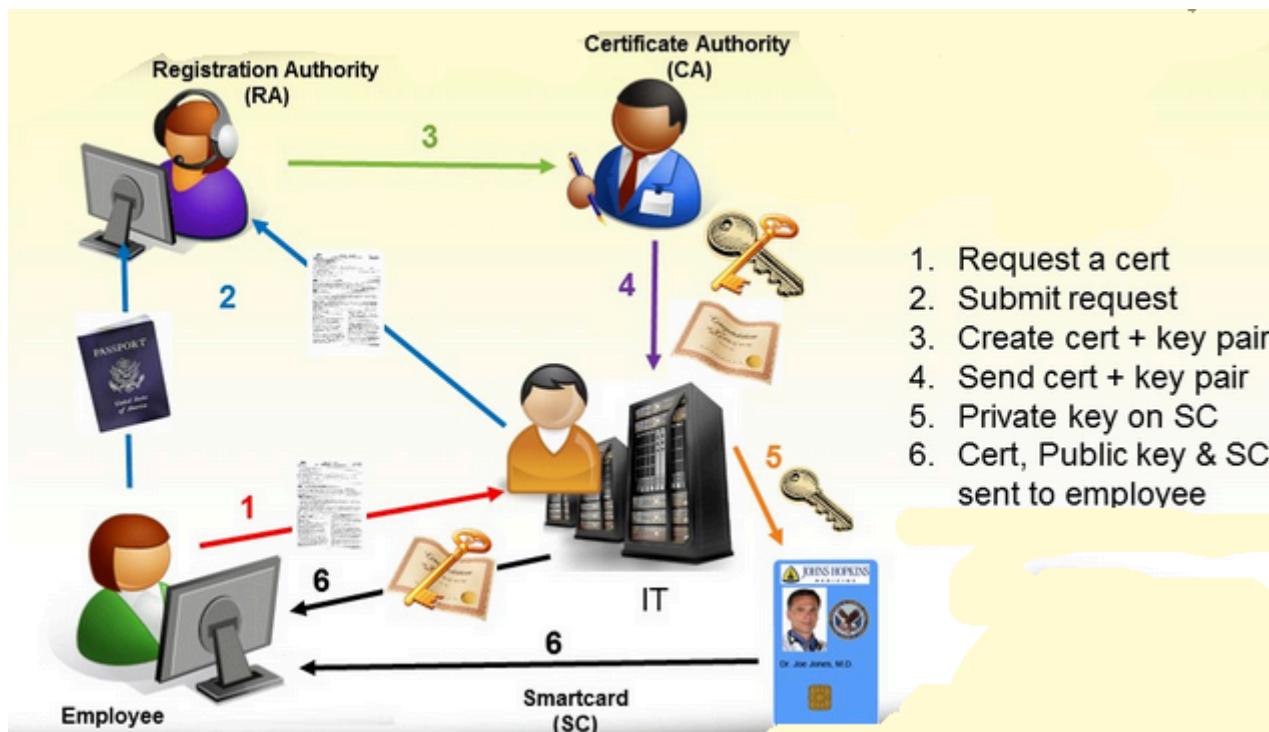
L'**Autorità di Certificazione** costituisce il cuore di una *PKI*, e la sua principale funzione consiste nel **creare i certificati elettronici per quegli utenti**.

precedentemente abilitati nella fase di registrazione al dominio di fiducia, di cui la CA è garante.

Un'**Autorità di Certificazione** non si deve limitare esclusivamente alla generazione dei certificati, ma deve poterne gestire l'intero ciclo di vita.

Il **ciclo di vita di un certificato** comprende le fasi di **generazione, aggiornamento** (nel caso in cui il certificato stia per perdere validità temporale), **sostituzione** (nel caso di scadenza della validità temporale) e **revoca** (nel caso in cui le condizioni di emissione del certificato non siano più valide).

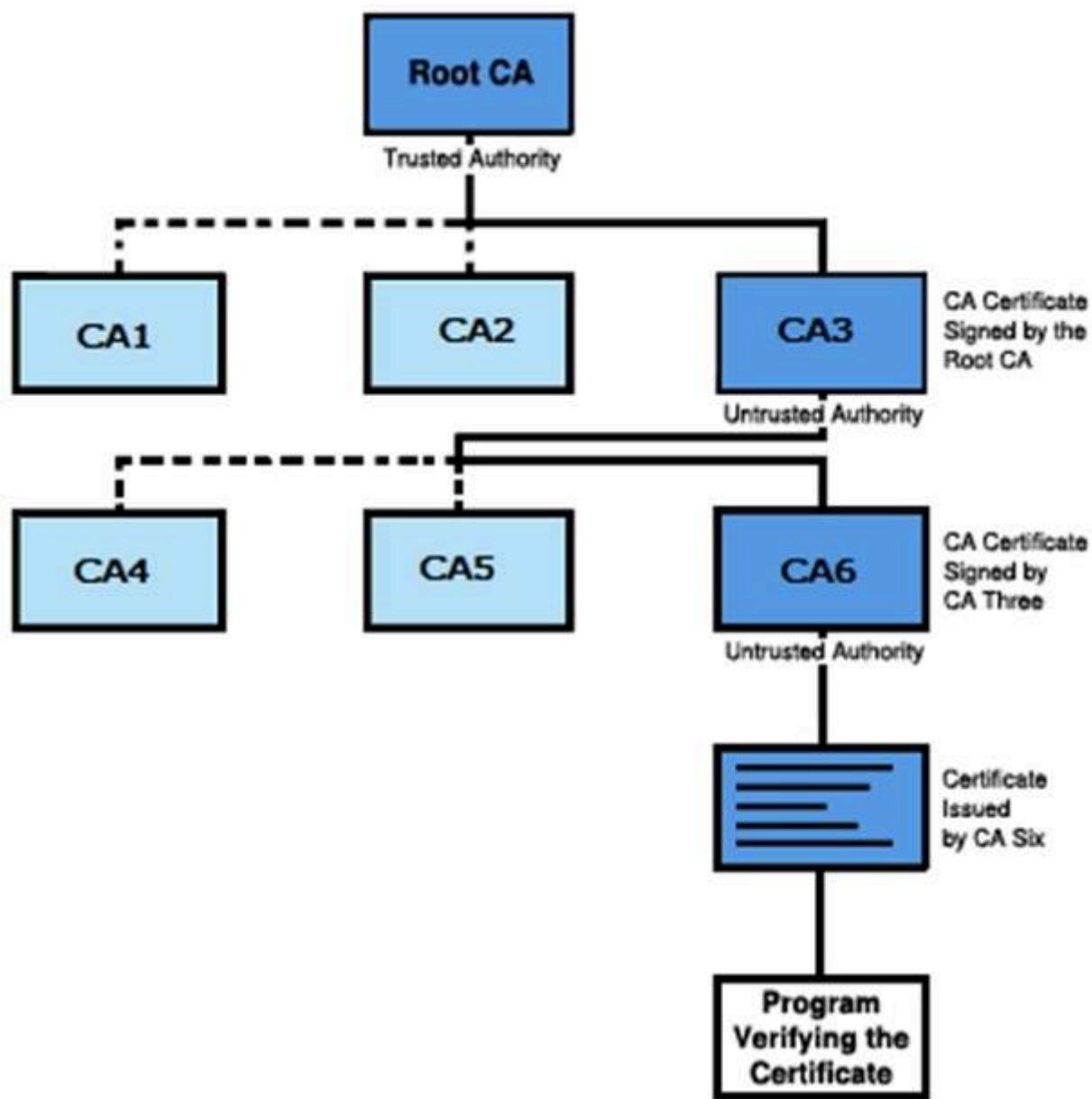
Un ulteriore compito della **CA** è quello di **stabilire relazioni di fiducia con altre CA**.



La **CA pubblica su** un specifico **server pubblico** detto **Certificate Server** liberamente accessibile, **la lista dei certificati in corso di validità** o con l'indicazione se questi certificati sono revocati o sospesi.

Una **CA può emettere un certificato anche per un'altra CA**, in modo da creare **catene gerarchiche di certificati**. Controllare la **validità di un certificato** significa quindi **risalire la sua catena di autorizzazioni**, fino a **raggiungere** quella di una **CA fidata**. In cima alla catena gerarchica c'è una **RootCA**, che possiede un **certificato auto-firmato (Root Certificate)**.

I browser in circolazione sono configurati per dare fiducia ad un certo numero, modificabile, di CA ben note, e memorizzano i relativi *Root Certificate*. Anche i certificati di CA fidate dall'utente sono memorizzate nel browser.



Lo standard X.509 per i certificati

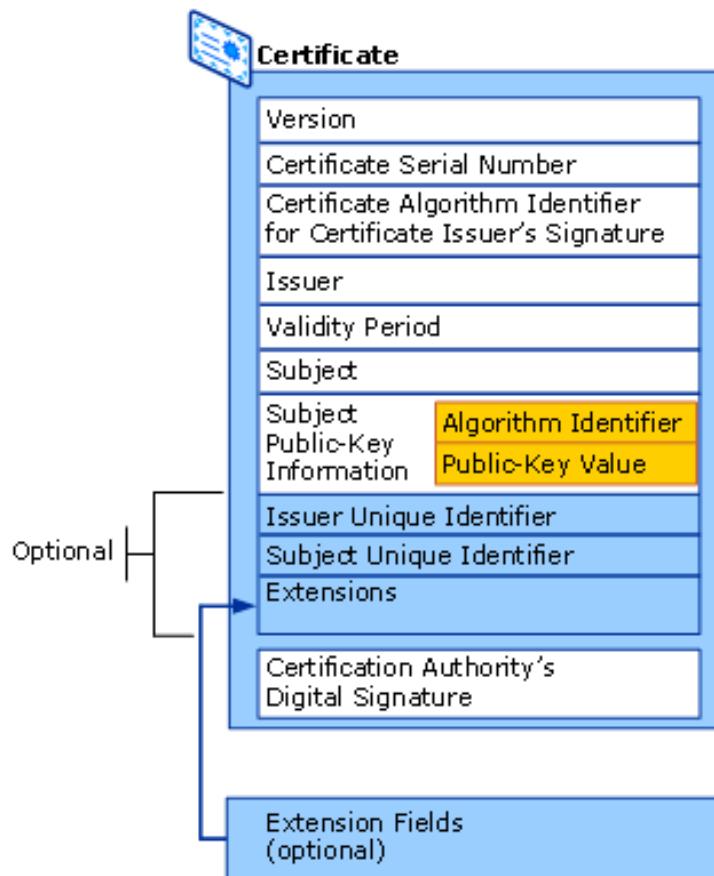
Lo **standard di definizione del formato dei certificati** e ormai diffusamente riconosciuto è quello descritto nello standard **X.509 ISO/IEC/ITU** [[RFC 2459](#)].

Visa e MasterCard hanno adottato le specifiche X.509 come base per la definizione dello standard per il commercio elettronico SET (*Secure Electronic Transaction*).

Lo **standard X.509** prevede che ogni certificato sia costituito da una struttura dati contenente almeno i seguenti campi:

- **versione**: indica la versione del formato del certificato (1, 2 o 3);
- **serial number**: è un codice numerico che identifica il certificato tra tutti i certificati emessi dall'**Autorità di Certificazione**;
- **signature algorithm**: specifica l'algoritmo utilizzato dalla **CA** per firmare il certificato, ed è data dalla coppia *funzione hash – algoritmo a chiave pubblica*;
- **issuer name**: è il nome della **CA**;

- **subject name:** informazioni che identificano univocamente il possessore di una chiave pubblica;
- **il valore della chiave pubblica;**
- **il periodo di validità temporale** del certificato (*da ... a*);
- la **firma digitale dell'Autorità di Certificazione** con cui si assicura l'**autenticità della chiave e l'integrità delle informazioni** contenute nel certificato;
- **Subject Unique Identifier:** è una stringa di bit aggiuntivi, opzionale, usata nel caso di omonimia di due membri in una stessa **CA**;
- **Issuer Unique Identifier:** è una stringa di bit aggiuntivi, opzionale, usata nel caso in cui nella struttura ad albero ci siano due **CA** con lo stesso nome.



Se un intruso tentasse di alterare il contenuto di un certificato durante la sua pubblicazione, la manomissione sarebbe immediatamente rilevata in fase di verifica della firma sul certificato; il processo di verifica fallirebbe e l'utente finale sarebbe avvertito della non integrità della chiave pubblica contenuta nel certificato.

Certificate Example

```
Data:  
Version: 1 (0x0)  
Serial Number: 123 (0x7B)  
Signature Algorithm: md5WithRSAEncryption  
Issuer: C=US, ST=New York, L=Armonk, O=IBM, OU=HQ, CN=IBM Headquarters  
Validity  
    Not Before: Jan 15 15:34:56 2001 GMT  
    Not After : Feb 14 15:34:56 2001 GMT  
Subject: C=US, ST=New York, L=Poughkeepsie, O=IBM, OU=TPF, CN=John Doe  
Subject Public Key Info:  
    Public Key Algorithm: rsaEncryption  
    RSA Public Key: (1024 bit)  
        Modulus (1024 bit):  
            00:a7:92:dd:6b:78:5a:45:69:69:c3:91:f8:05:a6:  
            55:15:8c:fc:d9:75:68:61:26:80:ce:0c:dc:60:dd:  
            55:f9:ef:08:42:10:10:1c:0b:14:6a:4a:0a:08:64:  
            42:51:7a:c7:78:ee:2e:0c:1e:17:75:b4:a3:46:c8:  
            79:fb:a2:23:11:3a:4f:f0:0e:ac:b1:d4:23:32:aa:  
            25:57:cf:08:9a:50:d4:88:73:b6:97:24:6a:9f:f9:  
            43:d8:fd:db:2a:f7:74:42:d8:e6:36:f2:b4:fe:fa:  
            e3:0b:4f:35:a0:53:ee:26:31:dc:87:7d:8e:c7:2e:  
            80:0d:31:a1:cb:26:73:a3:67  
        Exponent: 65537 (0x10001)  
  
Signature Algorithm: md5WithRSAEncryption  
5c:a8:33:a7:ca:19:19:0c:4c:3d:88:88:22:da:2e:03:14:c2:  
6e:5f:38:e5:d6:00:36:e2:0f:9d:60:3f:68:e2:3b:06:d3:51:  
9b:2f:b0:a5:8a:48:5e:e0:0a:ee:e2:a6:74:2a:87:c6:01:29:  
10:c2:8e:49:3b:0a:18:37:5a:dd:f2:9a:65:42:1c:0a:c9:dc:  
a8:70:27:82:b8:33:f3:5f:c2:5e:c4:7c:c5:f2:b8:ad:6a:93:  
b5:e7:68:b8:7d:92:2c:15:4a:ec:0a:c8:05:6c:36:32:93:3c:  
73:0f:10:3f:70:52:20:14:89:4a:16:5d:26:18:59:14:a7:1b:  
e2:cb
```

Data Section

Signature Section

21 - CLIL - Homework

Study all the topics your teacher spoke about and watch the following videos.

1. [What is a certificate authority?](#)

22 - Laboratorio su GnuPG

GnuPG è un programma che permette di comunicare in modo sicuro utilizzando la crittografia a chiave pubblica.

In questo laboratorio genererete delle chiavi, le scambierete con un vostro compagno, invierete un messaggio cifrato e decifrerete quello che avrete ricevuto, firmerete un messaggio e verificherete la firma del messaggio ricevuto.

Il lavoro dovrà essere svolto a coppie e ogni singolo studente dovrà consegnare una relazione distinta al termine del lavoro⁷⁰.

Materiali

Per svolgere il laboratorio utilizzerete i seguenti materiali pubblicati sul sito di e-learning dell'[I.I.S. Giuseppe Peano](#) di Torino⁷¹.

- [Cos'è gnuPG](#)
- [Generare una coppia di chiavi](#)
- [Scambio di chiavi](#)
- [Cifrare e decifrare messaggi](#)
- [Firmare e verificare la firma](#)
- <https://www.gnupg.org>

Relazione

1. Ogni studente ST1/ST2 crei la propria coppia di chiavi.

Comando:

Input del comando:

Output del comando:

Chiave pubblica:

Chiave privata:

2. ST1 esporti la chiave pubblica e la invii allo studente ST2 con cui lavora (e viceversa).

Comando:

Descrizione: *spiegare quali argomenti richiede il comando*

Output del comando:

Chiave pubblica ST1 formato ASCII:

3. ST1 importi la chiave del ST2 e ne verifichi l'impronta (e viceversa).

Comando per importare:

Descrizione: *spiegare quali argomenti richiede il comando*

Chiave pubblica ST2/ST1:

Comando per firmare:

Descrizione: *spiegare quali argomenti richiede il comando*

Descrizione: *spiegare perché è necessario firmare la chiave*

Comando per visualizzare le chiavi:

Output del comando per visualizzare le chiavi: *spiegare cosa viene visualizzato*

4. Inviare un messaggio cifrato.

Comando:

Descrizione: *spiegare quali argomenti richiede il comando*

Input del comando:

Output del comando:

⁷⁰ Si ringrazia la Prof.ssa Sophia Danesino per il materiale fornito per questo laboratorio.

⁷¹ Potreste riscontrare qualche differenza per la diversa versione del programma installato. Il segreto è adattarsi.

5. Decifrare un messaggio ricevuto dal studente ST2.

Comando:

Descrizione: *spiegare quali argomenti richiede il comando*

Input del comando:

Output del comando:

6. Firmare un messaggio ed inviare messaggio in chiaro e firma.

Comando:

Descrizione: *spiegare quali argomenti richiede il comando*

Input del comando:

Output del comando:

7. Ricevuto un messaggio verificare la firma.

Comando:

Descrizione: *spiegare quali argomenti richiede il comando*

Input del comando:

Output del comando:

8. Cifrare e firmare un messaggio con un unico comando.

Comando:

Descrizione: *spiegare quali argomenti richiede il comando*

Input del comando:

Output del comando:

9. Decifrare e verificare la firma con un unico comando.

Comando:

Descrizione: *spiegare quali argomenti richiede il comando*

Input del comando:

Output del comando:

Comandi gpg

Comando	Descrizione	
\$ gpg --gen-key	crea una coppia di chiavi	
\$ gpg --list-keys	elenca le chiavi conosciute	
\$ gpg --output Bob.gpg --export bob@segreto.net	esporta una chiave pubblica	
\$ gpg --armor --export bob@segreto.net	visualizza una chiave pubblica in formato ASCII	
\$ gpg --import Bob.gpg	importa una chiave pubblica	
\$ gpg --fingerprint	visualizza l'impronta	
\$ gpg --edit-key bob@segreto.net	modifica una chiave	
	> fpr	visualizza l'impronta
	> sign	firma
	> check	verifica la chiave
	> quit	esce
\$ gpg --output MsgCrt --encrypt --recipient alice@segreto.net MsgChiaro	cifra un documento	
\$ gpg --output MsgChiaro --decrypt MsgCrt	decifra un documento	
\$ gpg --output MsgFirma --sign MsgChiaro	genera un documento firmato e cifrato	
\$ gpg --output MsgVerificato --decrypt MsgFirma	verifica un messaggio firmato	
\$ gpg --output Firma --detach-sig MsgChiaro	genera un file con la firma del documento a partire dal documento in chiaro	
\$ gpg --verify Firma MsgChiaro	verifica la firma a partire dal documento in chiaro e dal file con la firma	
\$ gpg --delete-key Alice	cancella la chiave pubblica di Alice	
\$ gpg --delete-secret-key Bob	cancella la chiave privata (segreta) di Bob	

E - Protocolli per la sicurezza della rete

Prefazione

I seguenti appunti sono in parte basati sul lavoro svolto dalla Prof.ssa Sophia Danesino, mentore, amica, appassionata di conoscenza e insegnante fuori dal comune. Sul suo [canale YouTube](#) è possibile trovare i video su molti degli argomenti trattati.

Per l'arricchimento e la stesura finale degli appunti sono risultate fondamentali le lezioni del [Prof. Steven Gordon](#) a cui va la mia profonda gratitudine per la pubblicazione delle sue lectures sul suo [canale YouTube](#).

I protocolli sicuri

Le tecniche crittografiche presentate nel capitolo precedente sono applicate ai protocolli di comunicazione al fine di garantire la sicurezza nelle applicazioni di rete, e ciò è indispensabile per attuare le contromisure necessarie a garantire la riservatezza, l'integrità e autenticità come, per esempio, nelle transazioni commerciali o finanziarie.

La sicurezza tramite crittografia è applicata ai protocolli del modello TCP/IP, a partire dal livello di rete fino a quello di applicazione, a seconda delle esigenze.

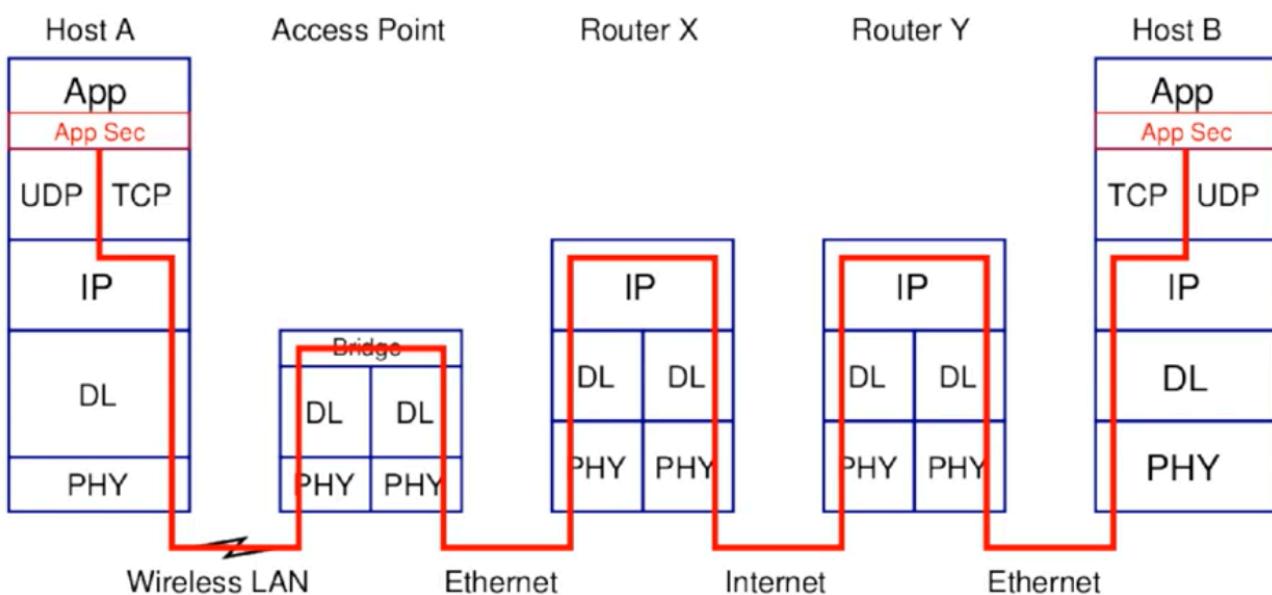
L'applicazione della sicurezza a un protocollo di un determinato livello della pila TCP/IP protegge anche i protocolli di livello superiore, e quindi se viene applicata la crittografia a livello di rete, risulteranno sicuri anche i protocolli di livello trasporto e applicazione. Se invece la sicurezza venisse applicata a livello di trasporto, sarà garantita la sicurezza al livello applicazione, ma non al livello di rete.

La tabella seguente evidenzia alcuni dei protocolli di sicurezza utilizzabili nei diversi livelli della pila protocollare TCP/IP.

Livello applicazione	PGP/GPG, Kerberos, Radius, SSH, S/MIME
Livello trasporto	Transport Layer Security (TLS)
Livello rete	IP Security (IPSec)

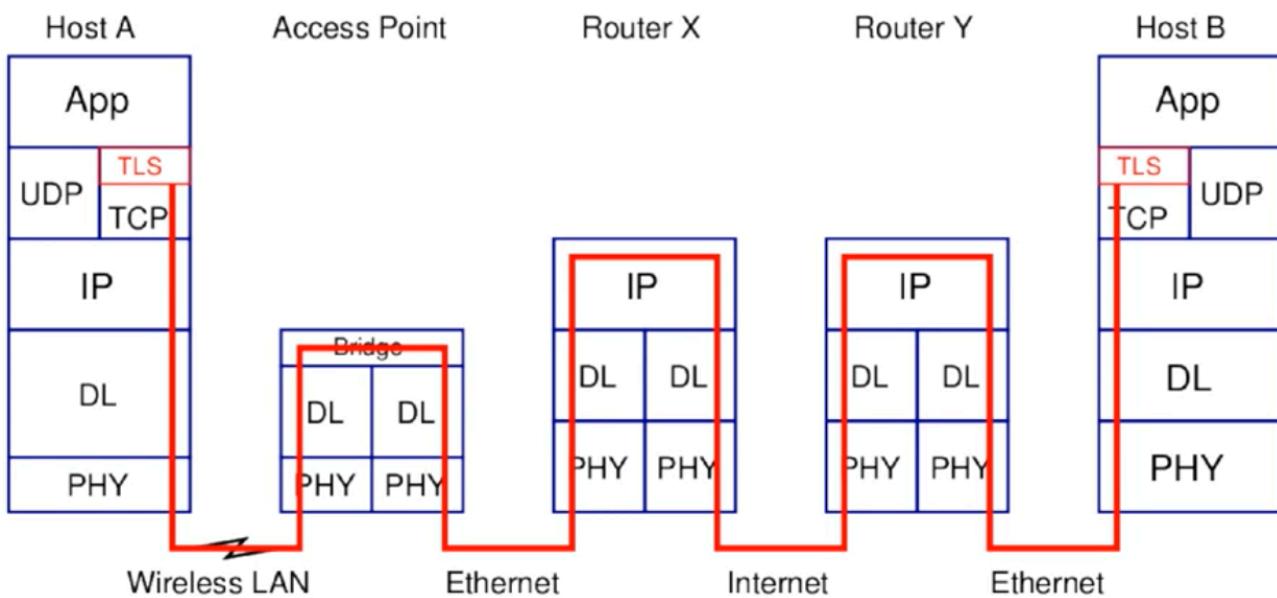
I protocolli per la gestione della sicurezza a livello applicazione, come **PGP/GPG, Kerberos, Radius, SSH, S/MIME**, ecc., sono protocolli che vengono associati ad altre applicazioni per renderle sicure. La sicurezza viene garantita solo al livello applicazione, i livelli sottostanti non sono gestiti in modo sicuro.

Application Level Security: Application-Specific



Il protocollo **TLS** garantisce la sicurezza al livello applicazione e si appoggia al protocollo TCP, non funzionando di solito con UDP. Un'applicazione che usa TLS non dovrà quindi occuparsi della sicurezza che verrà fornita da questo protocollo. Il protocollo TLS è comunemente usato con il protocollo HTTP (HTTPs), ma può essere usato anche con altre applicazioni come la posta elettronica o l'instant messaging.

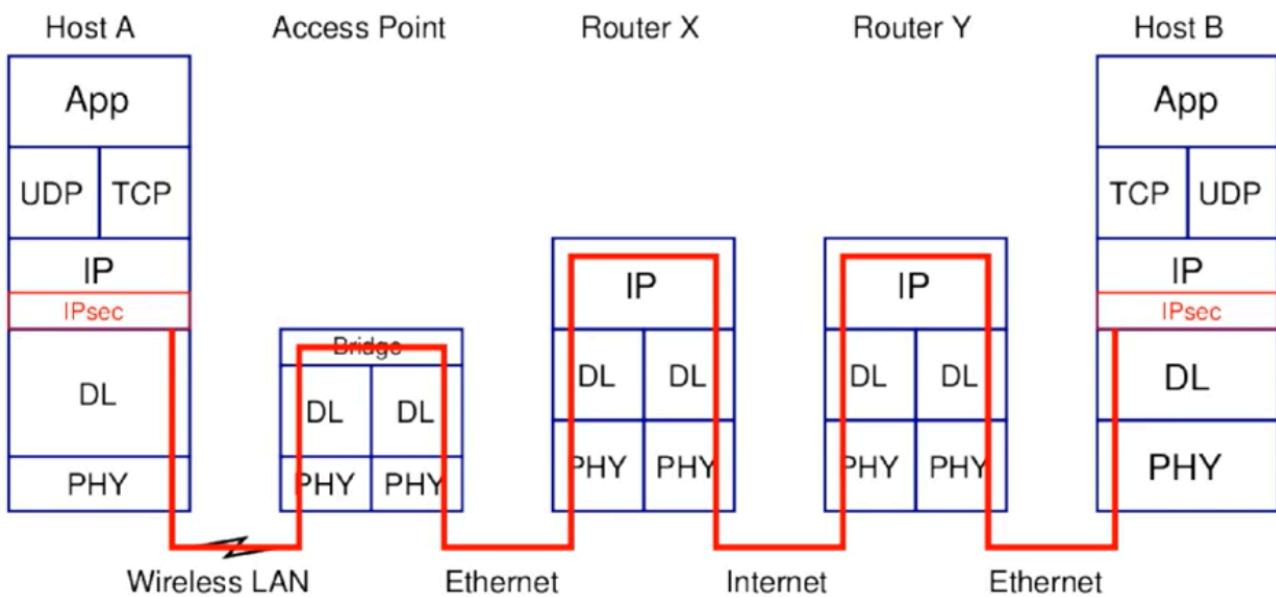
Transport Level Security: TLS/SSL



Il protocollo **IPSec** può offrire una protezione end-to-end, ma può anche essere limitato a porzioni della rete più limitati, e può proteggere gli indirizzi IP del mittente e del destinatario quando usato in modalità **tunnel mode** tra host e router, o tra router e router, oltre a proteggere comunque tutti gli header dei PDU dei livelli superiori a quello di rete e il contenuto del messaggio quando usato in **transport mode** tra end system. Il protocollo IPSec può essere adottato sia per applicazioni basate su TCP, sia per applicazioni basate su UDP.

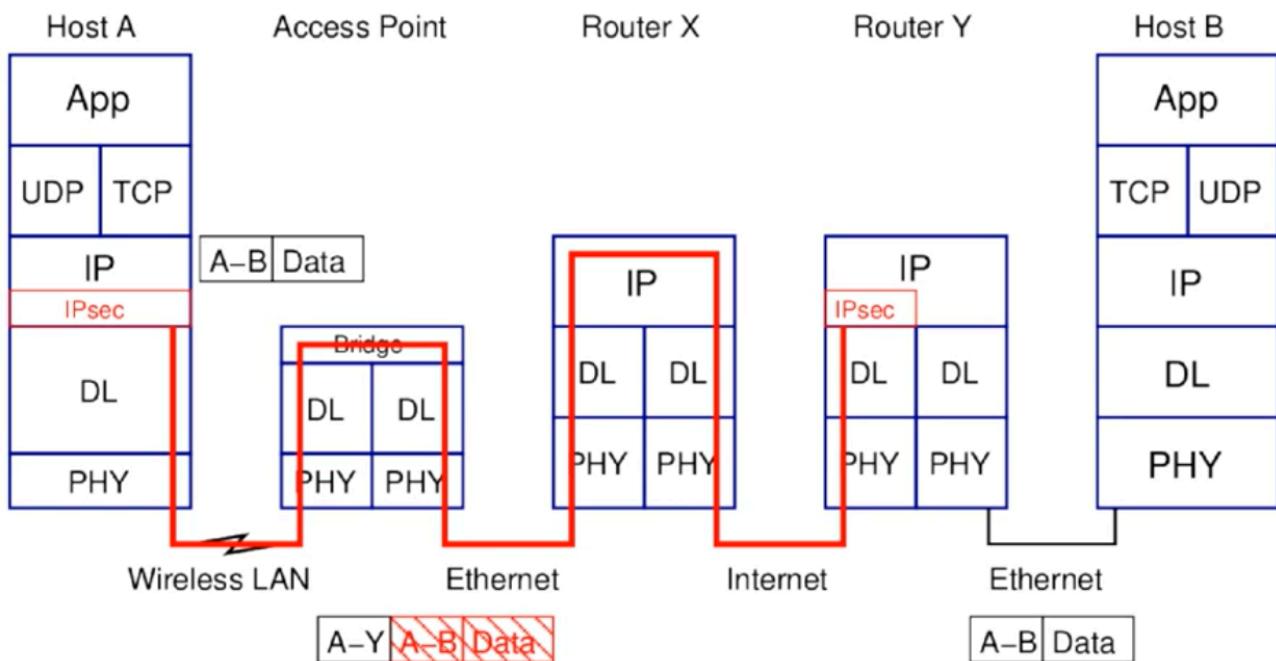
IPSec usato in **transport mode**.

Network Level Security: IPsec End-to-End

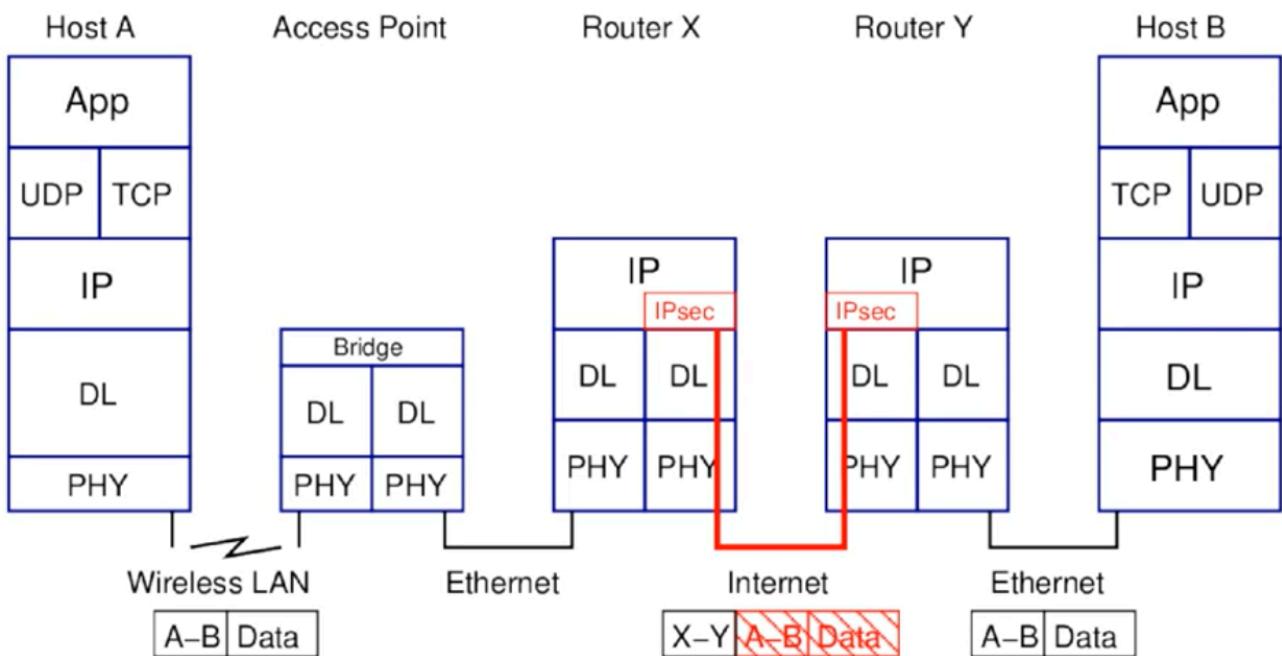


IPSec usato in **tunnel mode**.

Network Level Security: IPsec Host-to-Router



Network Level Security: IPsec Router-to-Router



Saranno analizzati alcuni dei principali protocolli di sicurezza, partendo da quelli più in basso nella pila protocollare TCP/IP.

IP Security

IPSec

IPsec (*Internet Protocol Security*) è un sistema che permette di proteggere il traffico IP a livello network. La ragione principale per usare IPsec risiede nel fatto che il protocollo IP non ha alcuna caratteristica di protezione o di autenticazione.

IPsec è in grado di proteggere il traffico raggiungendo i seguenti obiettivi:

- **Confidenzialità**: i dati vengono criptati in modo tale che nessuno possa leggerli all'infuori del mittente e del ricevente.
- **Integrità**: usando una funzione *hash* per ottenere il *digest* dei dati scambiati, il mittente e il ricevente possono essere ragionevolmente certi che una eventuale modifica venga rilevata.
- **Autenticazione**: il mittente e il ricevente si autenticheranno l'un l'altro in modo che siano sicuri di comunicare con l'entità corretta.
- **Anti-replay**: anche se un pacchetto è stato criptato e autenticato, un attaccante potrebbe tentare di catturare questi pacchetti per inviarli nuovamente. IPsec rileva i pacchetti duplicati usando un *sequence number*.

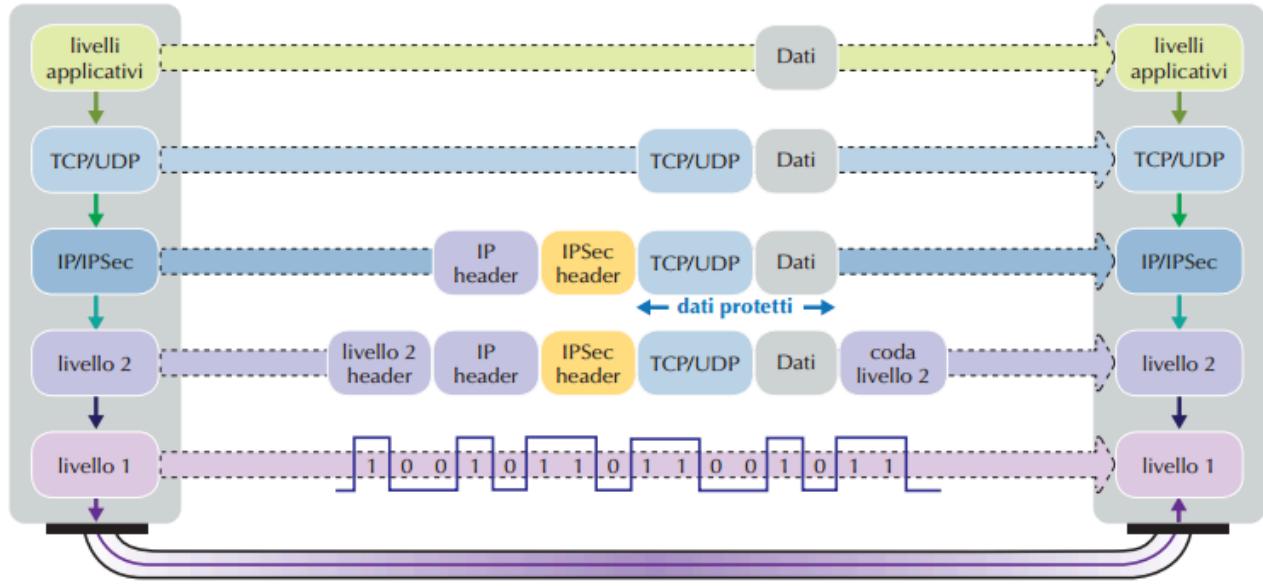
IPsec non è un singolo protocollo ma piuttosto un'architettura di sicurezza a livello Network, composta da vari protocolli e da altri elementi. I protocolli principali che costituiscono IPsec sono tre:

4. **Authentication Header (AH)**: garantisce l'autenticazione e l'integrità del messaggio ma non offre la confidenzialità;
5. **Encapsulating Security Payload (ESP)**: fornisce autenticazione, confidenzialità e controllo di integrità del messaggio;
6. **Internet Key Exchange (IKE)**: è un protocollo di livello applicativo che implementa lo scambio delle chiavi per realizzare il flusso crittografato.

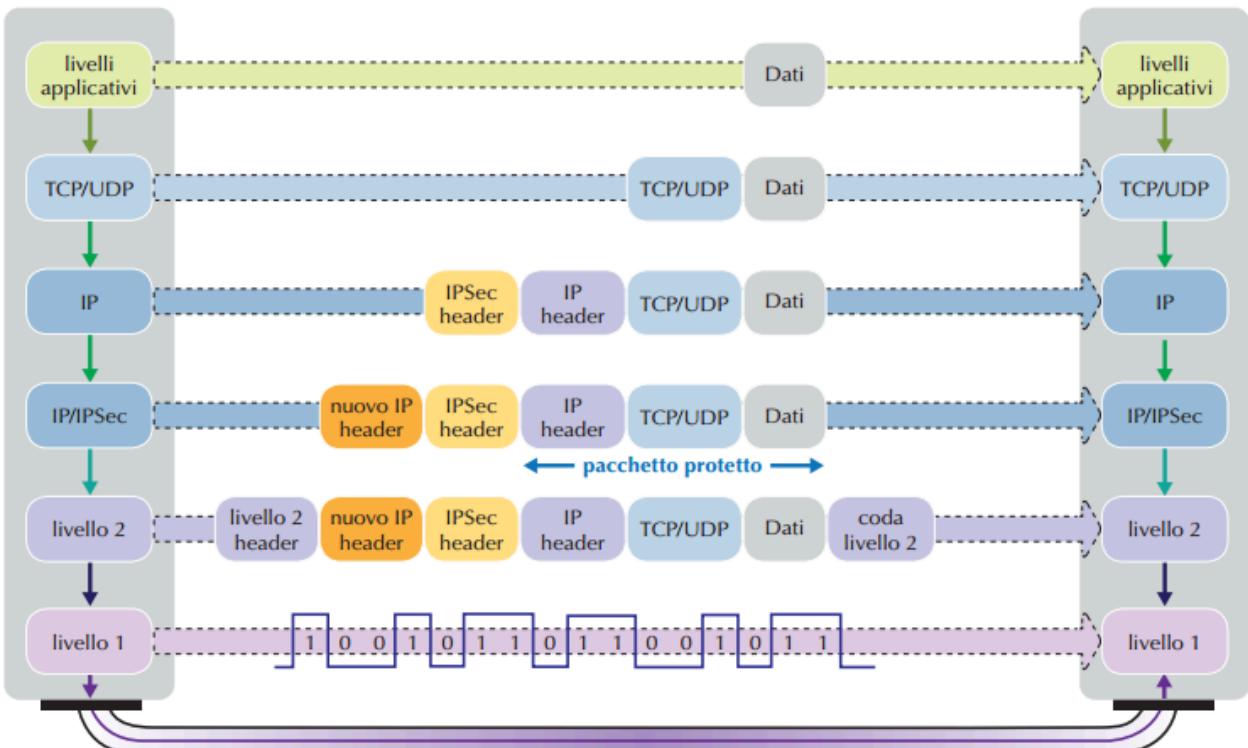
AH (*Authentication Header*) ed **ESP** (*Encapsulating Security Payload*) non si preoccupano dello scambio delle chiavi e presumono che i due interlocutori si siano già accordati usando il protocollo **IKE** (*Internet Key Exchange*) creando tra loro una **Security Association (SA)**, ovvero un "contratto" che specifica quali meccanismi di sicurezza utilizzare e con quali chiavi. È quindi affidato a **IKE** (*Internet Key Exchange*) il compito di negoziare e gestire le diverse **Security Association**, secondo politiche definite localmente. Una **SA** può poi essere associata ad **AH** (*Authentication Header*) o a **ESP** (*Encapsulating Security Payload*). Una **SA** (*Security Association*) ha un tempo di vita, che può essere specificato in termini temporali oppure come quantità di dati trasferiti.

Sia **AH** (*Authentication Header*), sia **ESP** (*Encapsulating Security Payload*) possono essere utilizzati in **modalità trasporto** o in **modalità tunnel** (*tunneling*).

In **modalità trasporto** (*end-to-end*) IPsec agisce tra due end system e aggiunge gli header dei protocolli utilizzati (**AH** o **ESP**) tra header IP e l'header del protocollo di trasporto (TCP o UDP).



In **modalità tunneling** il pacchetto IP originario viene interamente incapsulato in un nuovo pacchetto IP che riporta nel nuovo header IP gli indirizzi di uno o due intermediate system che implementano IPsec.



Security association e policy

Un concetto fondamentale nell'ambito di IPsec è quello di **SA** (*Security Association*) che rappresenta un'associazione tra due parti che comunicano tramite IPsec.

Una **SA** (*Security Association*) specifica i meccanismi di sicurezza, gli algoritmi e le chiavi usati per proteggere il traffico. Le **SA** (*Security Association*) sono unidirezionali, per cui sono necessarie due **SA** per permettere a due host di comunicare tra loro.

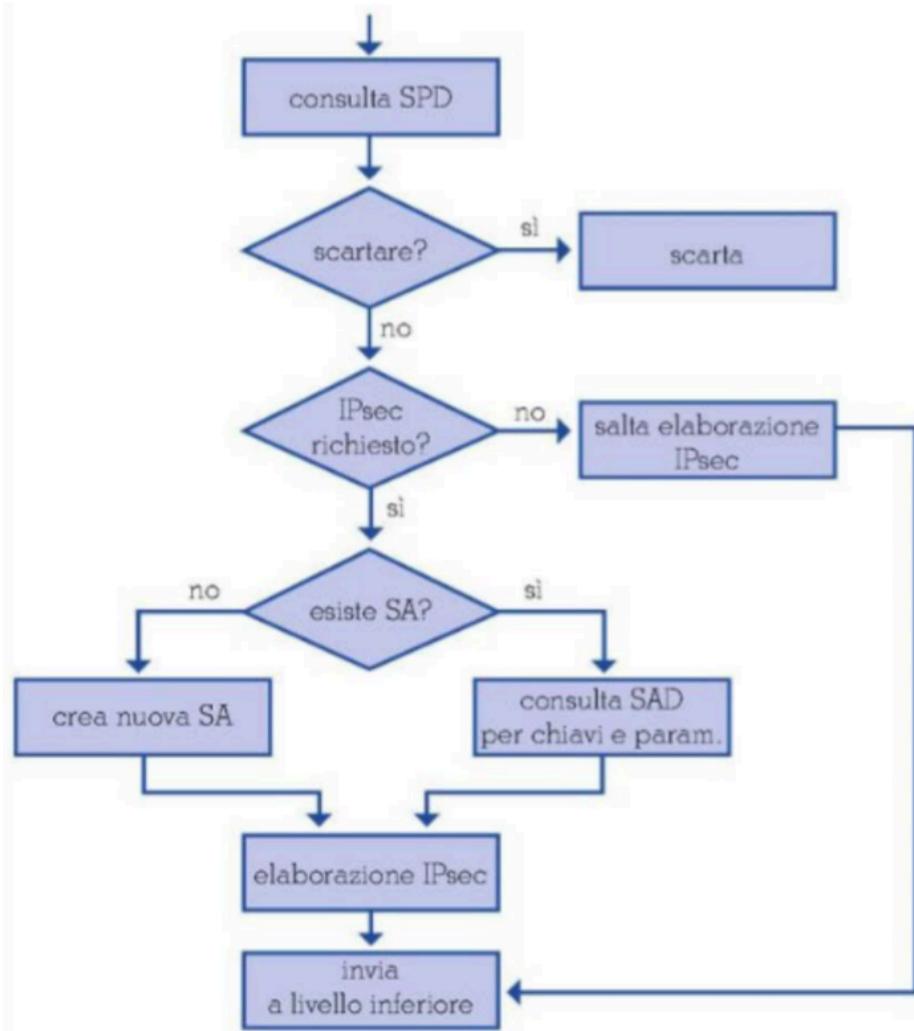
Tutte le **SA** (*Security Association*) attive su un host o su un *security gateway*⁷² sono contenute in un database detto **SAD** (*Security Association Database*).

Le politiche di sicurezza sono invece contenute in un altro database detto **SPD** (*Security Policy Database*). In una macchina che implementa IPsec tutto il traffico è soggetto alle politiche di sicurezza di IPsec per cui è necessario consultare il **SPD** (*Security Policy Database*) per decidere se un pacchetto debba essere lasciato passare normalmente, debba essere passato all'elaborazione IPsec, oppure debba essere scartato, e la scelta dei pacchetti su cui applicare le politiche di sicurezza si basa su dei parametri identificativi, come l'indirizzo IP sorgente o destinazione, la porta sorgente o destinazione, il protocollo di trasporto.

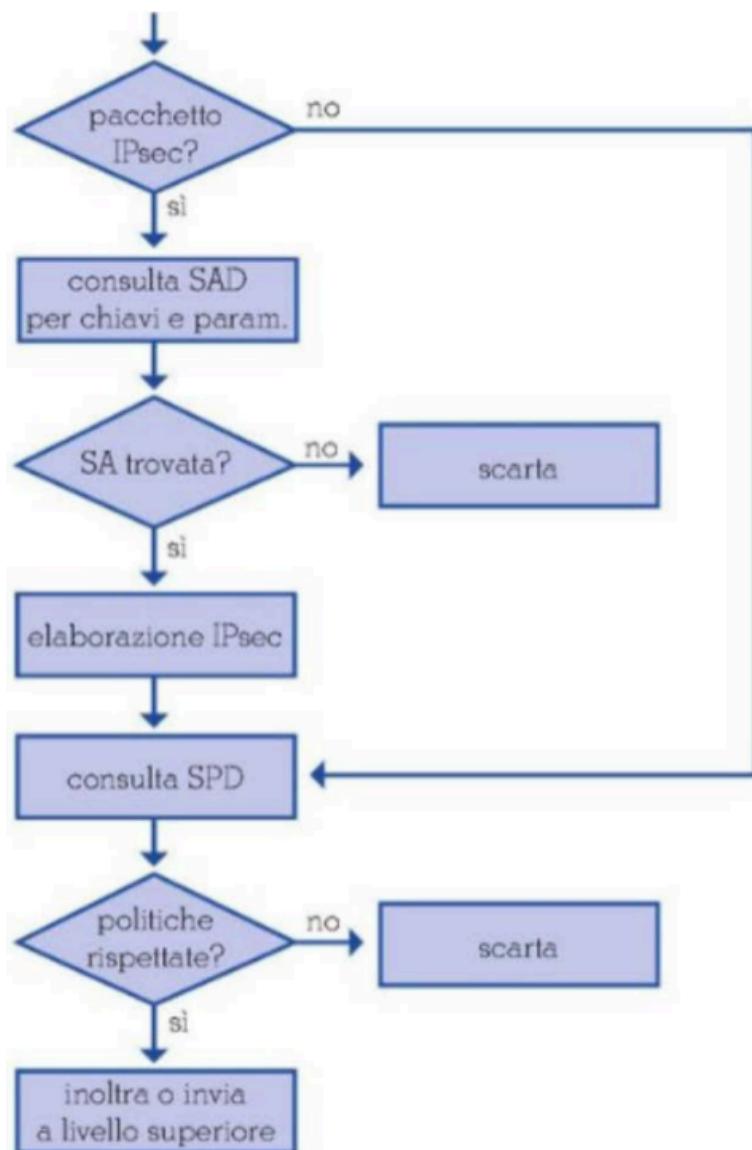
Nell'analizzare l'elaborazione dei pacchetti IPsec si distingue il traffico in uscita (*outbound*, ovvero diretto verso un'interfaccia di rete) dal traffico in entrata (*inbound*, ovvero proveniente da un'interfaccia di rete). Si noti che un *security gateway* può elaborare lo stesso pacchetto due volte: prima in entrata, proveniente da un'interfaccia, e poi in uscita, diretto verso un'altra interfaccia.

Per il traffico in uscita (*outbound*), bisogna innanzitutto cercare nel **SPD** (*Security Policy Database*) un selettore applicabile al pacchetto, usando i parametri identificativi. Se il pacchetto richiede un'elaborazione di tipo IPsec bisogna associarlo a una **SA** (*Security Association*) esistente, cercandola nel **SAD** (*Security Association Database*), oppure utilizzare **IKE** (*Internet Key Exchange*) per crearne una nuova. Poi si esegue l'elaborazione necessaria (autenticazione, cifratura, encapsulamento del pacchetto in uno nuovo in caso di modalità tunnel) e infine si passa il pacchetto al Livello inferiore.

⁷² Un *security gateway* è un *intermediate system* che gestisce le SA.



Per il traffico in ingresso (*inbound*), bisogna innanzitutto ricomporre il datagramma IP nel caso sia stato frammentato, dopo di che si identificano i pacchetti che devono essere elaborati con IPsec tramite la presenza dei valori identificativi di **ESP** (*Encapsulating Security Payload*) oppure di **AH** (*Authentication Header*) nel campo protocollo dell'header IP. Per i pacchetti IPsec, si identifica la **SA** (*Security Association*) relativa grazie al valore **SPI** (*Security Parameters Index*) presente nell'header **AH** (*Authentication Header*) o **ESP** (*Encapsulating Security Payload*), si applica l'elaborazione IPsec richiesta e si controlla il **SPD** (*Security Policy Database*) per accertarsi che le operazioni effettuate corrispondano alle politiche specificate. Infine il pacchetto viene inoltrato verso la destinazione successiva oppure passato al livello superiore.



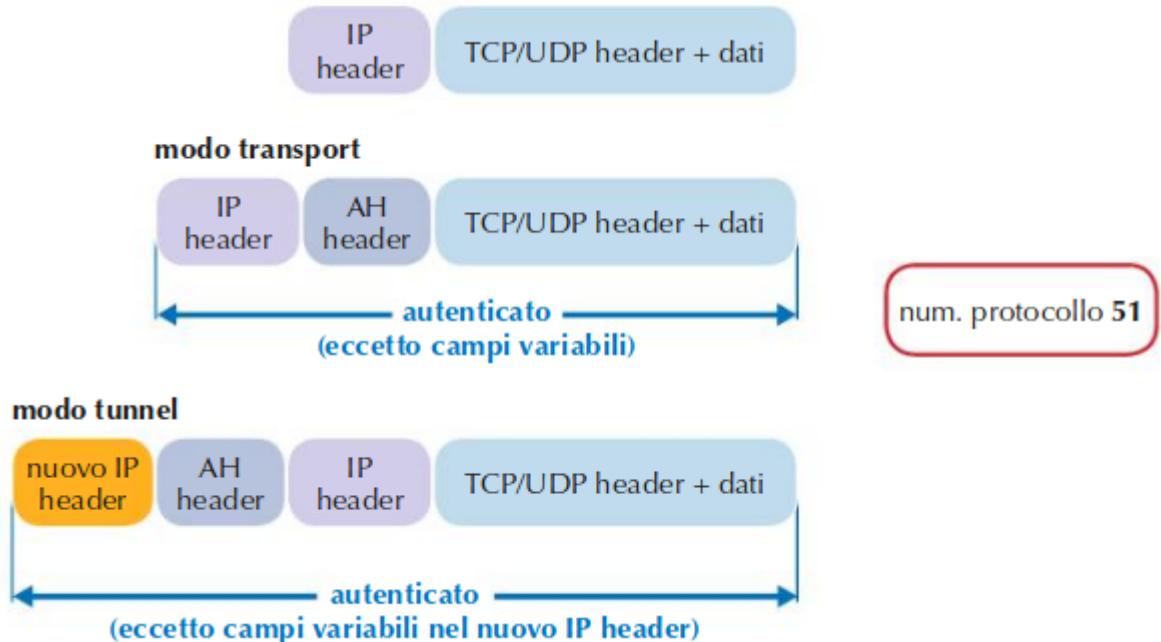
Le **SA** (*Security Association*) possono essere combinate tra loro, sia nel caso che i nodi terminali siano gli stessi sia nel caso siano diversi: per esempio si potrebbero avere due **SA** in modalità trasporto tra due host (una per **AH** e una per **ESP**), oppure una **SA** in modalità trasporto tra due host a cui si aggiunge una **SA** in modalità tunnel tra due security gateway che stanno tra i due host. Non c'è limite al numero di tunnel sovrapposti, mentre per la modalità trasporto è possibile applicare solamente una volta ciascuno dei protocolli **AH** e **ESP**, e in questo caso bisogna necessariamente applicare prima **ESP** e poi **AH**.

Authentication Header

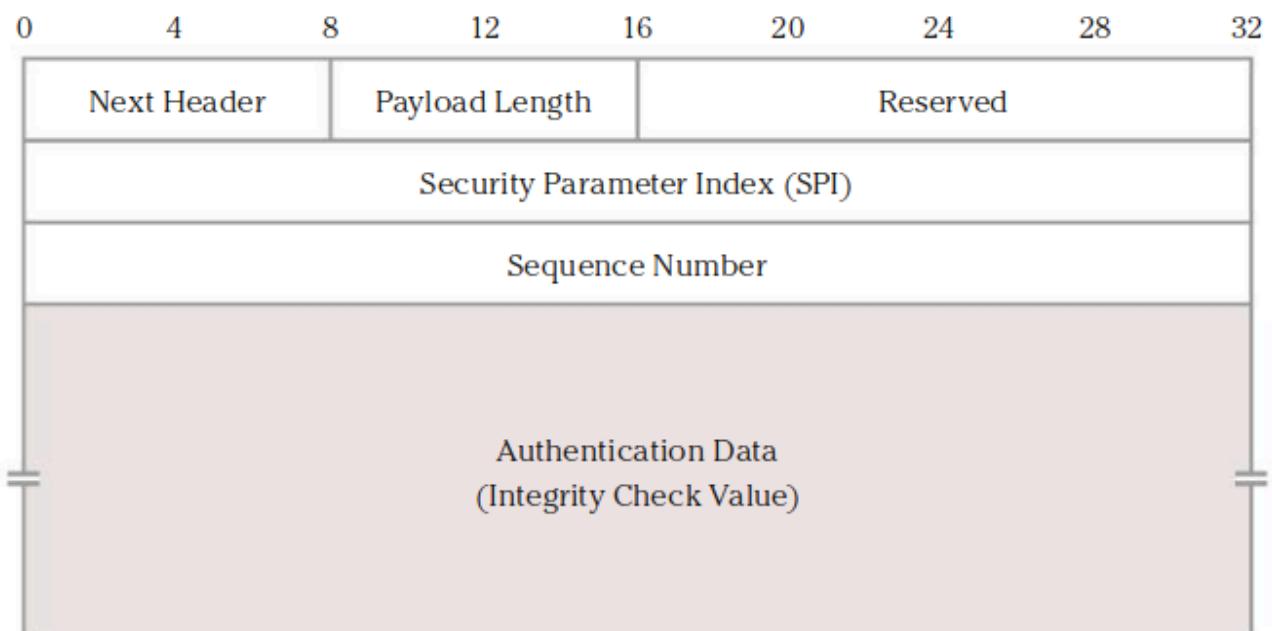
Authentication Header (AH) è un protocollo che garantisce l'autenticazione e l'integrità. Il software del mittente applica gli algoritmi di crittografia per "firmare" digitalmente il pacchetto in modo tale che il software del destinatario possa verificare l'identità del mittente e l'integrità dei dati.

Il mittente stabilisce una **SA** con il destinatario e inizia a inviare pacchetti sicuri inserendo un'intestazione **AH**.

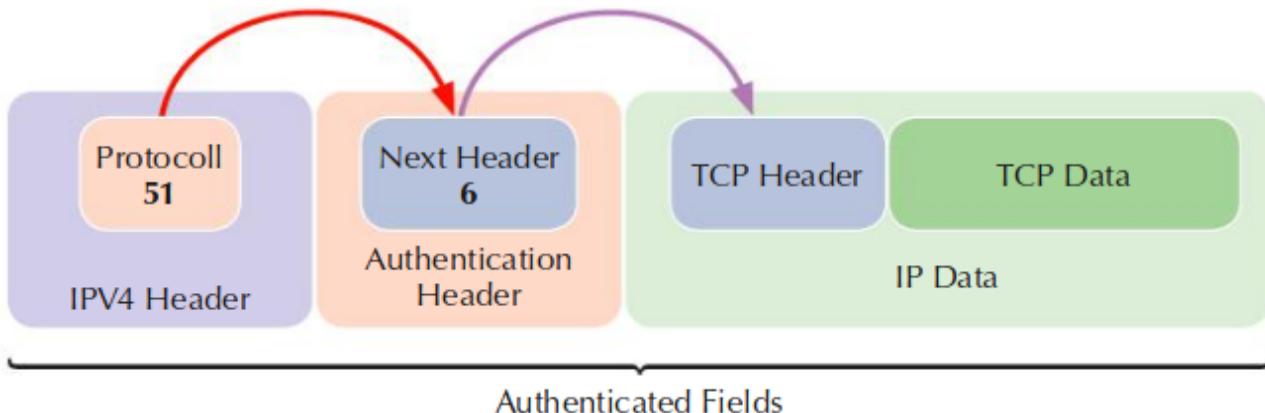
Nella figura seguente viene mostrato l'inserimento dell'intestazione AH nelle due modalità transport e tunnel. Il pacchetto originale viene incapsulato in un'intestazione AH e il tutto viene poi incapsulato con una intestazione IP, con numero di protocollo 51. Nel caso di modalità tunnel l'intestazione AH incapsula anche l'intestazione IP originale, che viene così intesa come dato, e nascosta.



L'intestazione AH, mostrata nell'immagine seguente, contiene i campi spiegati successivamente.



- **Next Header:** contiene il numero di protocollo dell'intestazione successiva a quella di AH. Nell'immagine seguente è mostrato come esempio, il caso della modalità transport.

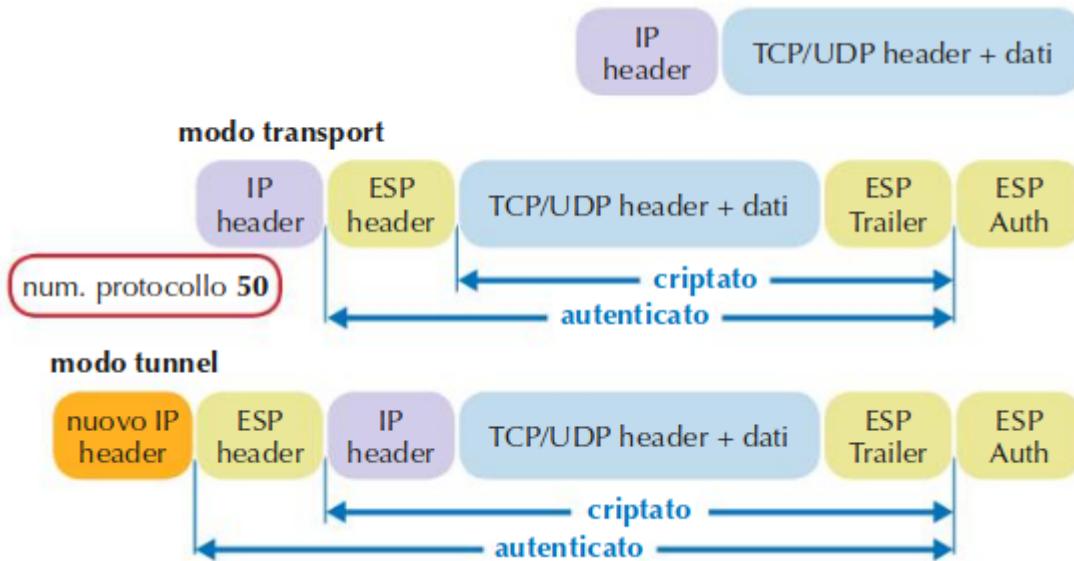


- **Payload Length:** definisce la lunghezza dell'header di *AH* in parole di 32 bit.
- **Reserved:** non usato.
- **SPI – Security Parameter Index:** numero di 32 bit che, con l'indirizzo IP di destinazione e il protocollo di sicurezza, identifica la security association da utilizzare per questo pacchetto.
- **Sequence Number:** contiene un numero sequenziale che, partendo da 0, è incrementato per ogni pacchetto appartenente a quella *SA*. Questo numero identifica in modo univoco ogni datagramma inviato, per impedire che lo stesso pacchetto possa essere rispedito se venisse catturato in un attacco di **reply** (replica) o di **man in the middle**, in cui un attaccante si interpone tra le due entità che sono in comunicazione e si impossessa del pacchetto per inoltrarlo successivamente.
- **Authentication Data:** ha lunghezza variabile e contiene lo **Integrity Check Value (ICV)**, un numero generato tramite un algoritmo di hashing, che costituisce la firma digitale realizzata con l'algoritmo specificato e concordato nella *SA*. La sintesi è calcolata sul pacchetto originale e assicura l'integrità e l'identità del mittente.

Encapsulating Security Payload

Encapsulating Security Payload (ESP) è un protocollo che, oltre a garantire l'autenticazione e l'integrità, garantisce la riservatezza.

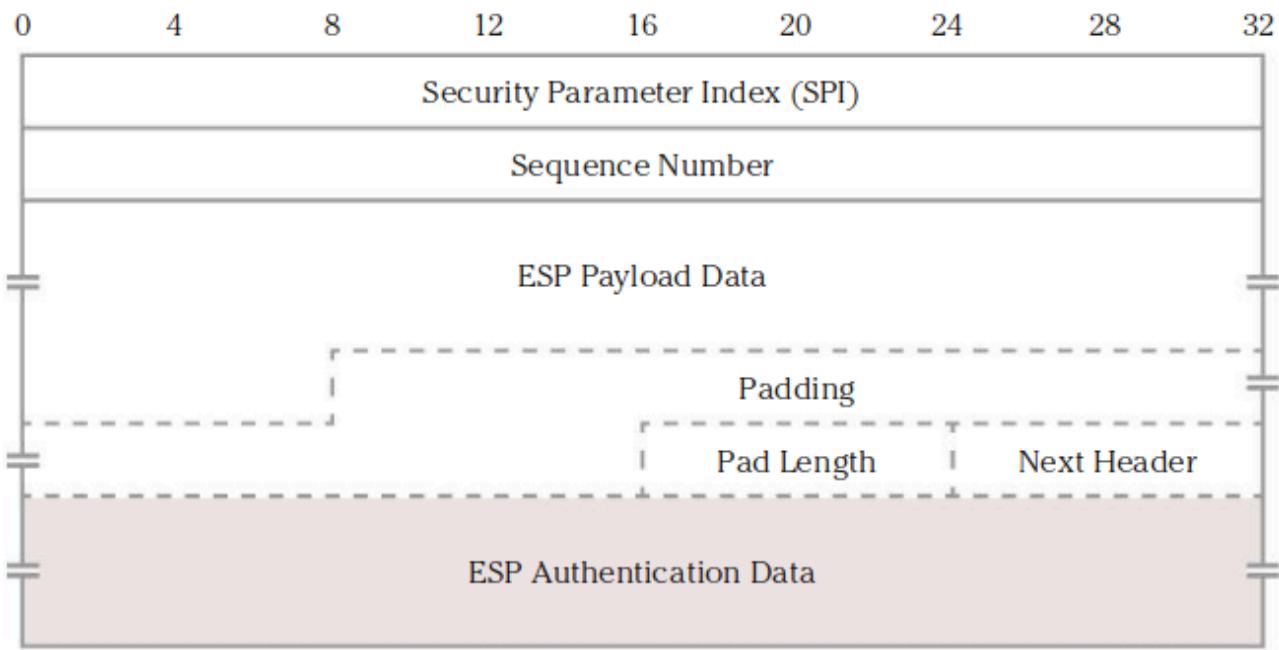
Il mittente stabilisce una *SA* con il destinatario e inizia a inviare pacchetti sicuri. Nell'immagine seguente è mostrato l'inserimento dell'intestazione e della coda *ESP* nelle due modalità transport e tunnel. Il pacchetto originale viene incapsulato in un'intestazione *ESP* e in una coda *ESP* seguita dai campi Autenticazione (opzionali), e il tutto viene poi incapsulato con un'intestazione IP con numero di protocollo 50. Nel caso di modalità tunnel, l'intestazione *ESP* incapsula anche l'intestazione IP originale, che viene così intesa come un dato e quindi nascosto.



L'intestazione *ESP*, mostrata nella figura che segue, sono presenti i seguenti campi:

- **SPI – Security Parameter Index:** come in *AH*, questo campo identifica la *SA* da usare per questo pacchetto.
- **Sequence Number:** come in *AH*, contiene un numero sequenziale che, partendo da 0, si incrementa per ogni pacchetto appartenente a quella *SA*.
- **ESP Payload data:** contiene i dati crittografati, tipicamente presenti nel pacchetto IP che è stato incapsulato.
- **Padding:** riempie il campo di payload per farlo diventare multiplo di 32 bit.
- **Pad Length:** fornisce il numero di ottetti aggiunti in padding.
- **Next Header:** contiene il numero di protocollo del successivo header.
- **Authentication Data:** contiene lo **Integrity Check Value (ICV)** calcolato applicando l'algoritmo di autenticazione (opzionale).

Si noti che la maggior parte dei campi è a lunghezza variabile. Le eccezioni riguardano i campi SPI, Sequence Number, che sono lunghi 4 byte, e Pad Length e Next Header, che sono lunghi 1 byte.



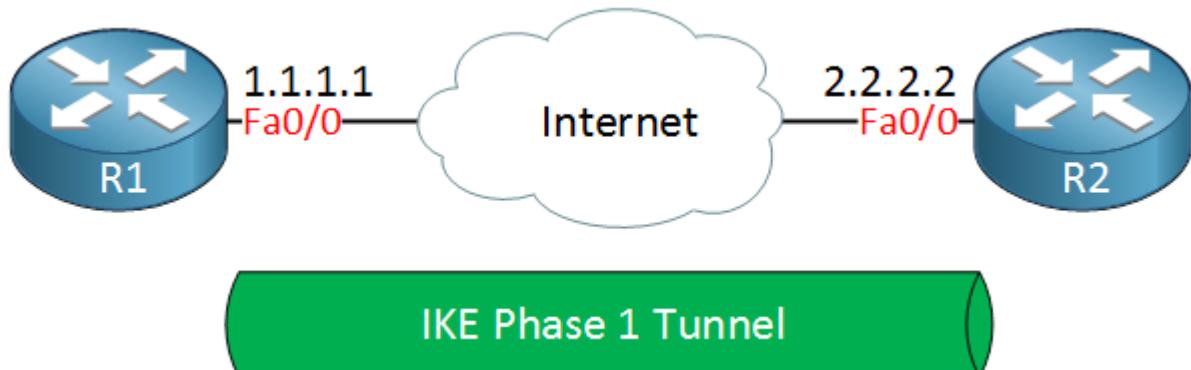
IKE (Internet Key Exchange)

Nell'architettura di IPsec è centrale il concetto di **Security Association**, ma né **AH** (*Authentication Header*) né **ESP** (*Encapsulating Security Payload*) si preoccupano della gestione delle **SA** (*Security Association*).

Le **Security Association** possono essere costruite manualmente o automaticamente, ma chiaramente una loro gestione manuale non è in genere praticabile se non in contesti molto limitati, per cui è necessario un meccanismo automatico.

Il protocollo **IKE** (*Internet Key Exchange*) realizza un collegamento *peer-to-peer* in due fasi.

1. Nella prima i due nodi creano una **Security Association** per **IKE** stesso, ovvero un canale sicuro da utilizzare per i messaggi di **IKE**:

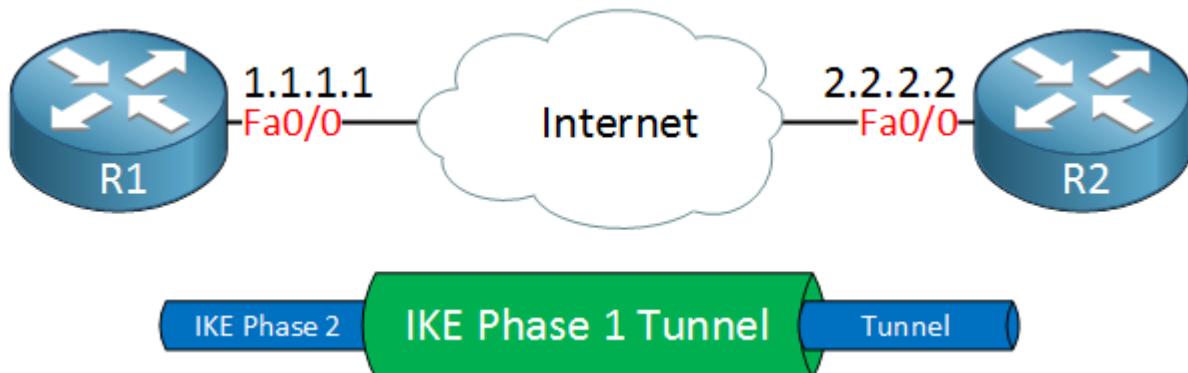


In questa fase vengono negoziati:

- **Hashing**: scelta dell'algoritmo di hash usato per verificare l'integrità (MD5 o SHA);
- **Autenticazione**: scelta di come garantire l'autenticità dei due peer (pre-shared key o certificati digitali);

- **Gruppo DH** (Diffie Hellman): il gruppo DH determina la robustezza della chiave scambiata, più alto è il gruppo, più sicuro è lo scambio di chiave (DH1, DH2, DH5);
- **Lifetime**: per quanto tempo sarà attivo il tunnel IKE fase 1. Minore è il lifetime maggiore è la sicurezza (valore di default 86400 secondi, cioè 1 giorno);
- **Crittografia**: scelta dell'algoritmo usato per cifrare (DES, 3DES o AES).

2. Nella seconda fase viene utilizzata la **SA** appena creata per negoziare delle **Security Association** per altri protocolli, come ad esempio per IPsec:

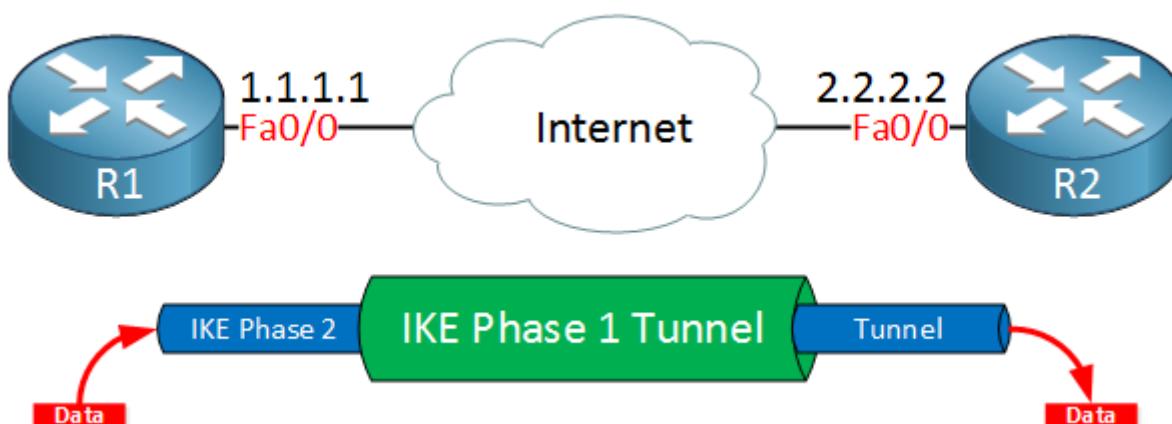


In questa seconda fase si negoziano le seguenti informazioni:

- **Protocollo IPsec**: si decide se usare AH o ESP;
- **Encapsulation Mode**: si decide se usare la modalità transport o tunnel;
- **Crittografia**: si sceglie l'algoritmo crittografico da usare (DES, 3DES o AES);
- **Autenticazione/Integrità**: si sceglie l'algoritmo di autenticazione da usare per la funzione hash MAC (MD5 o SHA);
- **Lifetime**: si decide per quanto tempo è valido il tunnel;
- **Scambio di chiavi DH**: viene effettuato lo scambio delle chiavi Diffie-Hellman.

Tutte queste negoziazioni avvengono all'interno del tunnel realizzato nella fase 1, quindi in maniera sicura.

Una volta terminata la seconda fase i dati verranno trasferiti attraverso il secondo tunnel:



IKE (*Internet Key Exchange*) fornisce quindi un **servizio di generazione dinamica di nuove chiavi**, che consiste nel generare chiavi di sessione diverse per ogni blocco di dati inviati. In questo modo, se un utente non autorizzato riuscisse a intercettare una parte della comunicazione e la relativa chiave di sessione, non potrebbe comunque avere tutte le informazioni perché la parte restante della comunicazione è criptata con chiavi diverse. È possibile gestire la frequenza di generazione delle chiavi o lasciare un valore di frequenza prestabilito.

Lo **scambio delle chiavi in IPSec** viene realizzato utilizzando l'algoritmo **Diffie-Hellman (DH)**, che è uno degli algoritmi più sicuri per lo scambio di chiave, infatti le due parti si scambiano pubblicamente le informazioni sulla generazione delle chiavi, che vengono ulteriormente protette con la firma di una funzione *hash*. Dopo aver ricevuto questi dati, mittente e ricevente possono generare la stessa chiave condivisa.

Si faccia attenzione a non confondere l'**autenticazione dell'identità dell'interlocutore**, che è quella di cui si preoccupa **IKE**, con l'**autenticazione della provenienza dei dati**, che è quella garantita dal **servizio di autenticazione di AH ed ESP**.

La prima, l'**autenticazione dell'identità dell'interlocutore**, viene realizzata tramite l'uso di un certificato digitale, o di una password condivisa, o della crittografia a chiave pubblica, ed è volta a garantire che l'interlocutore è chi effettivamente sostiene di essere.

La seconda, l'**autenticazione della provenienza dei dati**, è volta a garantire che i dati ricevuti provengono effettivamente dall'interlocutore identificato in precedenza.

VPN (Virtual Private Network)

La rete VPN

L'accesso da remoto alle risorse di una LAN pone innanzitutto dei problemi di sicurezza. Per questo motivo, a volte, le aziende preferiscono adottare costose linee dedicate per utilizzarle in proprio. Una soluzione alternativa, ormai molto diffusa, è quella di appoggiarsi alle *Virtual Private Network (VPN)*, reti private che sfruttano Internet come infrastruttura.

Poiché Internet è pubblica e aperta a chiunque, è relativamente semplice intercettare i dati. Per evitare questo pericolo, le VPN si servono di "tunnel virtuali" creati tra le varie sedi: i dati sono cifrati all'entrata del tunnel e decifrati all'uscita, in modo che possano viaggiare in modo sicuro e protetto.

Negli ultimi anni è aumentata l'esigenza di avere sedi aziendali distribuite su un vasto territorio geografico, così come la necessità di lavorare in mobilità. Per questa ragione è diventato strategico poter comunicare in sicurezza, come si lavora nella LAN aziendale.

La **VPN** risponde in pieno a queste esigenze e, in particolare, trova piena realizzazione in alcuni ambiti:

- collegamento tra sedi periferiche aziendali e con la sede centrale (LAN-to-LAN VPN o Site-to-Site VPN);
- collegamento alla LAN aziendale di un lavoratore remoto che opera in mobilità (Client-to-LAN VPN);
- collegamento a risorse aziendali da parte di consulenti, clienti e fornitori.

Le **VPN** offrono molti vantaggi:

- riduzione dei costi: non è necessario affittare delle linee dedicate (molto costose) per collegare tra loro le varie sedi;
- utilizzo di risorse già presenti: è possibile utilizzare le normali linee di collegamento a Internet già presenti nelle aziende;
- flessibilità di accesso alle risorse aziendali: con gli opportuni permessi di accesso, si può accedere a una risorsa della propria azienda indipendentemente dalla sede in cui ci si trova in quel momento;
- trasmissioni sicure: tutte le trasmissioni sono automaticamente cifrate;
- architettura facilmente scalabile: è molto semplice e veloce aggiungere ulteriori sedi alla rete aziendale;
- supporto dei servizi di nuova generazione compresi video e voce con protocollo di cifratura;
- semplicità d'uso.

Modalità di connessioni di una VPN

Un'azienda che ha diverse sedi dislocate anche a grande distanza tra loro, generalmente tenderà ad adottare una VPN come tecnologia per gestire l'insieme delle sedi remote come un'unica rete locale aziendale, cercando di estendere in ambito geografico la propria rete LAN privata e realizzando una WAN privata per il proprio business.

Una simile gestione permette inoltre di considerare anche forme alternative di lavoro come lo homeworking⁷³ e il teleworking⁷⁴, oltre alla possibilità di collaborare con partner consociati creando LAN estese che vanno oltre i confini della singola azienda.

Esistono due modalità di connessioni per una VPN:

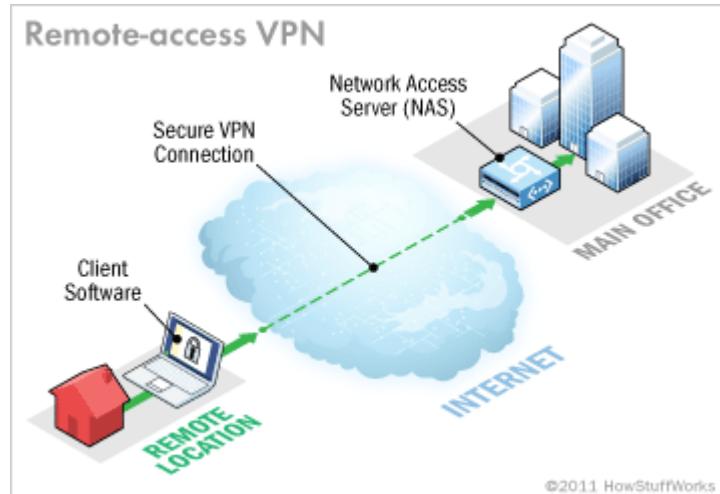
- **remote-access VPN** che permette ad un *teleworker* di emulare, tramite Desktop Remoto, il desktop dell'ufficio principale;
- **site-to-site VPN** che consente alle aziende di ampliare le risorse di rete alle filiali, agli uffici domestici e ai siti di partner.

Remote-access VPN

Una **Remote-Access VPN** consente ai singoli utenti di stabilire connessioni sicure con la LAN aziendale remota.

Gli utenti possono accedere alle risorse protette della rete locale lavorativa come se fossero direttamente collegati ai server della rete aziendale.

Ci sono due componenti indispensabili alla realizzazione di un accesso remoto VPN:



3. un **server di accesso** alla rete identificato con l'acronimo **NAS** (*Network Access Server*);
4. un **software VPN Client**.

Un **NAS** (*Network Access Server*) è un software che richiede all'utente di fornire le credenziali valide per accedere alla VPN. Un **NAS** funziona come un gateway per limitare gli accessi ad un dispositivo protetto, in questo caso la VPN, ma potrebbe essere un qualsiasi altro dispositivo o servizio. Quando un client si collega, il NAS effettua un collegamento ad un altro dispositivo, un **server AAA**, per verificare se le credenziali fornite dal client sono valide. Sulla base della risposta ricevuta il **NAS** autorizza o no l'accesso al dispositivo protetto. Il **NAS** non contiene informazioni su

⁷³ Lo *homeworker* svolge il proprio lavoro da casa (ufficio domestico), collegandosi alla rete aziendale.

⁷⁴ Il *teleworker* svolge il proprio lavoro collegandosi alla rete aziendale da qualsiasi luogo, usando il proprio dispositivo mobile.

quali client possono collegarsi o su quali siano le credenziali valide, infatti per autorizzare il client invia le credenziali ad un **server AAA** (*Authentication, Authorization, Accounting*) che sa come gestirle, tipicamente un server RADIUS. Per ogni connessione VPN, il **server AAA** conferma le credenziali del client che si è autenticato (**Authentication**), identifica ciò a cui il client può accedere tramite la connessione (**Authorization**) e tiene traccia di ciò che il client fa mentre è loggato (**Accounting**).

La componente del **software VPN Client** è necessaria per i dipendenti che desiderano utilizzare la VPN dal proprio computer, in quanto questo software stabilisce e mantiene una connessione alla rete VPN. La maggior parte dei sistemi operativi odierni sono dotati di software in grado di connettersi alle reti **Remote-Access VPN**, anche se alcune **VPN** potrebbero richiedere agli utenti di installare un'applicazione specifica.

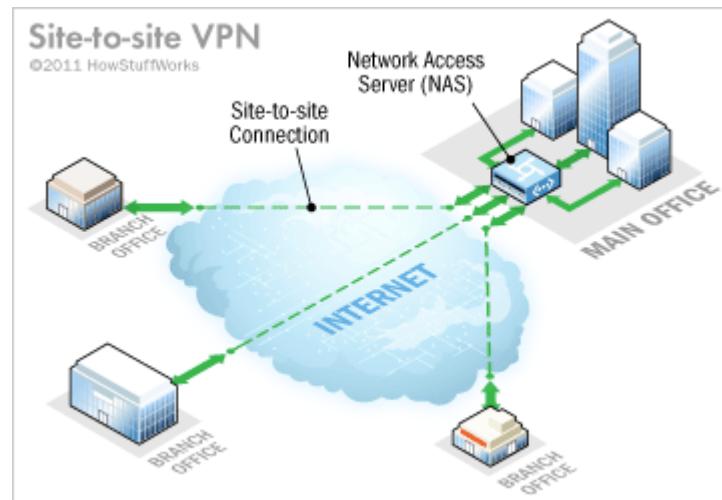
Un **Remote-Access VPN** è generalmente adatto per i singoli dipendenti o utenti, o per aziende con filiali costituite da piccoli uffici.

Site-to-site VPN

In aziende con filiali grandi e con decine o addirittura centinaia di dipendenti occorre affidarsi a un altro tipo di **VPN** che permettono di mantenere le aziende collegate LAN-to-LAN.

Un **Site-to-site VPN** permette di stabilire connessioni sicure attraverso una rete pubblica come Internet.

Il **Site-to-Site VPN** estende la rete aziendale, rendendo disponibili le risorse della sede principale alle sedi secondarie. Un'azienda in crescita con decine di filiali in tutto il mondo rappresenta il tipico esempio di un soggetto che ha bisogno di una **Site-to-site VPN**.



Ci sono due tipi di **Site to-site VPN**:

- **Intranet⁷⁵ based** utilizzata da società che hanno una o più sedi remote, ognuna con una propria LAN da riunire in un'unica rete WAN privata.
- **Extranet⁷⁶ based** usata da società che hanno rapporti di forte collaborazione con altre società, come ad esempio un partner fornitore o un'azienda cliente. In questi casi è possibile costruire una VPN extranet che collega le LAN delle diverse imprese, permettendo alle aziende di lavorare insieme in un ambiente

⁷⁵ Una **rete Intranet** è una rete interna aziendale che impiega le tecnologie e i protocolli di Internet.

⁷⁶ Una **rete Extranet** è una rete che impiega le tecnologie e i protocolli di Internet per collegare un'azienda ai propri fornitori, clienti o ad aziende consociate.

sicuro, condividendo le risorse e senza dover consentire l'accesso preventivo alla propria intranet.

Anche se lo scopo di un **Site-to-site VPN** è diverso da quello di un **Remote-access VPN**, i due tipi di VPN potrebbero utilizzare parte dello stesso software e gli stessi dispositivi. Idealmente però, un **Site-to-Site VPN** dovrebbe **eliminare in modo trasparente la necessità per ogni host di eseguire il software VPN Client**. Per raggiungere questo scopo vengono utilizzati alcuni dispositivi dedicati:

- **VPN Concentrator** è un dispositivo che sostituisce il **server AAA** installato su un server generico. L'hardware e il software lavorano insieme per stabilire il tunnel VPN e gestire un gran numero di connessioni simultanee.
- **VPN-enabled/VPN-optimized Router** è un tipico router delegato al traffico in rete ma con la caratteristica aggiuntiva di poter instradare i pacchetti utilizzando protocolli specifici per le VPN.
- **VPN-enabled Firewall** che rappresenta un firewall tradizionale adibito a proteggere il traffico tra le reti, ma con la caratteristica aggiunta di poter gestire il traffico utilizzando protocolli specifici per le VPN.
- **VPN Client** che è costituito da un software in esecuzione su un dispositivo dedicato che funge da tunnel-interfaccia per connessioni multiple. Il suo compito è anche quello di evitare che su ogni host sia in esecuzione un software VPN Client.

La sicurezza nelle VPN

Il fatto che le reti VPN abbiano un ambito geografico (WAN) e utilizzino la rete pubblica insicura (Internet), obbliga ad affrontare seri problemi legati alla sicurezza dei dati alla riservatezza delle trasmissioni concentrando l'attenzione su tre fattori cruciali: l'**autenticazione**, la **cifratura** e il **tunneling**.

L'**autenticazione** fornisce la verifica di accesso alla VPN solo ad utenti autorizzati ad usare i servizi della rete.

La **cifratura** permette di mantenere confidenziali le comunicazioni, oltre a garantire l'integrità e l'autenticità dei dati, e nell'ambito delle VPN possono essere usati un'ampia gamma di algoritmi crittografici simmetrici (3DES, CAST, IDEA, ecc.) per cifrare il traffico in rete. Sia l'algoritmo da usare, sia le chiavi segrete che l'algoritmo di cifratura utilizzato, sono concordate e scambiate tra mittente e destinatario attraverso protocolli di sicurezza, come ad esempio il protocollo **IKE** (*Internet Key Exchange*) nell'ambito di **IPSec**.

Il **tunneling** nelle VPN si riferisce a un processo di immissione di un intero pacchetto all'interno di un altro pacchetto prima di essere trasportato su Internet per introdurre funzionalità crittografiche. Il pacchetto esterno protegge il contenuto dalla vista del pubblico e assicura che il pacchetto si muova all'interno di un tunnel virtuale. **IPSec** è uno dei protocolli usati nell'ambito delle VPN per effettuare il **tunneling**.

Authentication, Authorization, Accounting

Le **reti VPN** sono **reti private**, e quindi **per potervi accedere occorre autenticarsi**.

Si definisce **autenticazione dell'identità il processo tramite il quale un sistema informatico, un applicativo o un utente, verifica la corretta**, o almeno presunta tale, **identità di un altro sistema informatico, applicativo o utente che vuole comunicare attraverso una connessione, concedendogli l'autorizzazione a usufruire dei relativi servizi associati**.

Come già detto la porta di accesso di un client alla sua VPN risulta essere un server **NAS** dotato di un processo di autenticazione che utilizza un **server AAA** dedicato a questo scopo. Solo dopo aver superato la fase di **autenticazione** si viene **autorizzati** ad accedere ai servizi della rete.

La VPN dovendo funzionare come la LAN aziendale, **richiederà l'impostazione di opportune autorizzazioni per l'accesso ai servizi della rete, specificando il dominio per ciascun utente creato** (*policy di servizio*). Alcuni di questi servizi, come la condivisione di risorse (dischi, stampanti ecc.), può essere autorizzata solo per il personale dell'azienda, mentre ai client VPN esterni può essere concesso di accedere ai servizi di navigazione Internet (HTTP, HTTPs) o di posta elettronica (SMTP, POP3, IMAP4, SMTPs, POP3s, IMAP4s). Altresì un rappresentante di commercio in visita a un cliente potrà collegarsi, dopo essersi autenticato, alla VPN aziendale attraverso Internet e accedere, autorizzato, alla banca dati aziendale e ai servizi per l'inoltro immediato dell'ordine di acquisto del cliente, contribuendo così a ottimizzare i tempi di consegna e garantendo la privacy.

Per controllare che non siano state effettuate azioni indesiderate e non autorizzate, occorre prevedere **meccanismi di Accounting**. Con **Accounting si intendono tutte le azioni volte a misurare e documentare le risorse concesse a un utente durante un accesso**. Ciò può includere la misura della durata della sessione di lavoro di un utente, oppure la misura del quantitativo di traffico dati inviati e ricevuti da un utente nella sessione di lavoro. Tali informazioni possono essere prodotte e memorizzate attraverso dei file log per poi essere successivamente utilizzate per applicare una tariffazione, per effettuare un controllo delle autorizzazioni, per una pianificazione della capacità o per fini statistici di analisi dei trend. Sostanzialmente dall'analisi dell'**Accounting** in un **server AAA** si possono trarre molte informazioni utili.

Cifratura

In termini di sicurezza spesso risulta fondamentale il rapporto fiduciario tra l'azienda che vuole creare una VPN e il fornitore di servizi Internet che gestisce l'infrastruttura pubblica, in quanto **il fornitore deve garantire la confidenzialità**.

Nella sicurezza informatica per confidenzialità, o riservatezza, si intende la protezione dei dati e delle informazioni scambiate tra un mittente e uno o più

destinatari nei confronti di terze parti. Tale protezione deve essere realizzata a prescindere dalla sicurezza del sistema di comunicazione utilizzata, anche e soprattutto quando il sistema di comunicazione utilizzato è intrinsecamente insicuro, come per esempio la rete Internet. Di conseguenza, per quanto riguarda la **cifratura**, le **VPN utilizzano un'ampia gamma di algoritmi di crittografia** (3DES, CAST, IDEA, ecc.) **per cifrare il traffico in rete.** Sia l'algoritmo da usare, sia le chiavi segrete che l'algoritmo stesso utilizza sono concordate e scambiate tra mittente e destinatario attraverso protocolli di sicurezza. Nello specifico caso delle **reti VPN**, viene soprattutto utilizzato il **protocollo Internet Key Exchange (IKE)** nell'ambito di **IPsec**. Il compito principale di **IKE** è proprio implementare lo "scambio delle chiavi" per cifrare i pacchetti. **IKE** automatizza la gestione delle chiavi e permette all'amministratore di gestire un maggior numero di reti sicure.

La maggior parte dei **protocolli per la sicurezza nelle VPN garantisce anche l'integrità e l'autenticità dei dati** e quindi, in pratica, che i pacchetti ricevuti non siano stati modificati durante la trasmissione e che provengano da fonte certa. L'integrità e l'autenticità vengono garantite mediante meccanismi di firma digitale e certificato digitale.

Tunneling

Lo **scopo dei protocolli di tunneling è aggiungere un livello di sicurezza al fine di proteggere ogni pacchetto nel suo viaggio su Internet.**

In realtà le **VPN possono essere realizzate sia in modalità trasporto, sia in modalità tunnel.**

Nella modalità trasporto hanno un ruolo fondamentale i software impiegati.

Immaginiamo un lavoratore mobile (*teleworker*) che deve collegarsi alla centrale attraverso Internet da qualsiasi punto del Mondo. Il suo dispositivo (notebook, palm, wap, ecc.) dovrà dotarsi di un software per VPN. Il collegamento potrà essere effettuato con qualsiasi ISP in quanto cifratura e decifratura dei dati verranno garantite dal software a bordo del notebook e dagli apparati riceventi presso la sede centrale. Internet lascerà in chiaro solamente le informazioni di instradamento IP.

Nella modalità tunnel hanno un ruolo fondamentale gli apparati, e in particolar modo i router e i firewall.

Questa tecnologia è quella tipica di un collegamento fra una filiale e la sede centrale. Gli apparati sono preposti a trasformare e codificare tutto il traffico fra gli end-point, mentre per gli utenti finali non vi è alcuna percezione delle trasformazioni applicate (trasparenza). In questa modalità i pacchetti di dati vengono inseriti in ulteriori pacchetti richiedendo una maggior performance alla rete.

Il termine **tunneling** si riferisce a un insieme di tecniche per cui **un protocollo viene incapsulato in un protocollo dello stesso livello o di livello superiore**: nel caso delle reti **VPN** viene inserito uno strato che introduce funzionalità crittografiche. Il **tunneling è dunque il processo di immissione di un intero pacchetto all'interno di un altro pacchetto prima di essere trasportato su Internet.** Il

pacchetto esterno protegge il contenuto dalla vista del pubblico e assicura che il pacchetto si muova all'interno di un tunnel virtuale. Tale stratificazione di pacchetti viene chiamata **incapsulamento**.

Gli host o i dispositivi di rete su entrambe le estremità del tunnel (*tunnel interface*), possono incapsulare i pacchetti in uscita e riaprire i pacchetti in entrata. Gli utenti (a un'estremità del tunnel) e il personale IT (a una o entrambe le estremità del tunnel) dovranno configurare le interfacce di cui sono responsabili per utilizzare un determinato protocollo di tunneling.

Un protocollo di tunneling, chiamato anche un protocollo di incapsulamento, è un modo standardizzato per incapsulare i pacchetti, e il pacchetto viaggia con lo stesso protocollo di trasporto che avrebbe utilizzato senza il tunnel. Come noto, il protocollo di trasporto definisce le modalità con cui ogni dispositivo trasferisce pacchetti attraverso Internet mediante il proprio ISP. Ciascun pacchetto interno mantiene però ancora il proprio protocollo (*passenger protocol*), come Internet Protocol (IP) o AppleTalk, che definisce come si viaggia sulle LAN poste a ciascuna estremità del tunnel.

I **protocolli usati per il tunneling** sono diversi e questi sono anche le tecnologie usate nella tipologia **Secure VPN**:

- IPsec (*IP security*):
 - *Encapsulating Security Payload* (ESP): fornisce autenticazione, confidenzialità e controllo di integrità del messaggio.
 - *Authentication Header* (AH): garantisce l'autenticazione e l'integrità del messaggio ma non offre la confidenzialità.
 - *Internet Key Exchange* (IKE): implementa lo scambio delle chiavi per realizzare un flusso cifrato.
- SSL/TLS (*Secure Sockets Layer/Transport Layer Security*):
 - garantisce confidenzialità e affidabilità delle comunicazioni su rete pubblica;
 - protegge da intrusioni, modifiche o falsificazioni;
 - si poggia sul solo protocollo di trasporto TCP.
- BGP/[MPLS](#) (*Border Gateway protocol/Multi-Protocol Label Switching*):
 - utilizzato su reti a commutazione di pacchetto, tipicamente IP;
 - le decisioni di instradamento, alle quali viene associata una etichetta, vengono prese in modo asincrono rispetto al trasporto del traffico (di solito il percorso viene stabilito prima della spedizione dei pacchetti) e si applicano ad una intera classe di destinazioni;
 - MPLS non può essere considerato un protocollo di rete, piuttosto è una tecnologia che all'interno delle reti potenzia il trasporto del traffico;
 - instrada più tipi di traffico (dati, voce, video) sullo stesso canale, consentendo di differenziare la banda di trasmissione in base al tipo di traffico e di aggirare le zone congestionate e i collegamenti interrotti.
- PPTP (*Point-to-Point Tunneling Protocol*):
 - sviluppato da Microsoft™;
 - assicura autenticazione, cifratura e compressione dei dati;
- IEEE 802.1Q (*Ethernet tagged VLAN*):

- aggiunge al frame Ethernet un tag contenente il VLAN Identifier corrispondente alla VLAN interessata al trasferimento;
- permette il trasferimento promiscuo di frame appartenenti a VLAN diverse utilizzando una sola interfaccia e un singolo collegamento di rete;
- SSH (*Secure SHell*) tunneling:
 - trasferisce qualsiasi dato in rete utilizzando una connessione cifrata;
- SOCKS (*SOCKetS*):
 - standard IETF definito nello RFC 1928;
 - proxy trasparente che permette di effettuare connessioni TCP dirette tra computer su due reti IP differenti nei casi in cui un instradamento diretto non sia disponibile.
- GRE (*Generic Routing Encapsulation*):
 - sviluppato dalla CISCO™;
 - crea un collegamento point-to-point virtuale in maniera che nessuno dei due punti si debba preoccupare dell'infrastruttura su cui passa la comunicazione.
- L2TP (*Layer 2 Tunneling Protocol*):
 - è un protocollo di livello 5 (*sessione*) che agisce però come un protocollo di livello 2 (*data-link*) usando pacchetti UDP per incapsulare i pacchetti L2TP e per mantenere una connessione point-to-point;
 - deve essere associato ad un altro protocollo per implementare autenticazione, confidenzialità e integrità dei dati (solitamente IPsec).
- L2TPv3 (*Layer 2 Tunneling Protocol version 3*):
 - evoluzione di L2TP creato come alternativa a MPLS.

Fra questi i **principali protocoli per la sicurezza** sono **IPsec (IP security)**, **SSL/TLS (Secure Sockets Layer/Transport Layer Security)** e **BGP/MPLS (Border Gateway Protocol/Multi-Protocol Label Switching)**.

Le **VPN IPsec-based, SSL/TLS-based e BGP/MPLS-based sono diventate le soluzioni VPN scelte dalla maggior parte delle aziende per collegare uffici remoti, utenti remoti e partner di business**, in quanto, con le loro differenze, forniscono comunicazioni sicure con diritti di accesso su misura per i singoli utenti, che si tratti di dipendenti, consulenti o partner, aumentano la produttività, ampliando la rete e le applicazioni aziendali, riducendo i costi delle comunicazioni e accrescendo la flessibilità.

Tipi di VPN

Il funzionamento delle **VPN**⁷⁷ è articolato e complesso, ma in generale il loro funzionamento può essere raggruppato in tre categorie distinte: **Trusted VPN**, **Secure VPN** e **Hybrid VPN**.

⁷⁷ [Virtual Private Network - Wikipedia](#)

Trusted VPN

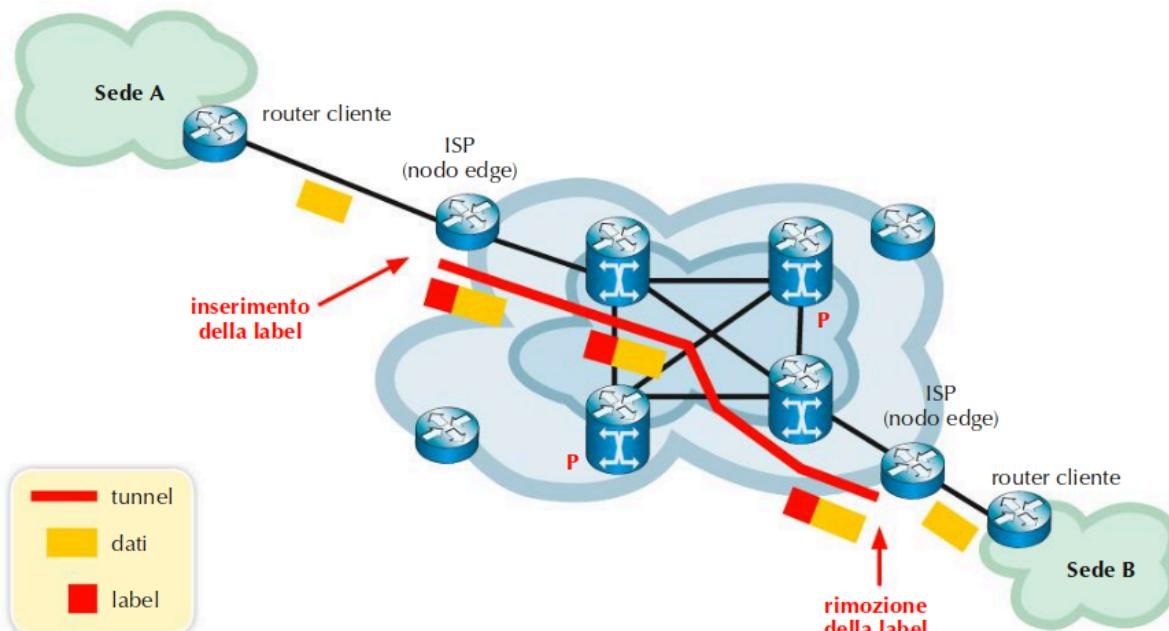
Una **Trusted VPN** garantisce che nessun terzo non autorizzato possa usufruire del circuito virtuale di comunicazione del cliente. Questo implica che il cliente abbia un proprio indirizzo IP e una propria politica di sicurezza.

Il circuito virtuale è instaurato attraverso uno o più "interruttori" di comunicazione che potrebbero essere compromessi da chi vuole disturbare il traffico della rete. Il cliente di una **Trusted VPN** si aspetta quindi che il fornitore del servizio, di solito il provider, mantenga l'integrità del circuito in modo da impedire l'accesso da parte di intrusi.

Le aziende che utilizzano una **Trusted VPN** vogliono avere la sicurezza che i loro dati si muovano attraverso una serie di **percorsi che hanno proprietà specifiche e che sono controllati da un ISP** (*Internet Service Provider*). Il cliente ha quindi fiducia che i percorsi attraverso i quali questi dati si muovono siano mantenuti sicuri secondo i criteri di un precedente accordo, anche se generalmente il cliente non conosce quali siano i percorsi utilizzati dal fornitore della **Trusted VPN**.

Più recentemente i fornitori di servizio hanno cominciato a offrire un nuovo tipo di **Trusted VPN**, questa volta usando Internet invece della rete telefonica come substrato di comunicazione. Queste nuove **Trusted VPN** non offrono sicurezza, ma danno ai clienti un modo di creare facilmente segmenti di rete su vasta scala (WAN). I segmenti **Trusted VPN** possono essere inoltre controllati da un posto unico e spesso con una qualità di servizio garantita (**QoS - Quality of Service**) dal provider.

In una **Trusted VPN** il pacchetto che parte dal router della sede A viene inoltrato a quello del fornitore (*nodo Edge*) su cui è implementato un protocollo che crea un circuito virtuale (**MPLS** - *MultiProtocol Label Switching*). Al pacchetto viene aggiunta, tra l'header di livello 2 (PPP o Ethernet) e quello di livello 3 (IP), una "label" (una nuova intestazione) che lo identifica e gli permette di viaggiare, passando tra i router di Core, fino ad arrivare al router dell'ISP di destinazione, che rimuove la "label" e consegna il pacchetto al router dell'utente.



Secure VPN

In una **Secure VPN** i dati viaggiano in un tunnel virtuale protetto sfruttando i protocolli IPSec e TLS/SSL, che garantiscono la riservatezza, l'autenticazione e l'integrità. Non viene però garantito il percorso seguito come avviene nella Trusted VPN.

In particolare, le VPN che utilizzano TLS/SSL consentono agli utenti di collegarsi, tramite browser, da qualsiasi postazione Internet, per stabilire connessioni sicure tra gli utenti remoti e le risorse interne alla rete privata. Queste applicazioni "clientless VPN" o "Web VPN", a differenza di quanto accade nei sistemi che sfruttano IPSec, non necessitano di particolari privilegi, come l'apertura di specifiche porte TCP, infatti l'utente, dopo l'autenticazione, può accedere in sicurezza alle risorse della LAN.

Il motivo principale per cui le società usano una **Secure VPN** è che possono trasmettere informazioni delicate su Internet senza temere che vengano intercettate.

Le **Secure VPN** sono particolarmente utili per permettere accessi remoti da parte di utenti connessi a Internet da zone non controllate dall'amministratore di rete. Una **Secure VPN** che usa ad esempio IPSec potrebbe essere la scelta migliore nel caso ci si appoggi alla connessione pubblica di Internet, in quanto fornisce una comunicazione confidenziale relativamente a basso costo. È il caso, ad esempio, di un esercente che può aver bisogno occasionalmente di una connessione Secure VPN con un fornitore, oppure per un teleworker che necessiti di connettersi alla rete aziendale.

Hybrid VP

La **Hybrid VPN**⁷⁸ combina la garanzia dei percorsi della Trusted VPN con la garanzia dei servizi di sicurezza di Secure VPN.

Una **Hybrid VPN** combina l'uso di **MPLS** (*Multiprotocol Label Switching*) e di **IPsec** (*Internet protocol security*) nella stessa **VPN**. Di solito queste due tipologie di **VPN** sono usate in ambiti diversi, ma è possibile combinarle usando la **IPSec VPN** come supporto alla **MPLS VPN**.

IPSec VPN (*Secure VPN*) richiede l'uso di dispositivi presso l'utente finale, tipicamente un router o un dispositivo di sicurezza multipurpose, e viene usata per criptare i dati all'interno di un **tunnel VPN**. Invece **MPLS VPN** (*Trusted VPN*) è fornita dallo ISP, utilizzando quindi la rete e i dispositivi dello ISP stesso. Per connettere le due tipologie di VPN è necessario l'uso di un gateway che permetta di collegare il tunnel IPSec con la MPLS VPN, mantenendo però sempre la sicurezza della VPN per tutto il percorso.

Una **Hybrid VPN** risulta facilmente gestibile se una singola filiale remota necessita di collegarsi alla sede centrale. Se invece più filiali necessitano di un collegamento vicendevole, oltre che con la sede centrale, la gestione dei vari tunnel IPSec

⁷⁸ [What is a hybrid VPN? - AT&T Business](#)

risulterebbe complesso, mentre il protocollo MPLS della Trusted VPN gestisce naturalmente queste situazioni di rete mesh.

TLS/SSL

La sicurezza delle reti

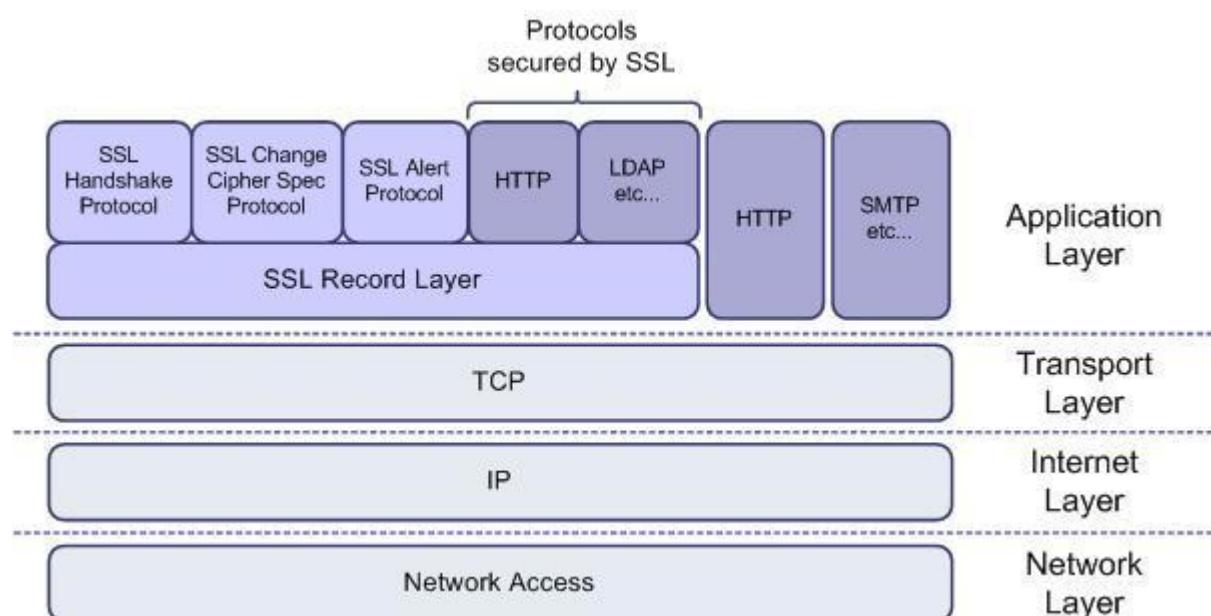
Libro vol.3 - La sicurezza delle connessioni con SSL/TLS

STUDIARE: L. Lo Russo, E. Bianchi, *Sistemi e Reti - Nuova Edizione OPENSCHOOL - Per l'articolazione INFORMATICA degli Istituti Tecnici settore Tecnologico*, vol. 3, ed. Hoepli, 2017.

- UdA4 - La sicurezza delle reti
 - Lezione 2 - La sicurezza delle connessioni con SSL/TLS
 - Generalità pp.183-184
 - Il protocollo SSL/TLS pp.184-186
 - Il funzionamento di TLS pp.186-187
 - Conclusioni pp.188-190

TLS - Transport Layer Security

Il **Transport Layer Security (TLS)** e il suo predecessore Secure Sockets Layer (SSL) ormai deprecato, sono dei protocolli crittografici che permettono una comunicazione sicura dal sorgente al destinatario (end-to-end) su reti TCP/IP fornendo **autenticazione, integrità** dei dati e **cifratura, operando al di sopra del livello di trasporto**. In questo modo viene garantita una comunicazione sicura a tutte le applicazioni che si appoggiano su **TLS/SSL**.

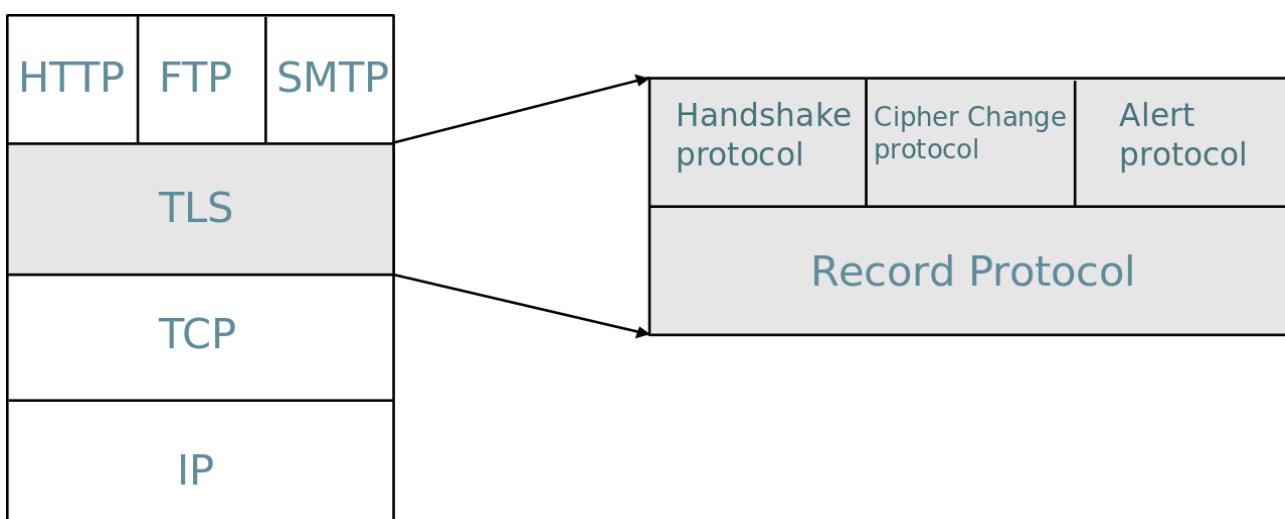


Diverse versioni del protocollo sono ampiamente **utilizzate in applicazioni come i browser, l'e-mail, la messaggistica istantanea e il voice over IP**. Un esempio di applicazione di TLS/SSL è nel protocollo **HTTPS**.

Il protocollo TLS/SSL garantisce:

- **segretezza** tramite l'utilizzo di una **chiave simmetrica**, ad esempio DES, 3DES, AES, IDEA o RC4;
- **autenticazione** tramite lo scambio di **certificati digitali** e l'uso di un **sistema a chiave pubblica**, come ad esempio **RSA**;
- **integrità** utilizzando un **MAC** (*Message Authentication Code*) tramite l'uso di funzioni hash come SHA o MD5.

Lo stack del protocollo **TLS/SSL** è il seguente:



Esamineremo lo **Handshake Protocol** e il **Record Protocol** che sono alla base della suite e che svolgono le seguenti funzioni:

- Fase 1: **handshake protocol per stabilire la sessione**;
- Fase 2: **TLS record protocol per trasferire i dati**.

Una sessione è un'associazione tra un client e un server (definisce i parametri crittografici che possono essere condivisi tra più connessioni). Le sessioni servono per evitare costose negoziazioni di nuovi parametri di sicurezza per ogni connessione.

Fase 1: Handshake Protocol

Gli obiettivi principali dello **Handshake Protocol** sono i seguenti:

- autenticazione del server al client e optionalmente autenticazione del client al server;
- negoziazione dei parametri come l'algoritmo di cifratura, la funzione di hashing, le chiavi di sessione.

Lo **Handshake Protocol** si basa su crittografia asimmetrica per lo **scambio dei parametri iniziali ed è responsabile della negoziazione dei parametri di sicurezza di una sessione.**

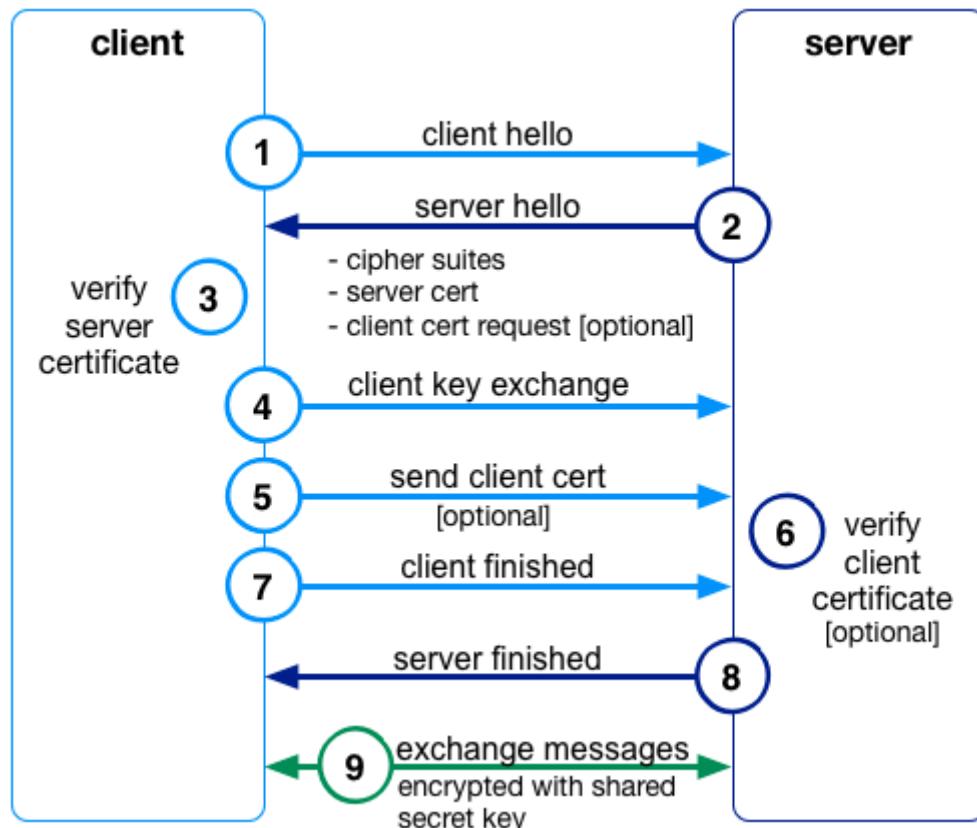
Questa prima fase viene avviata dal client che invia un messaggio "Hello" proponendo la versione del protocollo TLS da usare, le funzioni di cifratura e di hash che è in grado di gestire. Il server risponderà proponendo le funzioni più sicure che implementa, quindi eventualmente con funzioni che non aveva inizialmente proposto il client. Infine il server invierà anche il proprio certificato per autenticarsi verso il client.

L'invio del certificato digitale è importante perché in questo modo il client potrà verificare la validità della chiave pubblica del server tramite la certificazione di una *Certification Authority*, la cui affidabilità dovrà essere validata dal client risalendo la catena delle CA e validandone il relativo certificato digitale.

Se il client verificherà la validità del certificato ricevuto dal server, genererà una chiave di sessione K e la invierà cifrata al server usando la chiave pubblica del server. Nel caso in cui sia necessario autenticare anche il client, quest'ultimo invierà il proprio certificato digitale.

Infine il server la decifrerà la chiave simmetrica condivisa di sessione con la quale potrà avviare la comunicazione cifrata.

Lo schema generale è proposto nella figura seguente.

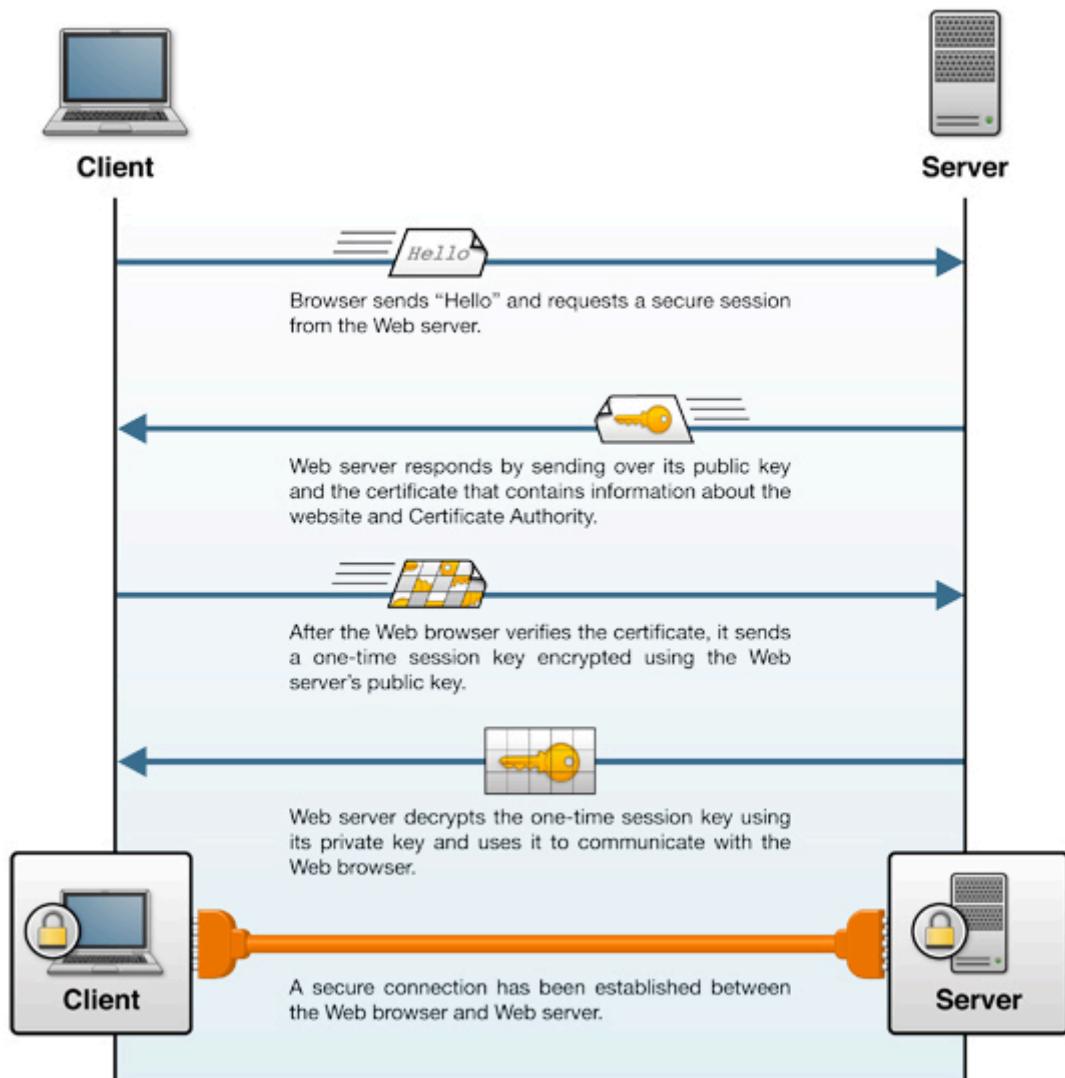


Per iniziare lo handshake il client deve effettuare una richiesta di connessione al server (1). Il server risponderà inviando il suo certificato (2) che il client dovrà verificare per autenticare il server (3).

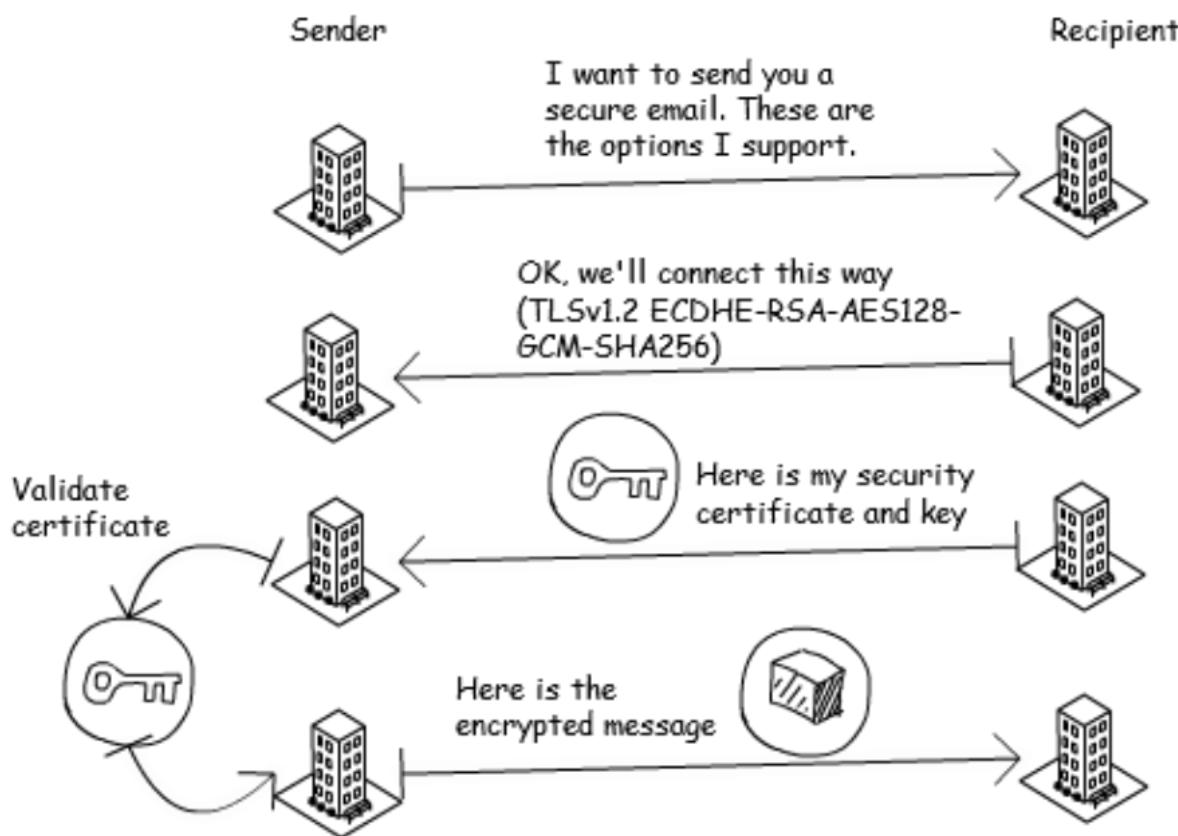
Opzionalmente il server può richiedere al client di autenticarsi inviandogli il suo certificato per raggiungere un livello di sicurezza maggiore (5)(6).

Il client genererà una nuova chiave simmetrica di sessione e, usando la chiave pubblica del server il client la cripterà e la invierà al server (4), avendo la garanzia che solo il server potrà decriptarla usando la sua chiave privata. A questo punto solo il client e il server saranno a conoscenza della chiave simmetrica di sessione e potranno concludere la fase di handshake (7)(8), per iniziare la comunicazione crittata dei dati (9).

La fase di handshake tra un browser e un server Web prevederà i seguenti passi:



Analogamente lo handshake per la posta elettronica prevede le seguenti fasi:



Fase 2: Record Protocol e trasferimento dati

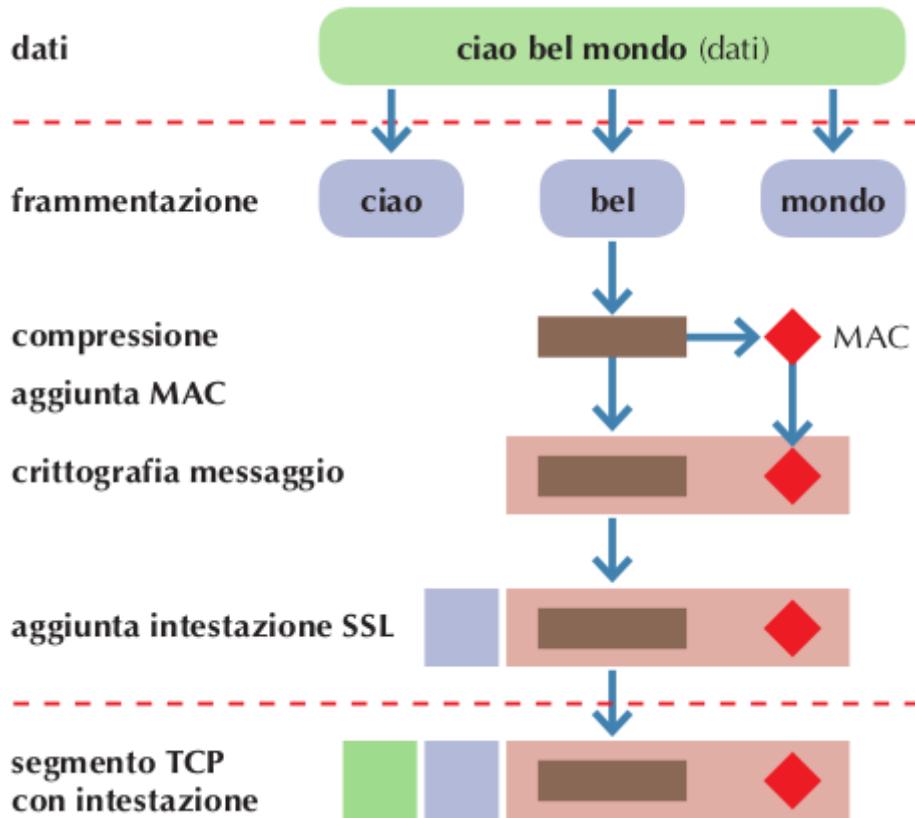
Il **TLS Record Protocol** garantisce sicurezza ed integrità, infatti, una volta concluso lo handshake il canale di comunicazione risulterà sicuro e i dati potranno essere trasferiti in modo cifrato utilizzando la chiave simmetrica di sessione K.

In altre parole TLS si basa sulla chiave di sessione definita nella fase di handshaking per cifrare con crittografia simmetrica.

In trasmissione i dati ricevuti da TLS da parte di un'applicazione vengono suddivisi in blocchi della stessa dimensione, eventualmente vengono compressi, se ne calcola il MAC (*Message Authentication Code*) che garantisce l'integrità, il blocco compresso con il MAC viene criptato con la chiave simmetrica di sessione, si aggiunge lo header TLS e infine passato al livello di trasporto per essere inviato.

In ricezione i blocchi in arrivo a TLS dal livello Trasporto vengono decriptati, se ne ricalcola il MAC e lo si confronta con quello ricevuto per verificarne l'integrità, si decomprime il blocco dati e infine quest'ultimo viene passato all'applicazione destinataria con tutti gli altri blocchi che componevano il messaggio originale.

Lo schema generale del funzionamento del **Record Protocol** è il seguente:



S/MIME e posta elettronica

La sicurezza delle reti

Libro vol.3 - La sicurezza nei sistemi informativi

STUDIARE: L. Lo Russo, E. Bianchi, *Sistemi e Reti - Nuova Edizione OPENSCHOOL - Per l'articolazione INFORMATICA degli Istituti Tecnici settore Tecnologico*, vol. 3, ed. Hoepli, 2017.

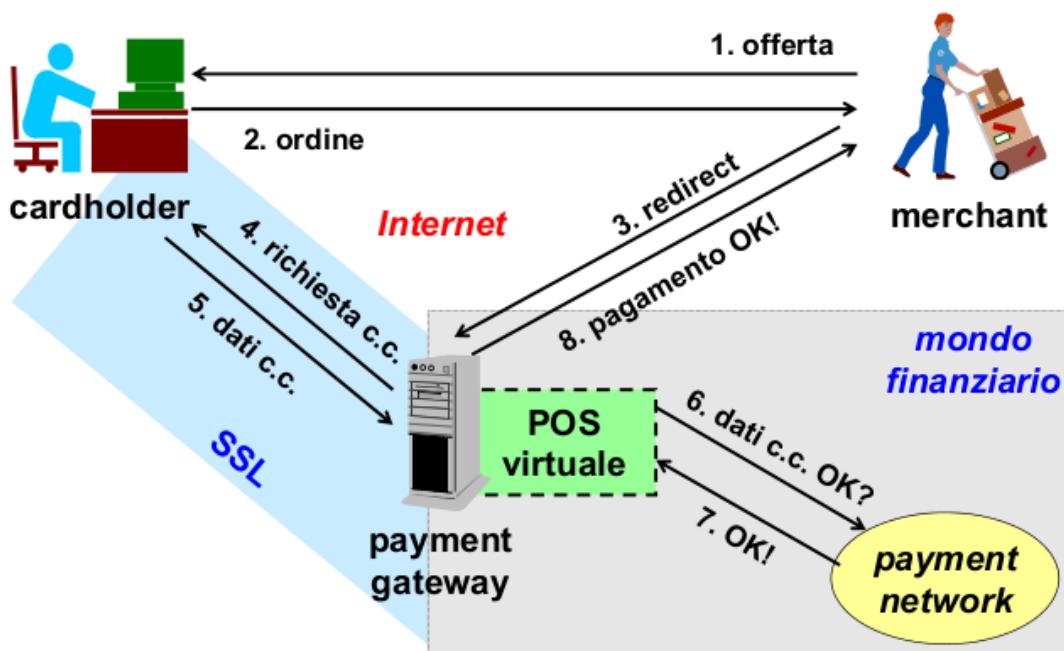
- UdA4 - La sicurezza delle reti
 - Lezione 1 - La sicurezza dei sistemi informativi.
 - La posta elettronica pp.175-176
 - Il protocollo S/MIME per la posta elettronica pp.176-179
 - Un software per la posta sicura: PGP pp.179-182
- [S/MIME](#) - [Wikipedia](#)

Sistemi di pagamento elettronico

L'uso dei **sistemi di pagamento elettronico** tramite carte di credito usando un canale TLS come sistema di protezione sta ormai diventando comune negli acquisti via Web.

L'ente **Payment Card Industry Data Security Standard Council** è l'ente che gestisce il **Payment Card Industry Data Security Standard** (PCI DSS) che è lo standard utilizzato per migliorare i controlli relativi ai dati forniti dal **cardholder** al fine di ridurre le frodi relative ai pagamenti via Web tramite carte di credito. Ad oggi questo standard viene richiesto per la gestione di tutti i pagamenti via Internet usando la carta di credito, essendo più prescrittivo rispetto ad altre tecniche.

L'immagine seguente mostra l'architettura alla base di queste forme di pagamento.



I pagamenti elettronici nel Web vengono svolti seguendo otto fasi:

1. il **merchant**, ovvero il negoziante online, pubblica un'offerta della merce che vuole vendere;
2. l'offerta attira l'attenzione di un **cardholder**, un possessore di carta di credito, che decide di effettuare un ordine;
3. il **cardholder** viene reindirizzato al **payment gateway** dal sito del **merchant**;
4. tramite **TLS** il **payment gateway** chiede i dati della carta di credito al **cardholder**;
5. il **cardholder** risponde fornendo i dati, sempre usando **TLS**;
6. il **POS virtuale** che si interfaccia tra il mondo di Internet e il mondo finanziario, chiede alla **payment network** se i dati ricevuti dal **cardholder** sono corretti;
7. il **payment network** invia la risposta sulla correttezza o meno dei dati del **cardholder**;

8. se i dati sono corretti si procede con il pagamento, il **merchant** viene informato dell'avvenuto pagamento e può quindi spedire la merce.

Nella configurazione illustrata il **payment gateway** è l'entità che possiede tutte le informazioni sulla transazione, sia quelle relative al pagamento, sia quelle relative alla merce, mentre il **merchant** avrà solo le informazioni relative alla merce, non essendo suo compito validare la bontà delle credenziali fornite dal **cardholder**.

L'ipotesi base affinché un pagamento elettronico vada a buon fine è che l'acquirente possieda una carta di credito e che abbia un browser con TLS. La sicurezza effettiva però dipende dalla configurazione e dalla sicurezza sia del server sia del client, infatti per avere una rete protetta è necessario installare e gestire una configurazione con firewall e non usare password di sistema predefinite o altri parametri di sicurezza impostati dai fornitori dei sistemi di sicurezza. Per proteggere invece i dati dei titolari delle carte di credito è sempre opportuno memorizzarli con attenzione e criptarli quando vengono trasmessi attraverso reti pubbliche aperte. Inoltre bisogna sempre rispettare un programma per la gestione delle vulnerabilità, di conseguenza bisogna usare e aggiornare con regolarità l'antivirus, ma anche sviluppare e mantenere protetti applicazioni e sistemi. In particolar modo è necessario implementare [misure forti per il controllo degli accessi](#), si deve consentire l'accesso ai dati solo se questo è effettivamente indispensabile per lo svolgimento dell'attività commerciale ([Principle of least privilege](#)) e di conseguenza limitare la possibilità di accesso fisico ai dati, chiudendo i server a chiave e introducendo l'uso di un ID univoco per ogni utente del sistema informativo.

Infine bisogna monitorare e testare le reti con regolarità, tenendo traccia di tutti gli accessi effettuati alle risorse della rete e ai dati dei titolari delle carte, ma anche eseguendo test periodici dei processi e dei sistemi di protezione. Quindi per mantenere un sistema sicuro bisogna stilare l'elenco delle regole che si vuole adottare e soprattutto **osservare rigorosamente le regole impostate dalla politica di sicurezza creata**, senza l'osservanza di quest'ultima importante regola tutte le altre risulterebbero inutili meno.

F - Sistemi distribuiti e tecniche di amministrazione

Modelli distribuiti per i servizi di rete e amministrazione dei sistemi

Libro vol. 3 - Modello client-server e distribuito per i servizi di rete

STUDIARE: L. Lo Russo, E. Bianchi, *Sistemi e Reti - Nuova Edizione OPENSCHOOL - Per l'articolazione INFORMATICA degli Istituti Tecnici settore Tecnologico*, vol. 3, ed. Hoepli, 2017.

- UdA6 - Modello client-server e distribuito per i servizi di rete
 - Lezione 1 - Le applicazioni e i sistemi distribuiti pp.298-304
 - Lezione 2 - Architetture dei sistemi Web pp.305-311
 - Lezione 3- Amministrazione di una rete pp.312-323
 - Lezione 4 - Active Directory pp.324-335 (*le parti di configurazione sono solo da leggere*)
 - Lezione 5 - Il troubleshooting pp.336-343 (*leggere*)
 - Lezione 6 - La sicurezza della rete pp.344-354

SSH Secure Shell

SSH (*Secure SHell*) è un protocollo di rete che **permette di stabilire una sessione remota cifrata tramite interfaccia a riga di comando con un altro host in rete**. È il protocollo che ha **sostituito** l'analogo ma insicuro [Telnet](#)⁷⁹.

SSH si basa su Diffie-Hellman e utilizza la **crittografia simmetrica per cifrare**, mentre usa la **crittografia asimmetrica per** risolvere l'**autenticazione** tra client e server. Il meccanismo di autenticazione del server e del client, quest'ultima quando fatta con la crittografia asimmetrica, non risultano però sicure nei confronti di un attacco man-in-the-middle dato che la chiave usata per verificare la firma dello host è solitamente autocertificata, non utilizzando una *Certification Authority*.

Il client **SSH** ha una **interfaccia a riga di comando** simile a quella di Telnet, ma **l'intera comunicazione avviene in maniera cifrata**, sia per quanto riguarda la mutua autenticazione tra client e server, sia la sessione di lavoro in cui vengono scambiati i comandi.

Il protocollo **SSH** è un **protocollo del livello Applicazione che usa TCP a livello di trasporto**. Il client ed il server sono installati o installabili su molte versioni di UNIX, GNU/Linux, Mac OS X e Microsoft Windows e **il server SSH risponde di default sulla porta 22**.

La **sintassi per avviare una sessione SSH** su sistemi UNIX-like è la seguente:

```
$ ssh [opzioni] nomeutente@host [comando]
```

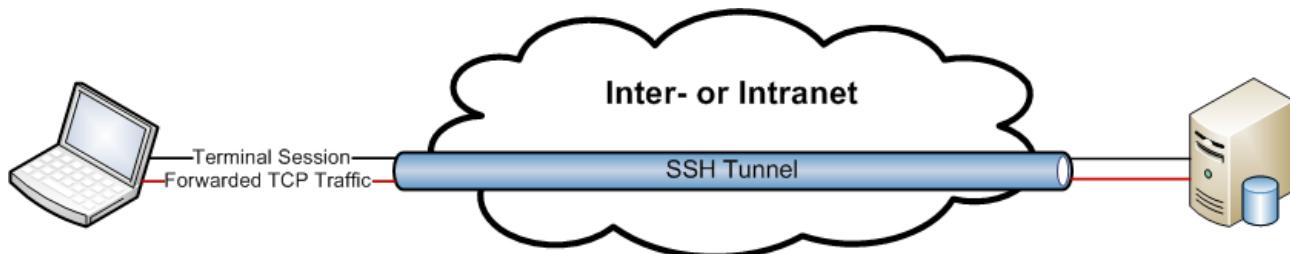
La prima versione di SSH era completamente open source, mentre la seconda è diventata commerciale; esiste comunque una **versione libera** detta **OpenSSH** che si basa sulla prima versione, ma che fornisce supporto anche alla seconda versione.

⁷⁹ L. Lo Russo, E. Bianchi, *Sistemi e Reti - Nuova Edizione OPEN SCHOOL - Per l'articolazione INFORMATICA degli Istituti Tecnici settore Tecnologico*, vol. 3, ed. Hoepli, 2017, UdA 1 - Il livello delle applicazioni, Lezione 3 - Email, Dns e Telnet pp.28-31.

Sicurezza di SSH

Scambio di dati sicuro

Utilizzando SSH lo **scambio dei dati** tra client e server avviene in modo **cifrato**, utilizzando un **meccanismo a chiave simmetrica**.



Per effettuare lo **scambio della chiave di sessione**, valida per una sola sessione SSH e utilizzata per crittografare i dati scambiati, vengono usate delle variazioni del metodo [**Diffie-Hellman**](#) nelle quali viene aggiunto un sistema di certificazione del server mediante l'uso di chiavi RSA usate per auto-certificarsi. Per garantire maggiore sicurezza le chiavi vengono generalmente rinegoziate ogni ora, o dopo il transito nella connessione di un certo quantitativo in GigaByte di dati.

Autenticazione

Autenticazione del Server

L'autenticazione del server ha lo scopo di evitare che un attaccante impersoni il server, facendosi fornire le credenziali dell'utente (spoofing da attacco man in the middle). A questo scopo il server genera una coppia di chiavi asimmetriche. La chiave privata rimane sul server, mentre la chiave pubblica deve essere conosciuta dal client. Il client può ottenere la chiave di un server utilizzando archivi pubblici delle chiavi disponibili sul Web, o ricevendola direttamente dal server durante la prima connessione.

L'autenticazione del server avviene durante lo scambio delle chiavi con il metodo Diffie-Hellman, e prevede che un **client** che si colleghi ad un server, di cui conosce la chiave pubblica, ne **verifichi l'identità accertandosi che sia in possesso della chiave privata**. Se questa verifica fallisce la connessione viene abbattuta, evitando di fornire credenziali a potenziali attaccanti.

Affinché un client possa verificare che il server sia in possesso della chiave privata, e quindi che non sia un attaccante, il server dovrà generare un messaggio contenente la sua chiave pubblica RSA (**RSAPU_s**) e la sua chiave pubblica Diffie-Hellman (**DHPU_s**). Quest'ultima chiave dovrà essere firmata usando la chiave privata RSA del server (**RSAPR_s**), allegando la firma (**E (RSAPR_s, DHPU_s)**) al messaggio.

Il client verificherà la firma decriptando il messaggio con la chiave pubblica RSA del server (**D (RSAPU_s, E (RSAPR_s, DHPU_s)**) e presente nello stesso messaggio.

Il meccanismo in realtà non risulta sicuro nei confronti di un attacco man-in-the-middle dato che la chiave **RSAPU_s** usata per verificare la firma del server è interna al messaggio stesso, in pratica il server si autocertifica non utilizzando una *Certification Authority*.

Nella pratica, quando un utente si collega per la prima volta ad un server, il client SSH chiederà all'utente di confermare l'accettazione della chiave pubblica del server. Se l'utente risponderà positivamente il client memorizzerà la chiave e proseguirà con la connessione. Per le successive connessioni con lo stesso server il client si limiterà a verificare l'autenticazione usando la chiave pubblica salvata e, nel caso la verifica non andasse a buon fine, impedirà il proseguimento della connessione. In ogni caso l'accettazione iniziale della chiave pubblica del server è completamente non sicura, a meno che non sia stato possibile verificare la correttezza della **RSAPU_s** in una fase precedente.

Autenticazione del client

Dopo aver creato un canale di comunicazione sicuro, il protocollo SSH prevede l'autenticazione del client. Esistono principalmente **due metodi di autenticazione** per controllare l'accesso di un **client** ad un server SSH, tramite **username e password**, o usando un **metodo a chiave asimmetrica**.

L'utilizzo di uno **username** e una **password** richiede che l'utente fornisca questi dati che dovranno essere validati dal server. Questo scambio avviene all'interno di un canale cifrato, che permette di evitare eventuali intercettazioni.

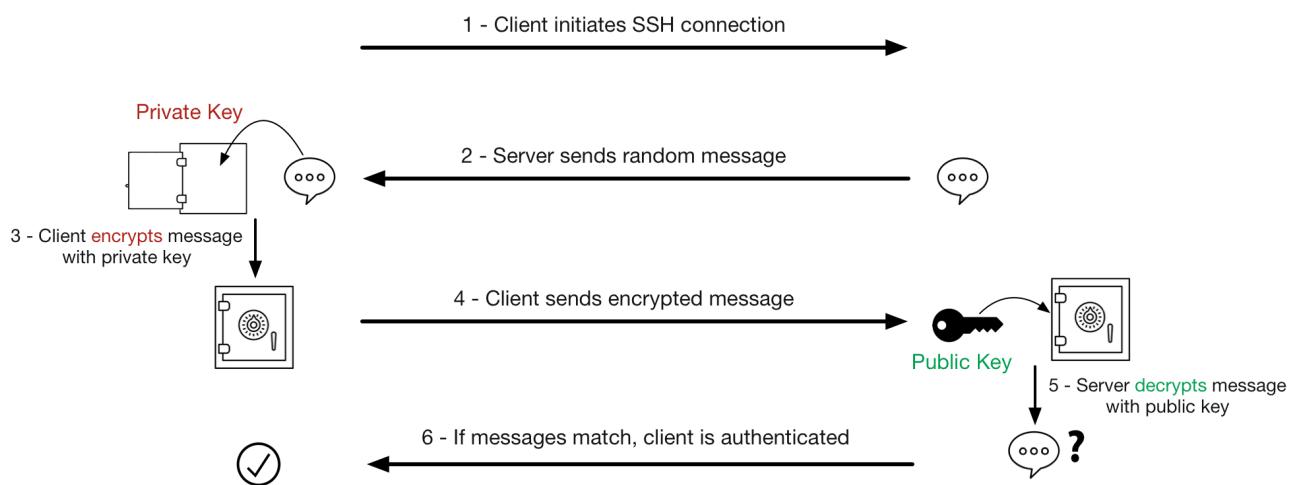
Il **metodo a chiave asimmetrica** prevede invece che l'utente generi una coppia di chiavi, una pubblica e una privata, usando ad esempio RSA. La chiave pubblica deve essere copiata sul server, trasferendola mediante supporti di memorizzazione fisici o utilizzando altri metodi, mentre la chiave privata deve essere conservata dall'utente, eventualmente protetta con una parola chiave (*passphrase*) per una maggior sicurezza.

Il server sorgente per verificare l'identità dell'utente genera una stringa casuale di 256 bit (**M**), la critta usando la chiave pubblica dell'utente (**E (PU_c, M)**) e la invia al client.

Il client decifra il messaggio utilizzando la propria chiave privata (**D (PR_c, E (PU_c, M))**) e invia lo hash della stringa ricevuta al server (**h (M)**). Se lo hash della stringa del client corrisponde allo hash della stringa calcolata dal server l'utente viene autenticato.

Nell'autenticazione mediante chiavi pubbliche non viene richiesta nessuna password all'utente tranne nel caso in cui sia stata applicata una *passphrase* alla chiave privata.

Anche in questo caso il protocollo SSH non offre alcun metodo di certificazione della chiave pubblica del client tramita *Certification Authority*, risultando quindi vulnerabile ad un attacco man-in-the-middle.



23 - Esercizi su netacad.com

Svolgere i seguenti esercizi che trovate sul corso CCNA1 R&S nel sito [netacad.com](#)

Capitolo 11 - Build a Small Network

- 11.2.4.5 (PT) - Configurazione di una password sicura e di SSH.
- 11.2.4.6 (Lab) - Accesso ad un dispositivo di rete usando SSH; usare Packet Tracer per emulare la situazione reale.
- 11.2.4.7 (Lab) - Esame con Wireshark dell'interazione Telnet ed SSH.

Kerberos

Kerberos⁸⁰ è un protocollo basato su un centro di distribuzione delle chiavi (**KDC** - Key Distribution Center) che **garantisce l'AAA**, ed è molto diffuso per gli accessi di tipo logon a sistemi operativi.

Kerberos è il sistema di autenticazione standard delle utenze e dei processi utilizzato da Windows di classe Server, e quindi Linux Samba, a partire dalle versioni Windows 2000⁸¹. È un applicativo di livello 7 che usa UDP come trasporto sulla porta 88, anche se nelle ultime versioni può utilizzare anche TCP.

Lo schema di **autenticazione Kerberos**, **basato su crittografia simmetrica**, si regge su un protocollo di tipo *sfida/risposta* dove viene utilizzato un *nonce*⁸² basato sul tempo rilevato quando l'utente immette la password. Questo *nonce* temporale inviato al destinatario deve essere ricostruito in modo preciso e reinviato tramite la risposta affinché si possa garantire la non interferenza nella comunicazione da parte di terzi malintenzionati. Il fatto che venga usato un *nonce* temporale implica che gli orologi dei sistemi siano sincronizzati tramite un *time server*.

Nel **protocollo Kerberos**, così come in ogni protocollo di tipo **KDC** (Key Distribution Center), l'obiettivo è quello di ottenere una chiave di sessione condivisa e sicura con il sistema su cui autenticarsi, da utilizzare poi con crittografia simmetrica per le successive interazioni.

Il **protocollo Kerberos** prevede sostanzialmente un sistema a *tre teste*:

- **l'Authentication System (AS)** che corrisponde al **KDC** (Key Distribution Center) che mantiene il database delle chiavi segrete dei partecipanti alla comunicazione e fornisce la **chiave di sessione** da usare nella comunicazione criptata;
- **Ticket Granting Service (TGS)** che fornisce su richiesta del client un **ticket** con il quale potrà presentarsi al server che erogherà il servizio. Il **ticket** è formato da un insieme di dati crittografati che identificano il client con una validità temporale limitata riportata nel **ticket** stesso;
- **Service Server (SS)** che rappresenta il server erogatore del servizio richiesto dal client.

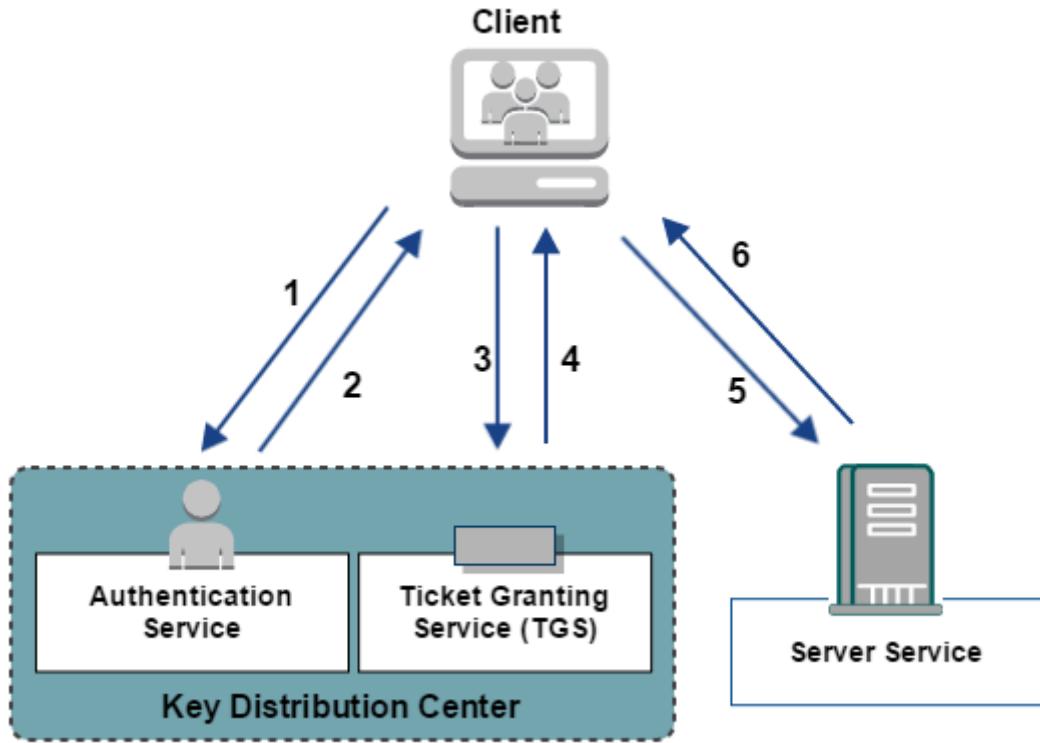
L'**autenticazione tramite Kerberos** avviene in **tre fasi** (da qui il nome Kerberos, dal mitico cane Cerbero a tre teste):

⁸⁰ RFC 1510 e 4120 versione 5

⁸¹ RFC 3244, «Microsoft Windows 2000 Kerberos Change Password and Set Password Protocols»

⁸² In crittografia il termine *nonce* indica un numero, generalmente casuale o pseudo-casuale, che ha un utilizzo unico. *Nonce* deriva infatti dall'espressione inglese *for the nonce*, che significa appunto "per l'occasione". Un *nonce* viene utilizzato spesso nei protocolli di autenticazione per assicurare che i dati scambiati nelle vecchie comunicazioni non possano essere riutilizzati in attacchi di tipo *replay attack*. Con il termine *nonce* ci si riferisce spesso anche ai vettori di inizializzazione utilizzati negli algoritmi di inizializzazione di molti algoritmi crittografici per inizializzare il cifrario stesso; per essere sicuri che il *nonce* sia veramente utilizzato solo una volta, dovrebbe variare nel tempo, includendo, ad esempio, la data e l'orario nel suo valore, o generato con un numero tale di bit casuali da rendere probabilisticamente insignificante la possibilità che uno stesso valore sia ripetuto più di una volta.

- a. il **client** richiedente si accredita presso l'**AS**, il quale gli dà gli strumenti per accreditarsi presso il **TGS**;
- b. il **TGS** fornisce al client un **ticket** (o token) utile per accreditarsi presso il **Service Server**;
- c. il **client** si accredita presso il **Service Server** per ottenere l'accesso ai servizi richiesti.



A differenza di altri sistemi basati su *sfida/risposta* (per esempio CHAP e WPA2), **Kerberos non fa mai circolare sulla rete né la password dell'host richiedente, né il suo hash**: quando l'host richiedente dovrà digitare la sua password, essa servirà solo localmente per decifrare un particolare messaggio giunto dall'**AS**, dopodiché verrà immediatamente eliminata.

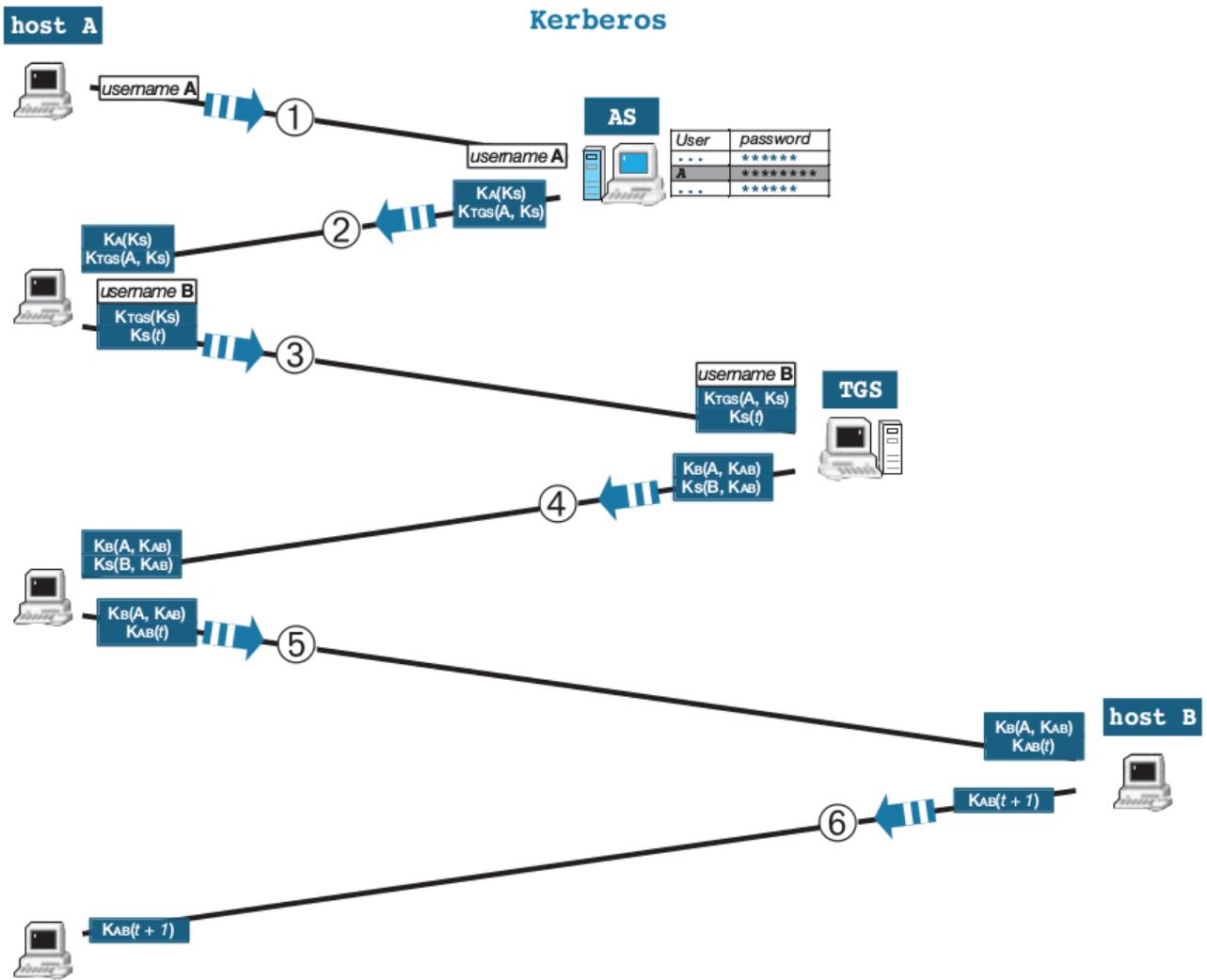
La procedura di autenticazione si basa sui seguenti passi:

1. Il **client A** vuole autenticarsi presso il **server B** per poterne usare i servizi, quindi il **client A invia in chiaro** allo **AS** (*Authentication System*) il proprio **username A**.
2. **AS**, che possiede l'archivio dei segreti:
 - a. individua la **chiave segreta di A** (K_A);
 - b. sceglie una **chiave di sessione intermedia** (K_s) e la cifra con la chiave segreta di **A** ($E(K_A, K_s)$);
 - c. l'**username** di **A** e la **chiave intermedia** K_s vengono cifrate con la chiave segreta che **AS** ha in comune con **TGS** ($E(K_{TGS}, (A || K_s))$);
 - d. invia $E(K_A, K_s)$ e $E(K_{TGS}, (A || K_s))$ al **client A**.
3. Solo ora l'utente **A inserisce la password** K_A per poter **decriptare** K_s , password usata solo localmente e che viene eliminata subito dopo.

Nel momento in cui viene inserita la password la macchina di **A memorizza** anche **il tempo t**. Il secondo valore ricevuto ($\mathbf{E}(\mathbf{K}_{\text{TGS}}, (\mathbf{A} \parallel \mathbf{K}_s))$), non è invece decodificabile da **A** perché non possiede la chiave \mathbf{K}_{TGS} . Questo secondo valore viene **rispedito al server TGS insieme al nonce t** che invece viene cifrato con \mathbf{K}_s e, insieme a questi viene inviato in chiaro anche lo **username** del sistema presso cui A vuole autenticarsi, ovvero **B** ($\mathbf{B} \parallel \mathbf{E}(\mathbf{K}_{\text{TGS}}, (\mathbf{A} \parallel \mathbf{K}_s)) \parallel \mathbf{E}(\mathbf{K}_s, t)$).

4. Il **TGS** decifra \mathbf{K}_s tramite la sua \mathbf{K}_{TGS} , e recupera quindi in *nonce t* usando \mathbf{K}_s . Tramite l'username **B** il **TGS** ne **recupera la chiave segreta \mathbf{K}_B** e **stabilisce** l'effettiva **chiave di sessione segreta** che sarà condivisa da **A** e **B**, cioè \mathbf{K}_{AB} . A questo punto il **TGS** cifra l'identificativo di **A** e la chiave \mathbf{K}_{AB} usando la chiave segreta di **B** ($\mathbf{E}(\mathbf{K}_B, (\mathbf{A} \parallel \mathbf{K}_{AB}))$), mentre critta l'identificativo di **B** e la chiave \mathbf{K}_{AB} con la chiave di sessione intermedia \mathbf{K}_s ($\mathbf{E}(\mathbf{K}_s, (\mathbf{B} \parallel \mathbf{K}_{AB}))$), quindi spedisce entrambi i due messaggi ad **A**.
5. Il client **A** ottiene la chiave di sessione effettiva \mathbf{K}_{AB} , usando la chiave \mathbf{K}_s . Il valore $\mathbf{E}(\mathbf{K}_B, (\mathbf{A} \parallel \mathbf{K}_{AB}))$ non è invece decodificabile perché **A** non possiede la chiave \mathbf{K}_B ; questo valore deve essere spedito così com'è al server **B**, insieme ad un secondo *nonce temporale t'* cifrato con la chiave di sessione \mathbf{K}_{AB} .
6. Il server **B** otterrà quindi la chiave di sessione \mathbf{K}_{AB} , decifrando il messaggio con la sua chiave \mathbf{K}_B . Come ultimo atto, il server **B si autenticherà con il client A inviandogli una risposta contenente in nonce t' + 1** cifrato con la chiave di sessione \mathbf{K}_{AB} .

Nell'immagine seguente le notazioni usate per criptare i dati sono diverse, ad esempio $\mathbf{E}(\mathbf{K}_A, \mathbf{K}_s) \rightarrow \mathbf{K}_A(\mathbf{K}_s)$ oppure $\mathbf{E}(\mathbf{K}_{\text{TGS}}, (\mathbf{A} \parallel \mathbf{K}_s)) \rightarrow \mathbf{K}_{\text{TGS}}(\mathbf{A}, \mathbf{K}_s)$.



Si noti che il primo *nonce temporale t* ricevuto da **TGS** verrà confrontato con il suo tempo attuale per verificare che i due tempi risultino sufficientemente *prossimi*, in modo da garantire la *freschezza* del messaggio, ed evitare attacchi di tipo *replay*.

Il secondo *nonce temporale t'* è invece una classica sfida, infatti **B** deve dimostrare di possedere la chiave di sessione **K_{AB}** per decifrare questo messaggio e cifrarlo con il *nonce temporale t' + 1*. Quando **A** lo decifrerà, il confronto tra il **t'** che aveva creato e il **t' + 1** ricevuto nel messaggio permetterà di capire se **B** è effettivamente chi dice di essere.

Il protocollo Kerberos è abbastanza articolato e, apparentemente, poco efficiente, però presenta anche diversi vantaggi:

- i client non condividono un segreto con il **KDC**, come avviene in altri protocolli di tipo **KDC**. Le entità che condividono un segreto sono lo **AS** e il **TGS**, ed è la chiave **K_{TGS}**. Quindi si ha un ottimo risparmio in termini di prestazioni e sicurezza: si elimina un segreto ripetuto molte volte (per tutti gli utenti) in cambio di un solo segreto tra due macchine veloci.
- Lo schema a sei messaggi non deve essere sempre svolto completamente. I primi due messaggi non devono più essere inviati da **A** nel caso volesse

autenticarsi presso un altro server, sarà sufficiente partire dal terzo messaggio. In pratica il client **A** è l'entità che richiede l'autenticazione più onerosa in quanto lo **AS** deve accedere a un database, ma questa autenticazione viene effettuata una sola volta durante la vita del processo client. Questa seconda caratteristica diventa particolarmente utile quando il server **B** non è un utente ma un processo di sistema operativo e le richieste del client **A** risultino molto frequenti e rivolte a numerosi servizi del sistema operativo, come le richieste di accedere al file system di rete o a risorse distribuite. Con questo metodo si implementa un **single sign-on**.

Nella **versione 5 di Kerberos** il protocollo può essere impostato per utilizzare praticamente ogni coppia di algoritmo simmetrico e modo di sintesi desiderati, come AES/SHA1, 3DES/SHA1, 3DES/MD5 ecc., compresi anche schemi a cifratura asimmetrica come RSA.

RADIUS [*rivedere, non mi piace*]

In⁸³ molti casi gli apparati o i processi NAS che devono autenticare un utente si ritrovano a dover conservare un grande archivio di account segreti per verificare la risposta di una sfida.

Per esempio i NAS degli Internet Service Provider (ISP) che devono autenticare i clienti sulla propria rete (per esempio con PPP e CHAP), oppure gli Access Point Wi-Fi che non vogliono una gestione generica della sicurezza tramite PSK condivisa, oppure ancora l'accesso da un host (logon) a una rete con molti utenti.

In tutti questi casi è necessario delegare l'autenticazione a un server fidato che opera come centro di distribuzione delle chiavi (KDC).

Il **protocollo RADIUS** (Remote Authentication Dial In User Service, RFC 2865) è uno dei protocolli più diffusi per l'**autenticazione basato su account con un centro di distribuzione delle chiavi**: in quanto tale **RADIUS garantisce AAA**.

In realtà RADIUS è un protocollo client/server di livello 7 Applicazione trasportato da UDP (sulle porte 1812 e 1823), ma in molti casi è utilizzato per aprire un accesso a livello 2 Collegamento Dati (per esempio PPP/CHAP per gli ISP o WPA2 enterprise per Wi-Fi).

La letteratura circa Il protocollo RADIUS usa vari nomi e acronimi per identificare i tre soggetti dello schema che implementa, ovvero l'**host**, il **client RADIUS** e il **server RADIUS**.

Usato **con Wi-Fi**, la **stazione che richiede l'autenticazione è detta anche supplicant**, mentre l'**Access Point è detto Authenticator** (sarebbe il client RADIUS). Il **server RADIUS** invece **prende il nome di AS** (Authentication Server).

⁸³ Da un estratto del libro di P. Ollari, *Corso di sistemi e reti per Informatica - Applicazioni e sicurezza in rete*, vol. 3, ed. Zanichelli, 2013

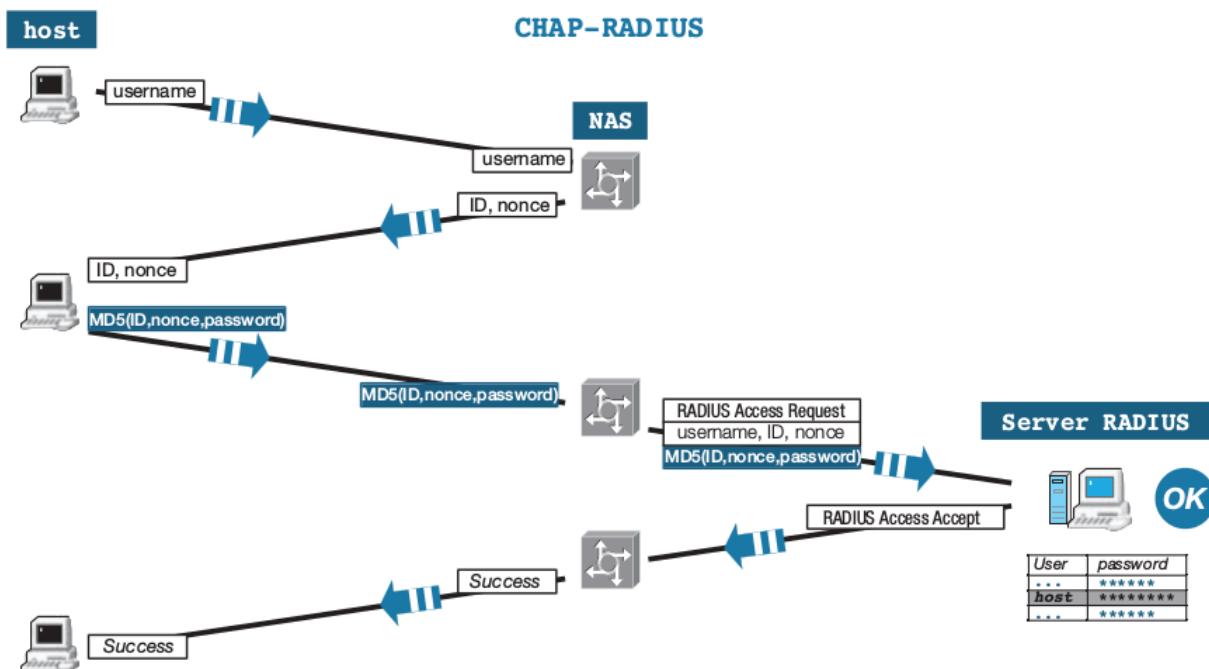
In altri contesti il client RADIUS è chiamato RAS (Remote Access Server) o direttamente NAS (Network Access Server).

Nello schema **RADIUS** quindi agiscono **tre entità**:

1. l'**host richiedente** (colui che richiede l'autenticazione);
2. il **client RADIUS** (client rispetto al server RADIUS);
3. il **server RADIUS** (che gestisce l'archivio dei segreti degli account).

Il protocollo RADIUS può essere impostato per agire con vari schemi di autenticazione usando EAP, comunque basati sul modello *sfida/risposta*.

Nel caso **PPP/CHAP**, per esempio, è il server CHAP che implementa il client RADIUS: quando riceve la risposta della sfida dall'host richiedente, invita il server RADIUS a eseguire l'autenticazione inviandogli utente/password, la sfida e la risposta ricevuta tramite il protocollo RADIUS. In altri casi delega completamente il server RADIUS a implementare la sfida/risposta. In ogni caso il client RADIUS, sull'esito della risposta del server RADIUS, emetterà il pacchetto *Success* (o *Failure*) all'host richiedente.



Nel caso **WPA2 enterprise** è l'**Access Point AP** a svolgere il ruolo di **client RADIUS**: delega un server RADIUS affinché autentichi direttamente la stazione (per esempio tramite *username/password* dell'utente della stazione) e stabilisca la chiave segreta con essa (la PMK).

In questi casi AP e Server RADIUS sono connessi in modo cablato (per esempio, sulla porta Ethernet dell'AP), pertanto tutti i pacchetti di scambio tra la stazione e il server RADIUS passano attraverso l'AP che si limita a trasportarli nei due versi, bloccando contemporaneamente qualsiasi altra comunicazione wireless fino al termine dell'autenticazione.

La chiave segreta rilasciata dal server RADIUS (sia alla stazione sia all'AP) sarà la PMK – a questo punto equivalente a una PSK, che consentirà alla stazione e all'AP di avviare il four way handshake per individuare le chiavi di sessione per la crittografia.

In entrambi gli schemi esaminati si è tralasciato un dettaglio non insignificante: **client RADIUS e server RADIUS devono condividere un segreto** (una **chiave simmetrica**) **per autenticarsi vicendevolmente prima di avviare i loro scambi**.

Infine bisogna dire che l'archivio dei segreti degli account non necessariamente risiede sempre sul server RADIUS: il protocollo prevede che il server RADIUS possa ricavare gli account richiesti da altri archivi come LDAP, Active Directory, o tramite altri protocolli di sicurezza come Kerberos.

RADIUS è un protocollo molto utilizzato perché è reso disponibile da molti costruttori sui propri NAS (Network Access Server). Per esempio tutti gli Access Point anche di fascia bassa implementano client RADIUS e, a volte, anche server RADIUS.

In alternativa a RADIUS viene utilizzato un protocollo analogo ma più recente ed efficiente denominato Diameter (RFC 4004), retrocompatibile con RADIUS e basato su TCP.

G - Sicurezza nelle WLAN

Prefazione

I seguenti appunti sono basati sul lavoro svolto dalla Prof.ssa Sophia Danesino, mentore, amica, appassionata di conoscenza e insegnante fuori dal comune. Sul suo [canale YouTube](#) è possibile trovare i video su molti degli argomenti trattati.

Libro vol.3 - Wireless e reti mobili

STUDIARE: L. Lo Russo, E. Bianchi, Sistemi e Reti - Nuova Edizione OPENSCHOOL - Per l'articolazione INFORMATICA degli Istituti Tecnici settore Tecnologico, vol. 3, ed. Hoepli, 2017

- Wireless: comunicare senza fili pp.241-249
- L'autenticazione nelle reti wireless pp.250-259
- La trasmissione wireless pp.260-270
- L'architettura delle reti wireless pp.271-280

Collegamento ad una WLAN

Il **Service Set Identifier (SSID)** è il nome con cui una rete Wi-Fi, o in generale una WLAN (*Basic Service Set - BSS*), si identifica ai suoi utenti.

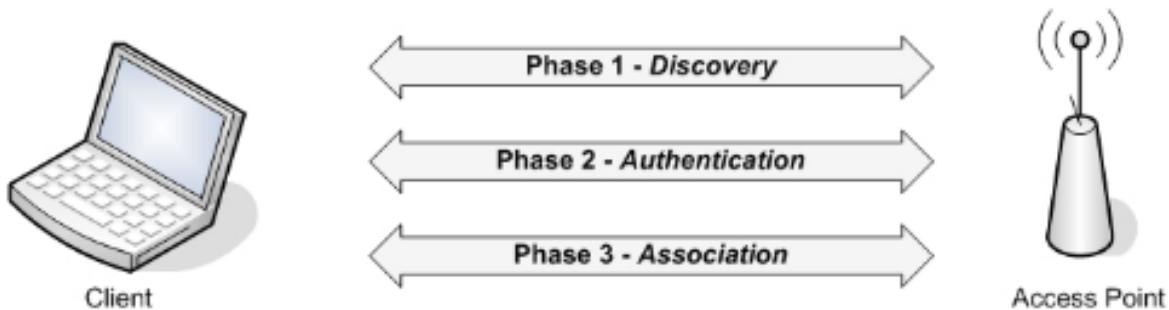
La configurazione dello **SSID** è generalmente assegnata dall'amministratore di rete e consiste in genere in una serie di caratteri ASCII stampabili; lo standard comunque non specifica nulla in proposito.

Lo **SSID** è quindi un nome che viene sovente mostrato agli utenti, e viene trasmesso periodicamente in frame detti **beacon**⁸⁴, se però non si vuole annunciare la rete, lo **SSID** può essere nascosto rendendolo non visibile.

Tra **Access Point** e **SSID** non c'è una stretta relazione, infatti è normale che più **Access Point** possano condividere lo stesso **SSID** se forniscono accesso alla stessa rete, così come è anche possibile che uno stesso **Access Point** si presenti con più **SSID** se fornisce accesso a diverse reti.

Il collegamento ad una rete wireless si compone di tre fasi, come illustrato anche in figura:

- scanning o discovery della rete;
- autenticazione;
- associazione.



⁸⁴ Beacon: sorgente di luce (radiofaro).

Scanning o discovery della rete

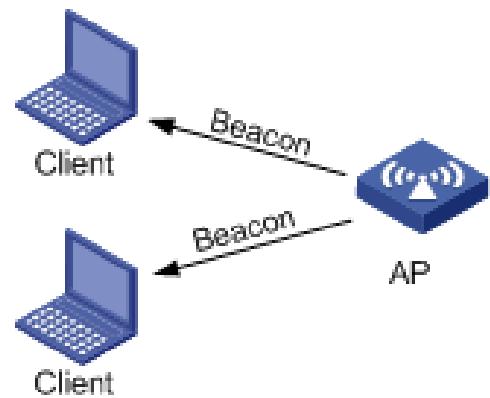
Quando un host wireless si accende, deve determinare se è presente un altro host wireless o un Access Point per fare il **join alla rete**. Lo host wireless realizza questa fase di scoperta mediante una **scansione passiva o attiva**.

Dopo che lo host wireless ha effettuato il join con un **BSS** (*Basic Service Set*) o un **ESS** (*Extended Service Set*), scetterà lo **SSID** (*Service Set Identifier*), **TSF** (*Timer Synchronization Function*) e i parametri fisici stabiliti dallo Access Point.

Passive Scanning mode

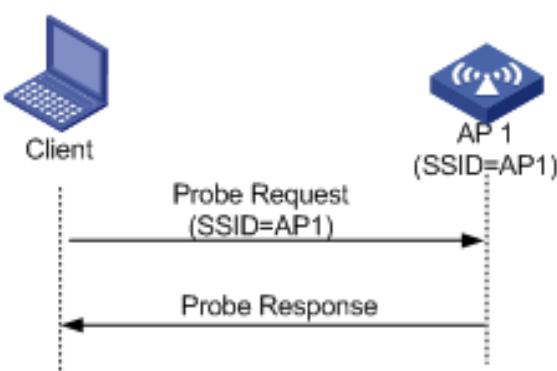
La **scansione passiva** (*Passive Scanning mode*) prevede che lo host wireless ascolti ogni canale per uno specifico periodo di tempo, nell'attesa che venga trasmesso il frame beacon contenente lo SSID necessario per fare il join.

In questa modalità lo Access Point invia periodicamente il Beacon Frame che contiene le informazioni che servono ai client per collegarsi (SSID, Access Point address, intervallo di invio dei Beacon Frame, velocità di trasferimento disponibili, ecc.).



Se si volesse **nascondere la rete** si dovrà configurare lo **Access Point** in modo che **non annunci lo SSID**, disabilitandone il broadcast. In questo modo il nome della rete non apparirà negli elenchi delle reti disponibili dei dispositivi Wi-Fi, ma nonostante questa modalità **la rete risulterà comunque individuabile**, quindi **il mancato annuncio dello SSID di una WLAN non può essere considerata una misura di sicurezza**.

Active Scanning Mode

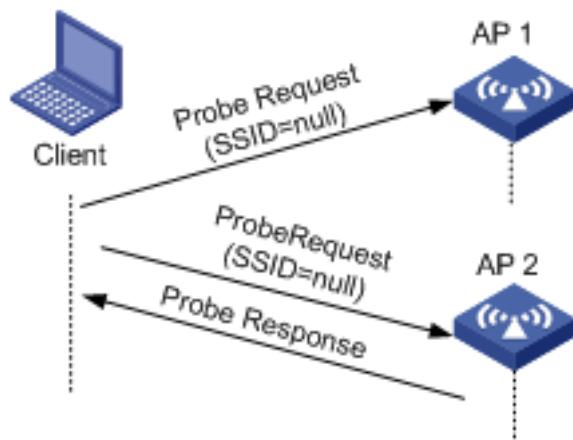


La **scansione attiva** (*Active Scanning Mode*) prevede che **lo host wireless invii** un frame di richiesta, denominato **Probe⁸⁵ Request**, contenente lo SSID della rete con la quale vuole effettuare il join; ciò implica che lo host wireless sia già a conoscenza dello SSID della rete WLAN a cui vuole connettersi.

Successivamente lo host wireless si porrà in attesa di ricevere la risposta (**Probe Response**) che gli annuncia la presenza della rete desiderata.

⁸⁵ To probe: sondare, indagare.

Lo host wireless può effettuare la **scansione attiva inviando in broadcast lo SSID** della WLAN a cui vuole connettersi. Questa modalità indurrà tutte le reti appartenenti a un'area a rispondere. Il frame di risposta indicherà la presenza della rete desiderata, e lo host wireless potrà completare la connessione effettuando l'autenticazione e il processo di associazione.



Autenticazione

L'autenticazione 802.11 richiede che un host wireless presenti la propria identità ad un punto di accesso (*Access Point*) o ad un router wireless.

Lo standard **IEEE 802.11** definisce **due tipi di collegamento** a livello di **autenticazione**:

- **sistema aperto**;
- **sistema a chiave condivisa**.

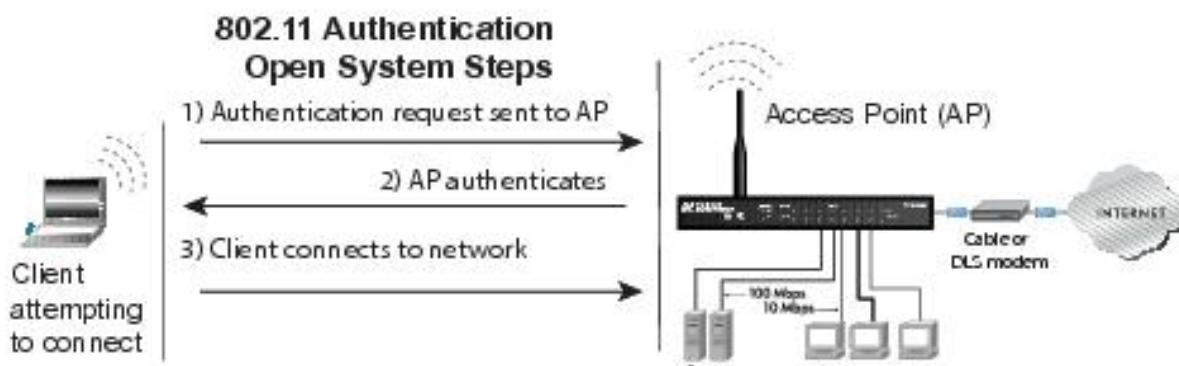
Esiste poi un terzo protocollo, **802.1X**, che **utilizza un server esterno per l'autenticazione**.

Autenticazione a sistema aperto

Lo host wireless invia una richiesta allo Access Point che lo autenticherà, permettendogli di effettuare l'associazione.

L'autenticazione a sistema aperto è quindi **un sistema costituito da due comunicazioni**:

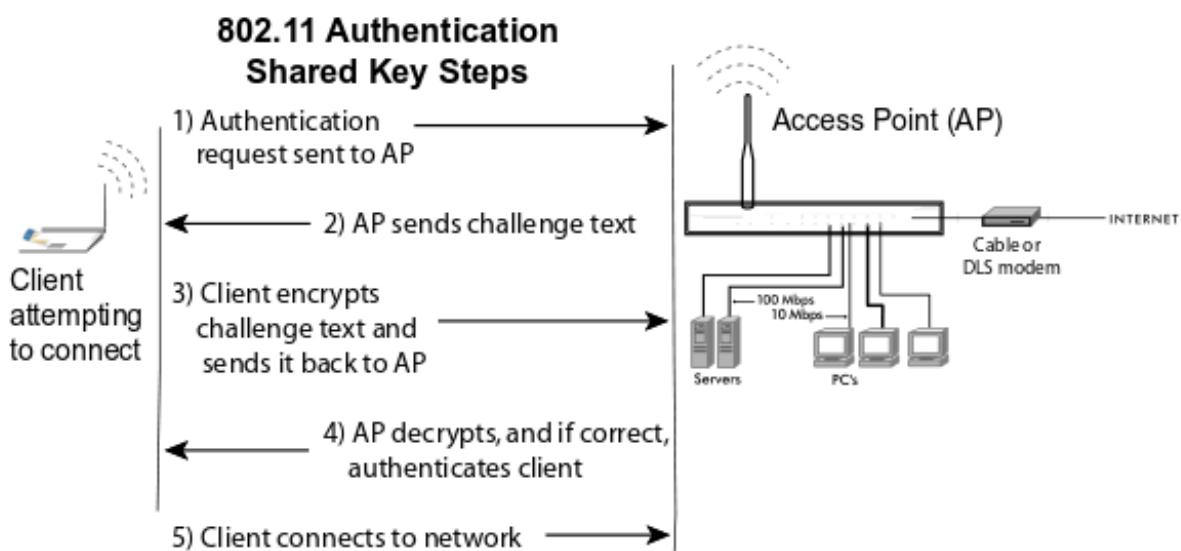
- una **richiesta di autenticazione** dal dispositivo mobile che contiene l'ID dello host wireless (in genere l'indirizzo MAC);
- una **risposta di autenticazione** dal router o dallo Access Point contenente un messaggio di esito positivo o negativo.



Autenticazione a chiave condivisa

Con l'**autenticazione a chiave condivisa (PSK - Pre-Shared Key)** una **chiave** o una **passphrase** viene **impostata manualmente** sul dispositivo mobile e sul router o Access Point.

Il client invierà una richiesta di autenticazione allo Access Point, il quale risponderà inviando un numero casuale, detto *challenge text*. Il client cifrerà questo testo con la chiave condivisa e lo invierà allo Access Point. Lo Access Point verificherà il testo cifrato e, se corrisponderà alla sua chiave, invierà una conferma di autenticazione del client.



Esistono diversi tipi di autenticazione a chiave condivisa:

- **WEP (Wired Equivalent Privacy)**: si veda il libro a p.217 per una spiegazione;
- **WPA** e la sua evoluzione **WPA2 (Wi-Fi Protected Access)**: si veda il libro a p.217 per una spiegazione.

Nonostante la tecnica **WPA2** risulti essere la più sicura fra quelle presentate, questo secondo meccanismo non è comunque sicuro poiché:

- è facilmente perpetrabile un attacco di forza bruta per individuare la chiave segreta condivisa;
- lo Access Point autentica il client ma il client non autentica lo Access Point, risultando quindi vulnerabile ad un attaccante che si mascheri da Access Point e che ridireziona il traffico destinando allo Access Point originale;
- l'autenticazione avviene sul device e non sull'utente.

Autenticazione 802.1X

I **componenti del sistema di autenticazione 802.1X** sono:

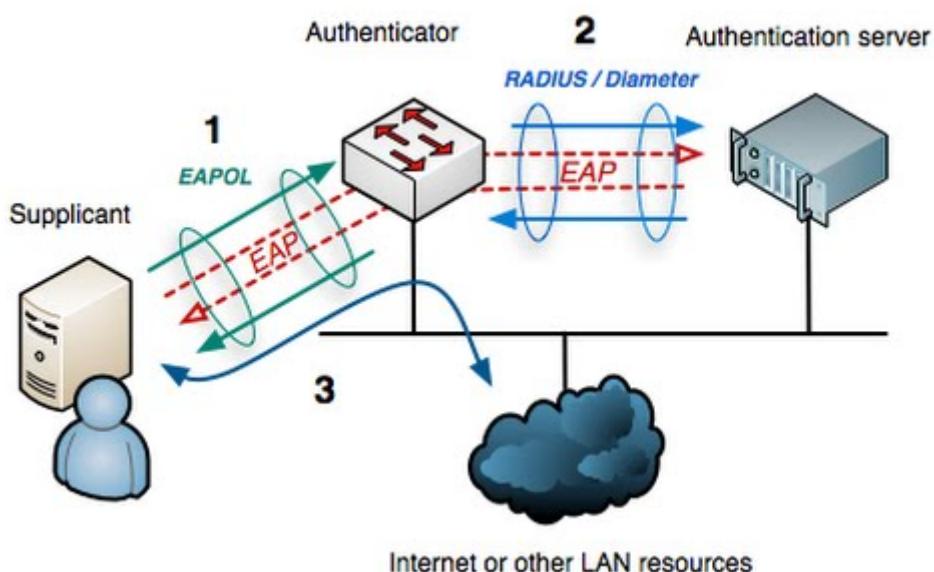
- un **client**, denominato **Supplicant**, che cerca di associarsi ad una rete che utilizza il metodo di autenticazione 802.1x;
- lo **Authenticator** che identifica lo/gli **Access Point**;

- lo **Authentication server** che identifica il server a cui è delegato il compito di stabilire se l'utente che vuole accedere alla rete è veramente chi dice di essere.

Molto spesso lo **Authentication server** coincide con un **server RADIUS** (*Remote Authentication Dial-In User Service*) che **implementa un protocollo AAA** (*Authentication⁸⁶, Authorization⁸⁷, Accounting⁸⁸*) utilizzato in applicazioni di accesso alle reti o di mobilità IP.

RADIUS è un protocollo che **utilizza** a livello di Trasporto dei datagram **UDP** per trasportare informazioni di autenticazione e di configurazione tra lo Authenticator e il server RADIUS. L'autenticazione è **basata su username, password e, opzionalmente, una risposta a una richiesta di riconoscimento** (una sorta di "parola d'ordine").

Lo standard 802.1X si basa su un protocollo IETF denominato **EAP** (*Extensible Authentication Protocol*) per gestire lo scambio di dati di autenticazione tra il client e il server RADIUS (inoltrati dal punto di accesso). **EAP** è un protocollo generico per l'autenticazione che supporta numerosi metodi di autenticazione: con password, usando certificati digitali o altri tipi di credenziali. Proprio per questo è detto *Extensible*.



⁸⁶ **Authentication:** rappresenta il processo con cui si conferma che un utente che richiede un servizio sia un utente valido e conosciuto. Viene realizzata con la presentazione di una identità o di credenziali, come ad esempio una password, l'indirizzo MAC della scheda wireless di un portatile o un certificato digitale.

⁸⁷ **Authorization:** rappresenta il processo con cui si stabilisce a quali servizi può accedere l'utente che si è autenticato sul sistema. Questo processo può basarsi su politiche che tengano in considerazione l'orario di richiesta del servizio, la collocazione fisica del richiedente o altre informazioni.

⁸⁸ **Accounting:** rappresenta il processo con cui è possibile tracciare il consumo delle risorse di rete di un utente. Queste informazioni possono essere utilizzate per la gestione futura, per motivi statistici, o magari per redigere un conto mensile, come viene fatto negli ISP. Vengono solitamente memorizzate informazioni sull'identità dell'utente, sulla natura del servizio fornito e sull'orario di inizio e di fine della fruizione.

Associazione

Una volta completata l'autenticazione, i dispositivi mobili si possono **associare** (*register*) con un router o punto di accesso per ottenere l'accesso completo alla rete. L'associazione consente al punto di accesso/router di registrare ciascun dispositivo mobile in modo che i frame possano essere recapitati correttamente. L'associazione si verifica solo su reti con infrastruttura, non in modalità ad hoc (peer-peer) e ogni host wireless può essere associata a un solo router o Access Point alla volta.

Esercizi

Nella play list [CCNA Security](#) sono presenti alcuni video che permetteranno di provare le configurazioni di diversi apparati per la sicurezza in rete. Si consiglia di vederli provando a svolgerli in contemporanea alla visione.

H - La comunicazione non cablata

Protocolli per la comunicazione non cablata

Streaming

[Streaming media](#) - [Wikipedia](#)

[Content delivery network](#) - [Wikipedia](#)

[Web application firewall](#) - [Wikipedia](#)

[Which Video Streaming Protocol Should You Use?](#), 2017 (*seguire i link nel testo e leggerli*) - [Dacast](#)

[Here's What You Need to Know About Streaming Protocols](#), 2018 (*seguire i link nel testo e leggerli*) - [Dacast](#)

[What Is MPEG DASH And Why Should We Use It?](#), 2016 (*seguire i link nel testo e leggerli*) - [Dacast](#)

[What is Adaptive Bitrate Streaming?](#), 2014 - [Encoding.com](#)

[Choosing a Live Video Streaming Solution for Broadcasting Live Audio](#), 2018 - [Dacast](#)

[How to Easily Get Access to A Powerful Live Streaming CDN](#), 2018 - [Dacast](#)

[All you need to know about video codecs containers and compression](#) - [MakeUseOf](#)

Sensori

[Reti di sensori wireless: sfide e soluzioni](#) e in inglese [qui](#) con le immagini visibili

[Rete di sensori wireless \(WSN\)](#)

[Wireless Sensor Networks: Security Issues, Challenges and Solutions](#)

[Issues, Challenges and Solution for Security in Wireless Sensor Networks: A Review](#)

[Secure Wireless Sensor Networks: Problems and Solutions](#)

[Object Tracking in Wireless Sensor Networks: Challenges and Solutions](#)

[Security in Wireless Sensor Networks: Issues and Challenges](#)

[CSP](#) - Sito del CSP di Torino illustra molti esempi di utilizzo di sensori in molteplici ambiti. Se ne suggerisce la lettura per contestualizzare in un ambito di realtà le possibili applicazioni di quanto appreso durante l'intero corso di studi.

[RFID](#), versione in inglese: [Radio-frequency identification](#)

Standard IEEE 802.15.4, ZigBee e Bluetooth

[Standard IEEE 802.15.4 e ZigBee](#)

[Le tecnologie wireless per i sistemi di misura distribuiti: Bluetooth ZigBee WiFi](#)

[IEEE 802.15.4](#)

[Lo Standard IEEE 802.15.4](#)

[Una rassegna di sistemi di data collection basati su tecnologia ZigBee](#)

GSM, UMTS, LTE, 3G, 4G

[Telefonia cellulare](#)

[GSM - Global System for Mobile Communications e 2G](#)

[GPRS - General Packet Radio Service](#)

[UMTS - Universal Mobile Telecommunications System e 3G](#)

[LTE - Long Term Evolution, WiMAX e 4G](#)

I - Cablaggio strutturato

Prefazione

Alcuni degli esercizi dei seguenti appunti fanno riferimento al materiale del corso Router & Switch della CISCO seguito durante l'a.s. 2016/2017 sul sito netacad.com. Dato che tali esercizi trattano bene gli argomenti di ripasso, sono stati presi come riferimento. Si consiglia di rivedere comunque gli argomenti indicati nelle note sul corso CISCO nel sito netacad.com.

Gli appunti fanno riferimento al lavoro di molte persone che ringrazio soprattutto per la condivisione dei loro sforzi: la Prof.ssa [Sophia Danesino](#), il Prof. [Luigi Alcuri](#), il Prof. [Nicola Ceccon](#), Pietro Gallo e Fabio Salerno.

Esercizi di ripasso

24 - Esercizi generali su netacad.com

Come ripasso generale si suggerisce di rileggere i capitoli 7, 8, 9, 10 e 11 del corso CCNA1 sul sito [netacad.com](#). Come esercizi di ripasso si svolgono i laboratori e i Packet Tracer dei capitoli indicati, considerando le seguenti indicazioni per l'importanza di ogni esercizio, indicata nella colonna **Val** (1 = poco, 2 = media, 3 = alta).

Capitolo 7 - IP Addressing

PT	Lab	Descrizione	Val
	7.1.2.8	Using the Windows Calc. with Network addresses - fa fare un po' di conversioni decimale-esadecimale-binario usando il Windows Calulator (calc.exe).	1
	7.1.2.9	Converting IPv4 addresses to Binary - fa fare a mente un po' di conversioni decimale-binario	3
7.1.3.8		Investigating Unicast, Broadcast and Multicast traffic	2
	7.1.4.9	Identifying IPv4 addresses - fa studiare vari indirizzi IPv4 con S.M. e dire se sono di rete, di host o di broadcast	3
7.2.4.9		Configuring IPv6 addressing	3
	7.2.5.3	Identifying IPv6 addresses - offre un ripasso degli IPv6 più comuni, ne fa studiare alcuni e li mostra su un PC	3
	7.2.5.4	Configuring IPv6 addresses on Network Devices - fa configurare e provare gli IPv6 su una rete con due PC, uno Switch e un Router, come 6.5.1.2; da adattare ai POD!	3
7.3.2.5		Verifying IPv4 and IPv6 addressing	1
7.3.2.6		Pinging and Tracing to Test the Path - 2er	2
	7.3.2.7	Testing Network Connectivity with Ping and Tracer - su una rete POD con tre Router, due Switch e due PC (da emulare con gli Switch) fa configurare e provare gli IPv4	3
	7.3.2.8	Mapping the Internet - fa fare un po' di ping sul web	1
7.3.2.9		Troubleshooting IPv4 and IPv6 Addressing	2
7.4.1.2		Skills Integration Challenge	3

Capitolo 8 - Subnetting IP Networks

PT	Lab	Descrizione	Val
	8.1.4.6	Calculating IPv4 Subnets - fa determinare il n. di host e vari altri dati su vari tipi di subnet, proponendo 6 esercizi	3
8.1.4.7		Subnetting Scenario FLSM	2
	8.1.4.8	Designing and Implementing a Subnetting... Scheme - sulla solita rete con due PC, uno Switch e un Router (da adattare suo POD) fa svolgere un FLSM e configurarlo	3

8.2.1.4		Designing and Implementing a VLSM addr. Scheme	3
	8.2.1.5	Designing and Implementing a VLSM addr. Scheme - su una rete simil-POD con tre Router fa svolgere un VLSM	3
8.3.1.4		Implementing a Subnetted IPv6 addressing Scheme	3
8.4.1.2		Skills Integration Challenge (IPv4 and IPv6)	3

Capitolo 9 - Transport Layer

PT	Lab	Descrizione	Val
	9.2.1.6	Using Wireshark to Observe a TCP 3-way Handshake (7pp) - fa catturare e analizzare una connessione TCP	3
	9.2.3.5	Using Wireshark to Examine a UDP DNS Capture (6pp) - su un PC fa catturare e analizzare richieste e risposte DNS	3
	9.2.4.3	Using Wireshark to Examine FTP & TFTP Captures (14pp) - NB: nel Lab Manual il titolo è diverso. Fa esaminare traffico downloadato con FTP dal Server "ftp.cdc.gov", e salvare la configurazione di uno Switch su un PC con TFTP32	3
9.3.1.2		TCP and UDP Communications - 0p	2

Capitolo 10 - Application Layer

PT	Lab	Descrizione	Val
	10.1.2.5	Researching Peer-to-peer File Sharing (4pp) - fa rispondere a domande generali sulle reti P2P	1
10.2.1.7		Web and Email - 26p	3
10.2.2.7		DHCP and DNS Servers - 19p	3
	10.2.2.8	Observing DNS Servers (6pp) - fa osservare la conversione DNS operata da un browser, e usare due sole nslookup	3
10.2.3.3		FTP Servers - 18p	3
	10.2.3.4	Exploring FTP (10pp) - fa usare FTP da prompt di DOS, da un browser, e da una piccola App scaricata da Internet	3
10.3.1.2		Explore a Network - 0p	2
10.3.1.3		Multiuser - Tutorial (Client & Server side)	1
10.3.1.4		Multiuser - Implement Services (Client & Server side)	1

Capitolo 11 - Build a Small Network⁸⁹

PT	Lab	Descrizione	Val
	11.2.2.6	Researching Network Security Threats (4pp) - fa esaminare alcune pagine e altre risorse sul sito di www.SANS.org	1
11.2.4.5		Configuring Secure Passwords and SSH - 19p	3
	11.2.4.6	Accessing Network Devices with SSH (11pp) - fa configurare e usare SSH su una rete fatta da un Router, uno Switch e un PC (emulabile sullo Switch, per operare sui POD)	3

⁸⁹ In rosso gli esercizi utili anche per la sicurezza.

	11.2.4.7	Examining Telnet and SSH in Wireshark (11pp) - fa connettere da un PC, coi due protocolli, a un Router, ed esaminare il traffico catturato con Wireshark	2
	11.2.4.8	Securing Network Devices (13pp) - sulla solita rete fatta da Router+Switch+PC fa configurare vari settaggi di sicurezza	3
11.3.2.3		Test Connectivity with Traceroute - 1er	2
	11.3.2.4	Testing Network Latency with Ping and Traceroute (6pp) - fa esaminare alcune latenze sul web con ping e traceroute anche estesi da PC, catturando gli output su file .txt	2
11.3.3.3		Using "show" Commands - 0p	1
	11.3.4.6	Using the CLI to Gather Network Device Information (14pp) - fa raccogliere un po' di informazioni da Router e Switch usando vari comandi comandi IOS show e DOS CLI	2
	11.4.3.5	Troubleshooting Connectivity Issues (19pp) - Questo Lab è "recommended", e fa cercare vari problemi (D.G. mancante, IP errato, duplex mismatch, ecc) e li fa rimediare	3
11.4.3.6		Troubleshooting Connectivity Issues - 3p in chiaro	2
11.5.1.2		Skills Integration Challenge - 69p	3
11.5.1.3		Troubleshooting Challenge - 11p in chiaro	3

25 - Subnetting e VLAN

Progettare con Packet Tracer, la simulazione di una piccola **rete dell'università di POLISTUDIO**.

La rete prevede:

- due laboratori per gli studenti:
 - LABINFO con 5 PC al 1° piano e un PC TOTEM (rilevazione presenze) collegato a UFFDOCENTI;
 - LABTLC con 5 PC al piano terra e un PC TOTEM (rilevazione presenze) collegato a UFFDOCENTI;
- due uffici:
 - UFFDOCENTI con 3 PC al piano terra e 2 PC al 1° piano;
 - UFFSEGRETERIA 2 PC al 1° piano e 3 PC al 2° piano.

Le reti dei laboratori e delle segreterie prevedono i seguenti indirizzi:

NOME	Primo IP disponibile Host	GATEWAY	VLAN
LABINFO	192.168.10.1/24	192.168.10.254/24	10
LABTLC	192.168.20.1/24	192.168.20.254/24	20
UFFDOCENTI	192.168.50.1/24	192.168.50.254/24	50
UFFSEGRETERIA	192.168.60.1/24	192.168.60.254/24	60

Progettare e realizzare con un simulatore di rete le seguenti soluzioni:

4. utilizzare le VLAN usando router-on-a-stick;
5. utilizzare le VLAN usando switch L3;
6. utilizzare solo switch L2 e un router per fare subnetting usando DHCP.

Si realizzi⁹⁰ anche una relazione scritta dove si confronteranno le diverse soluzioni spiegandone i vantaggi e gli svantaggi, sia a livello implementativo, sia a livello di costi (cercare su Internet informazioni al riguardo)

⁹⁰ Questa parte deve essere consegnata anche al docente di teoria in formato elettronico.

Sistemi di cablaggio strutturato

La diffusione e la frequenza con cui le installazioni di reti per le telecomunicazioni hanno seguito lo sviluppo delle tecnologie di trasmissione dei dati, hanno posto in evidenza la necessità di realizzare sistemi di cablaggio adeguati alle attuali richieste, relative alla qualità e alle specifiche tecniche dei dispositivi di comunicazione, ma anche tali da poter adattarsi con elevata flessibilità all'evoluzione tecnologica.

L'elevata dinamicità ed eterogeneità dalle aziende odierne necessitano dell'integrazione di elementi che trasmettono e ricevono segnali di tipo diverso (fonia, dati, antincendio, sicurezza e controllo, video, controllo macchine) e che non possono più prevedere l'aggiunta di impianti con cablaggi utilizzati in modo esclusivo e separato. Risulta bensì necessario **integrare i segnali diversi in un'unica rete di trasmissione**. Il sistema che risponde a questi requisiti è quello del **cablaggio strutturato**.

La dicitura **cablaggio strutturato** designa un **sistema di telecomunicazione** che **attui e garantisca l'interconnessione di dispositivi fra loro eterogenei, assicurando servizi con elevate velocità di trasmissione, modularità, espandibilità e sicurezza**.

Un progetto basato sui principi del cablaggio strutturato e dell'integrazione dei servizi elimina la distinzione tra impianto telefonico e rete dati (anche di molteplici tipologie), creando un **impianto integrato** indipendente dalle applicazioni e dai sistemi utilizzatori.

In sintesi, la principale differenza tra un **sistema** di tipo **strutturato** ed uno di tipo tradizionale consiste nel fatto che il primo **non** risulta **vincolato ad una sola tipologia di protocollo dati o**, più in generale, **ad una sola tipologia di servizio**. Tale caratteristica offre ai sistemi strutturati la versatilità necessaria per soddisfare le diverse esigenze di interconnessione di dispositivi: ciò verso cui si tende è la possibilità di offrire, tramite un'unica struttura, una molteplicità di servizi, in ambito commerciale e domestico, interfacciando apparecchiature di tipo diverso.

L'**attività normativa** che regola la realizzazione di **sistemi di cablaggio strutturato** si è **sviluppata in origine negli Stati Uniti**, su proposta delle varie industrie elettroniche del settore, che esprimevano in tal modo la necessità di un unico standard cui far riferimento durante la fase di progettazione e di successiva installazione. Tale attività è stata sintetizzata e recepita negli **standard TIA/EIA** (*Telecommunications Industry Association, Electronics Industry Association*), che forniscono a tutt'oggi le linee guida alle varie compagnie.

L'**equivalente europeo** di tale organismo è il **Cenelec**, la commissione europea per la standardizzazione elettrotecnica, che tramite i comitati tecnici ha fornito le norme per il mercato europeo. Infine si hanno gli **standard internazionali dell'ISO**, molto **simili** a quelli del **Cenelec**, che definiscono sistemi generici di cablaggio, indipendentemente dal tipo di applicazione.

In **Italia** si fa riferimento all'attività del **CEI** (*Comitato Elettrotecnico Italiano*) che recepisce tramite i propri comitati tecnici, le norme europee.

Nello specifico, i diversi standard e normative di riferimento sono:

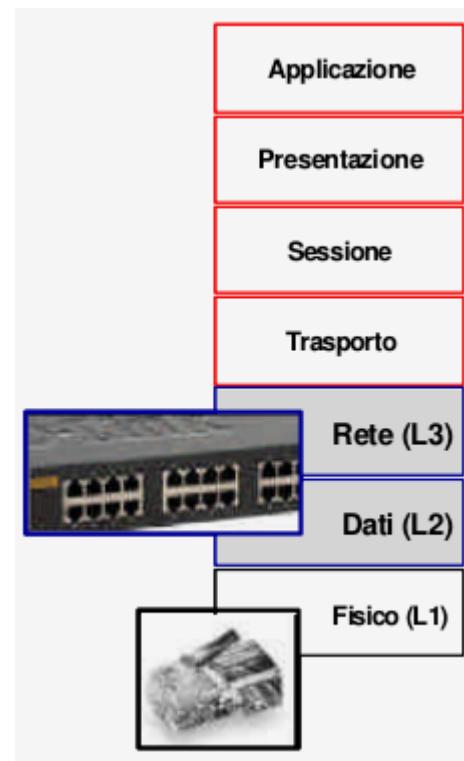
- la **normativa americana EIA/TIA 568A** (e **568B** dal 2002);
- la **normativa europea EN50173** (e **EN50173-1** dal 2002);
- la **normativa internazionale ISO/IEC 11801**⁹¹.

L'insieme di tali norme può venire applicato sia ad un singolo edificio, sia ad un comprensorio (*campus*), che includa diverse strutture; si vengono così a definire delle aree interconnesse, in cui i singoli utenti, dalle loro postazioni di lavoro possano interagire fra loro.

Il rispetto della normativa nell'installazione di un sistema di cablaggio strutturato è necessario per avere la **certificazione in ambito comunitario**.

Il **cablaggio strutturato** è un insieme di **componenti passivi standard** (cavi, connettori, prese...) posti in opera per poter supportare diverse applicazioni di telecomunicazione, come 100Base-TX (Fast Ethernet), 1000Base-T (Gigabit Ethernet) e una centralina telefonica digitale o analogica.

È importante sottolineare che **le apparecchiature attive**, come switch o hub, e **gli apparecchi terminali dell'applicazione**, come il telefono, **non fanno parte del cablaggio strutturato**: il **cablaggio strutturato**, infatti, **si pone al livello fisico, cioè al primo livello del modello ISO/OSI**.



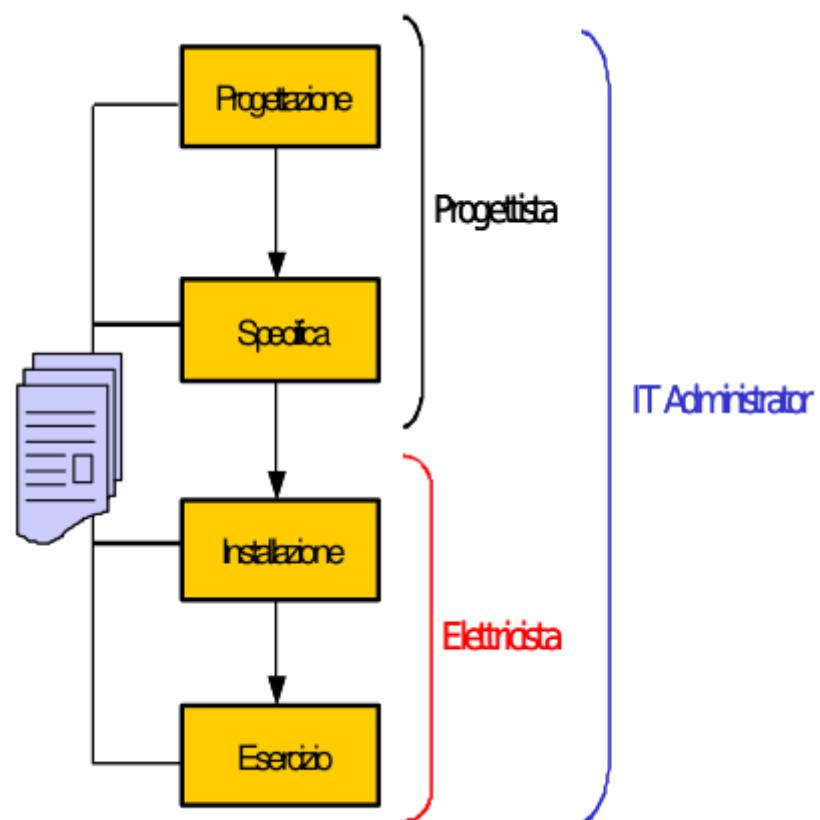
Fasi del cablaggio strutturato

Nella realizzazione di un impianto di cablaggio strutturato si possono individuare le seguenti fasi a cui corrispondono anche diverse figure professionali (progettista, elettricista, collaudatore, responsabile della rete o del sistema informativo, acc.):

1. la **progettazione**, cioè la scelta dei componenti e loro configurazione in base alle dimensioni dell'impianto e delle applicazioni veicolabili;
2. la **specifica dettagliata dell'impianto** che deve individuare dove e come installare l'impianto e il livello qualitativo dei componenti in termini di prestazioni che devono garantire;
3. **installazione** in base alle specifiche;
4. **collaudo** in base ai criteri stabiliti dallo standard scelto;
5. **esercizio** che riguarda la gestione e la manutenzione dell'impianto.

⁹¹ Nel seguito del testo, per semplificare, le normative saranno individuate con le seguenti sigle: EIA/TIA 568A -> EIA, EN50173-1 -> EN, ISO/IEC 11801 -> ISO.

Ogni fase viene corredata da una opportuna documentazione di riferimento, comprendente per esempio lo schema dell'impianto e la certificazione.



Specifiche generali delle normative e degli standard

Le normative che regolano i sistemi di cablaggio sono applicabili ad un singolo edificio privato o ad un gruppo di edifici facenti parte della stessa area privata (insediamento o comprensorio o campus) e definiscono i requisiti minimi o specifiche tecniche per il cablaggio, oltre a specificare i limiti geografici e temporali:

- estensione geografica (3000m/EIA – 2000m/EN);
- superficie massima edifici (1.000.000m²/EIA);
- popolazione massima di un edificio (50.000/EIA);
- validità impianto anni (10/EIA – più a lungo possibile/EN).

La normativa EN non stabilisce limiti di utenza e di validità.

Le specifiche degli standard riguardano i seguenti aspetti:

- struttura generica di un sistema di cablaggio, cioè topologia e elementi;
- caratteristica dei mezzi trasmissivi;
- prestazioni degli elementi del cablaggio e del cablaggio in termini di prescrizioni (norme, regole, precetti e requisiti obbligatori);
- le norme di installazione;
- norme per l'identificazione degli elementi e componenti del cablaggio;

- norme per la documentazione del progetto e dell'impianto
- norme per il collaudo dell'impianto di cablaggio.

Nel seguito verranno analizzati alcuni degli aspetti indicati.

Struttura generica di un sistema di cablaggio

Un sistema di cablaggio generico è costituito da **elementi funzionali** e **gruppi di elementi funzionali** tra loro **interconnessi**, allo scopo di formare dei **sottosistemi del cablaggio**.

Gli elementi funzionali (passivi) nella **normativa EN** sono i seguenti:

1. **CD** – *Campus Distributor* (distributore di campus);
2. **Cavo di dorsale di campus** (cavo per il cablaggio tra edifici);
3. **BD** – *Building Distributor* (distributore di edificio);
4. **Cavo di dorsale di edificio** (cavo per il cablaggio dell'edificio);
5. **FD** – *Floor Distributor* (distributore di piano);
6. **Cavo orizzontale** (cavo per il cablaggio del piano);
7. Opzionali:
 - a. **CP** – *Consolidation point* (punto di transizione)⁹²;
 - b. **Cavo CP** – Cavo del punto del *consolidation point*;
 - c. **MUTO** – *Multi-User Telecommunications Outlet* (punto di terminazione formato da un insieme di TO)⁹³;
8. **TO** – *Telecommunications Outlet* (presa di telecomunicazione, cioè la presa utente a cui si collega una apparecchiatura terminale TE).

Nella **normativa EIA** gli elementi funzionali hanno nomi diversi pur svolgendo lo stesso ruolo, la tabella seguente illustra le differenze tra i principali elementi.

EN	EIA
CD – Campus Distributor	MC – Main Cross Connect
BD – Building Distributor	IC – Intermediate Cross Connect
FD - Floor Distributor	HC – Horizontal Cross Connect
CP – Consolidation point	CP – Consolidation point
MUTO – Multi-User Telecommunications Outlet	MUTOA – Multi-User Telecommunications Outlet Assembly

⁹² Il Consolidation Point (CP) è un sistema passivo di prese (permutatore) posto tra FD e TO. Viene usato negli open space dove è richiesta flessibilità nello spostamento delle TO, e rappresenta un punto di connessione intermedio sul cavo orizzontale. Vincoli: non sono ammesse connessioni dirette tra apparecchi attivi, quindi **non è possibile collegare direttamente al CP un computer**, possono essere servita al massimo 12 aree di lavoro, la distanza minima tra CP e un distributore (FD) deve essere di 15m, è ammesso un solo CP tra FD e qualsiasi TO servita dal distributore, deve essere posizionato in un'area accessibile all'utente. [Posizionamento di un CP](#).

⁹³ Si ricorre all'utilizzo del MUTO se i posti di lavoro cambiano spesso di posizione o di numero, come negli open space e nelle sale riunioni, oppure quando un utente ha necessità di molte prese di telecomunicazione. **AI MUTO si collegano direttamente le postazioni di lavoro.** Vincoli: può servire al massimo 12 aree di lavoro, non deve essere installato in un controsoffitto o sottopavimento, deve essere posizionato in un'area accessibile all'utente, il cavo che va dalla stazione di lavoro al MUTO non può superare i 20m. [MUTO e CP, cablaggio orizzontale, CP e MUTO](#).

TO – Telecommunications Outlet

TO – Telecommunications Outlet

I diversi elementi funzionali, gli apparati attivi e i terminali sono collegati fra loro da **elementi di connessione** che sono costituiti da dispositivi usati per connettere cavi o elementi di cavo, ad esempio i connettori RJ45 oppure i giunti per la fibra ottica.

La topologia del cablaggio

La topologia di una rete è una rappresentazione sia fisica, sia logica del sistema di interconnessione dei cavi per supportare i servizi richiesti. Non necessariamente una topologia logica impone direttamente la stessa topologia fisica, è possibile, ad esempio, installare una topologia logica a stella tramite canalizzazione di rete a bus.

Il **sistema di cablaggio** è costituito da una **struttura a stella gerarchica**, che include un insieme di **sottosistemi locali**, anch'essi gerarchicamente distribuiti e che complessivamente sono organizzati su tre livelli:

1. il centro stella di comprensorio costituito dal **CD**;
2. il centro stella di edificio costituito dal **BD**;
3. il centro stella di piano costituito dal **FD**.

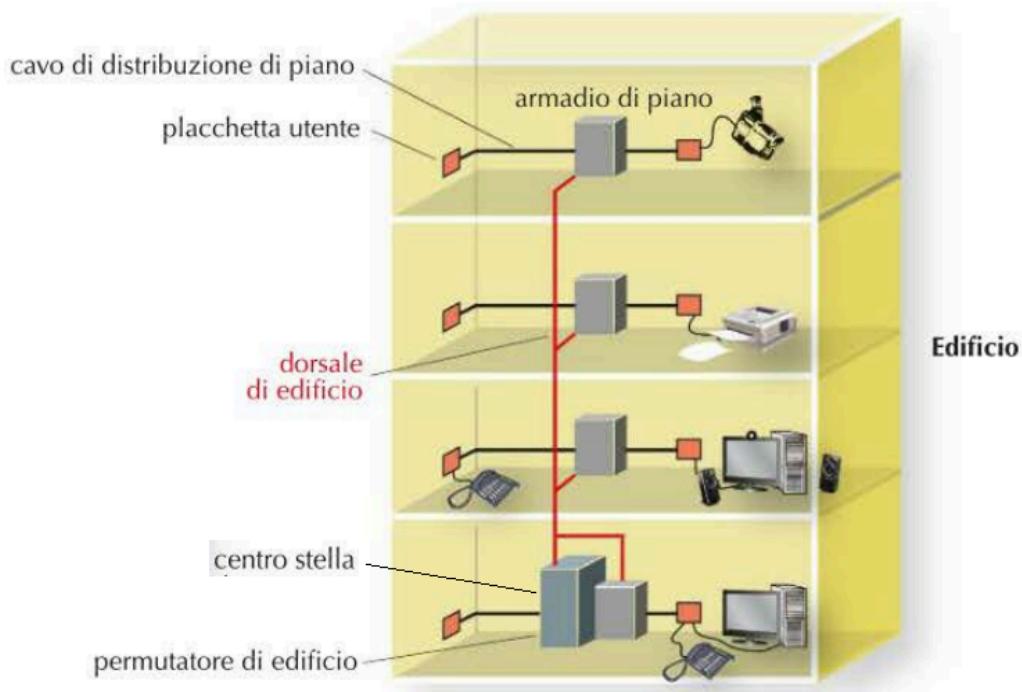
Fra gli elementi del cablaggio possono essere presenti anche:

- il **locale tecnico (ER - Equipment Room)** che può contenere apparati di rete e centralini telefonici;
- il **punto di presenza (POP - Point of Presence)** che identifica il locale in cui il gestore della rete, sia dati, sia voce, ha attestato le linee per fornire i servizi all'interno dell'edificio;
- il **centralino telefonico (PBX - Private Branch eXchange)**.

Un sistema di cablaggio strutturato è anche composto, in maniera schematica ed essenziale, da due strutture:

- un **cablaggio orizzontale (horizontal cabling systems structure)** che si sviluppa in ogni singolo piano dell'edificio con lo scopo di collegare le prese delle singole postazioni di lavoro all'armadio di piano;
- un **cablaggio verticale (backbone cabling system structure)** che **si distingue** ulteriormente in **dorsali di edificio** e **dorsali di comprensorio**, e collega fra loro tutti gli armadi di piano con il centro di distribuzione dell'edificio e con la dorsale di comprensorio.

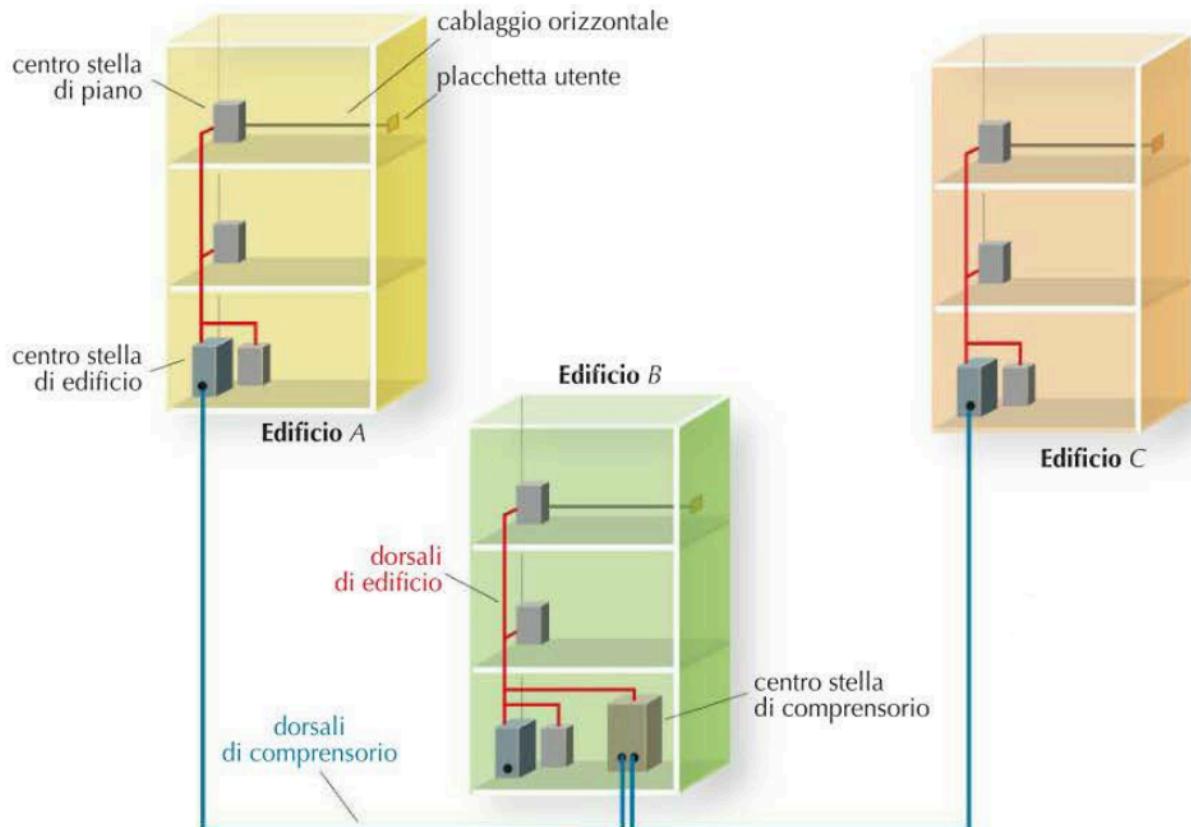
La seguente immagine mostra uno schema di cablaggio strutturato di un singolo edificio. Dal **centro stella**, posto al piano terreno, partono le **dorsali verticali** in fibra ottica verso i piani, mentre ogni piano è dotato di un **centro stella di piano** cui confluiscono le **dorsali orizzontali** di piano in rame.



La gerarchia di questo cablaggio è costituita da due livelli:

1. il livello di edificio con il centro stella di edificio (**BD** - *Buildings distributor*);
2. il livello di piano con il centro stella di piano (**FD** - *Floor Distributor*).

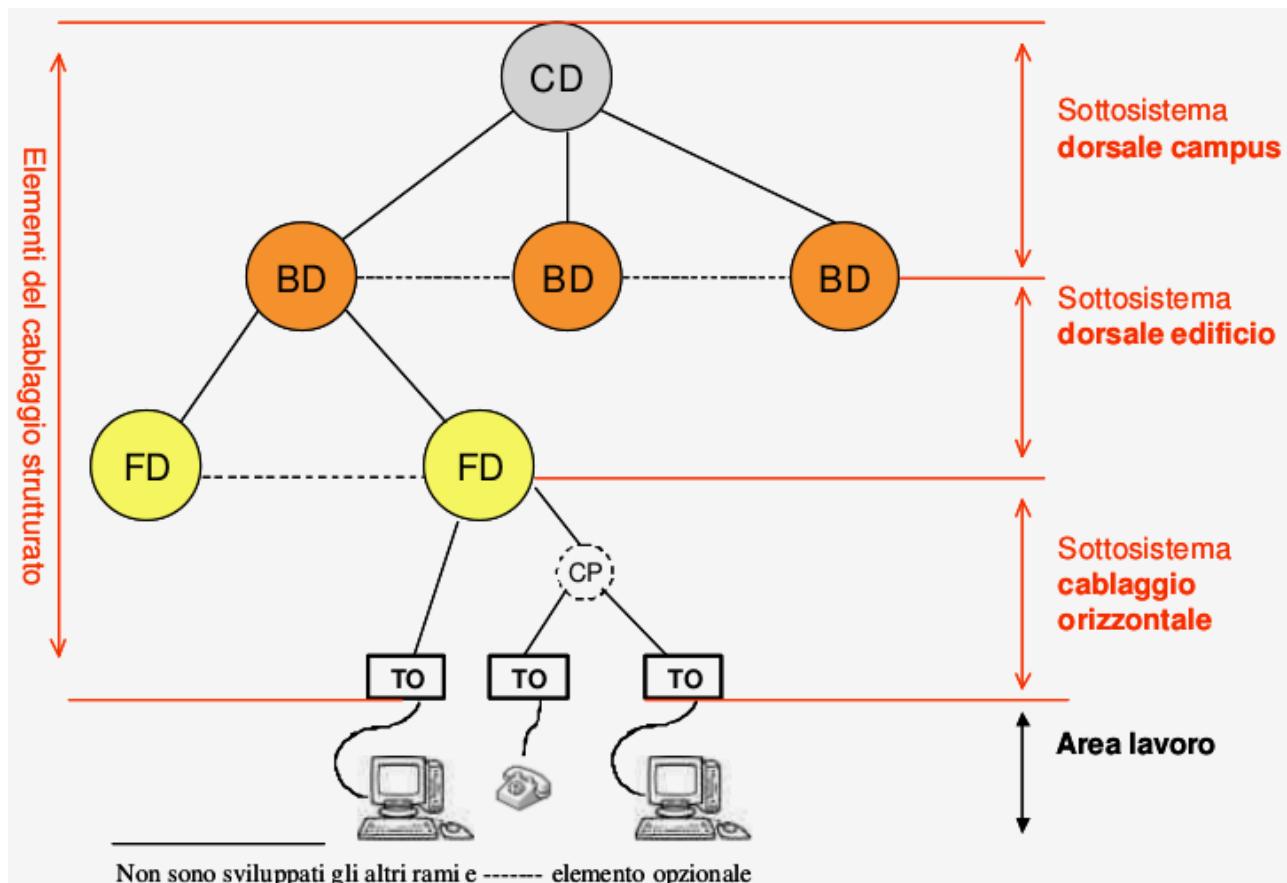
Nell'immagine seguente è invece mostrato il cablaggio strutturato di comprensorio costituito da tre edifici. Nell'edificio B è presente il **centro stella di comprensorio** collegato in fibra ottica ai **centri stella di edificio** degli edifici A e C.



La gerarchia di questo cablaggio è costituita da tre livelli:

1. il livello di comprensorio con il centro stella di comprensorio (**CD** – *Campus Distributor*);
2. il livello di edificio con il centro stella di edificio (**BD** – *Buildings distributor*);;
3. il livello di piano con il centro stella di piano (**FD** – *Floor Distributor*).

Il cablaggio strutturato di un edificio può essere sostanzialmente schematizzato attraverso un **albero** che ne evidenzi la **struttura gerarchica**:



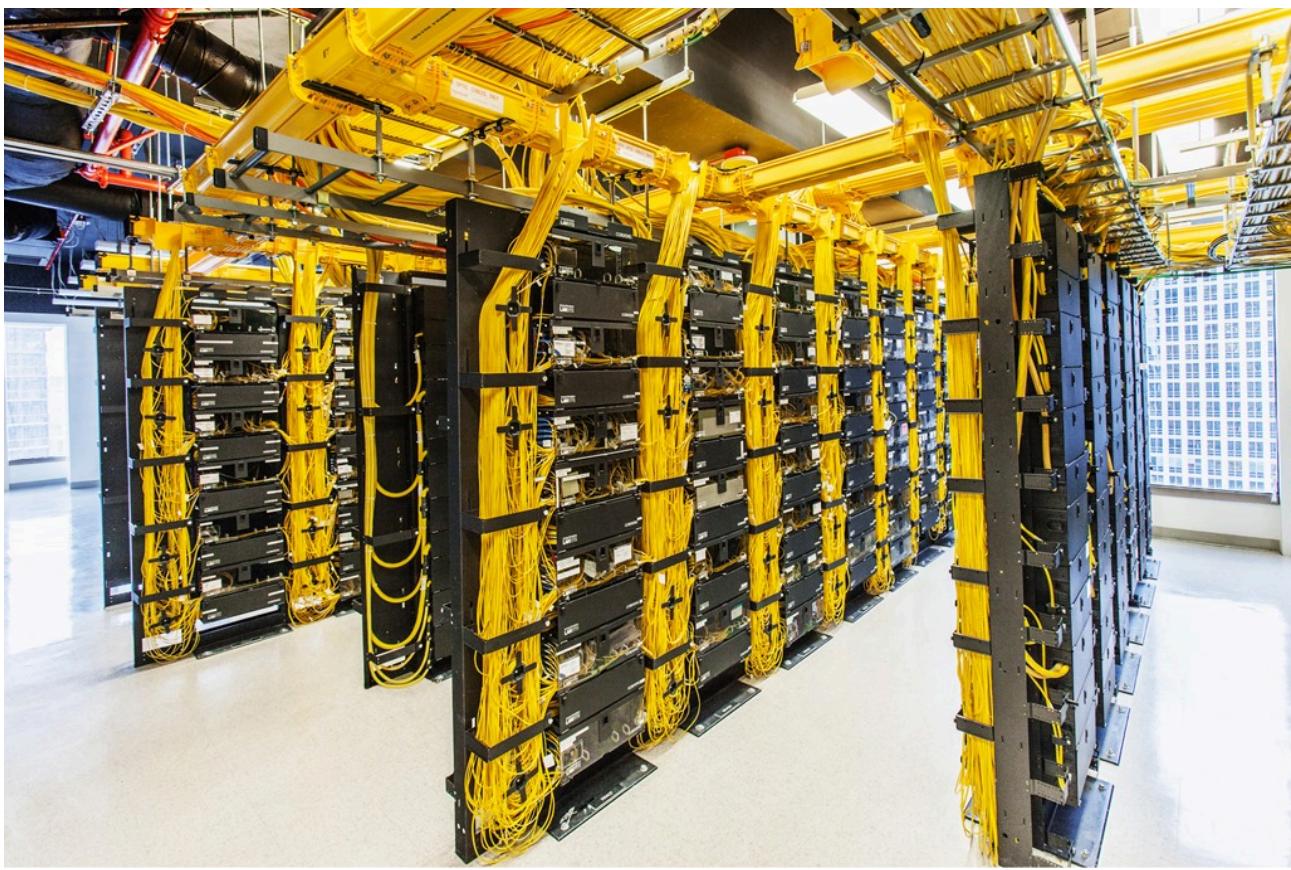
I **collegamenti orizzontali**, che formano un collegamento diretto tra gli **FD** o tra i **BD**, se presenti, **devono essere di riserva e supplementari** a quelli per la topologia base costituita da un albero gerarchico. Questi collegamenti costituiscono una **ridondanza implementata per motivi di sicurezza e affidabilità** allo scopo di fornire una certa protezione contro rischi.

Gli elementi del cablaggio

Le **dorsali (backbone)** sono gli **elementi portanti del cablaggio** e possono interconnettere edifici diversi (**BD**) con l'edificio centro stella del comprensorio (**CD**) costituendo lo **interbuilding backbone**, o armadi di piano diversi (**FD**) con l'armadio di piano di edificio (**BD**).

Nel **centro stella di comprensorio (CD)** risiede il permutatore principale (**MDF - Main Distribution Frame**) da cui **partono le dorsali di comprensorio (interbuilding**

backbone) che vanno verso gli altri edifici della rete e che può avere anche dimensioni considerevoli, a seconda delle dimensioni dell'azienda.



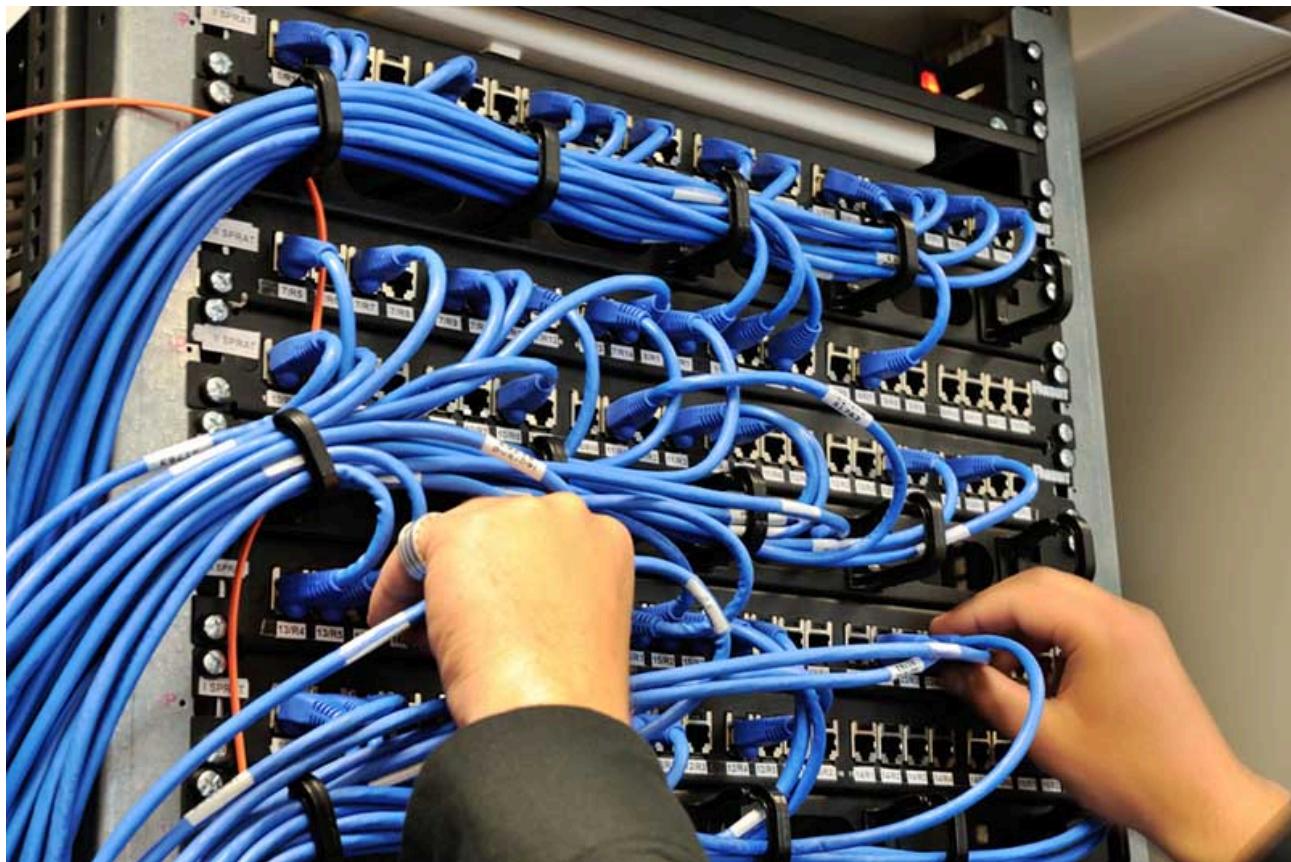
Il permutatore principale (**MDF**) è costituito da blocchi di connessione (strisce di permutazione) su cui si attestano i cavi provenienti dalla rete esterna, il cui segnale verrà inoltrato all'interno della LAN.

Il **centro stella di edificio (BD)**, che rappresenta il secondo livello nella gerarchia del cablaggio, è costituito da un **permutatore intermedio (IDF - *Intermediate Distribution Frame*)** che è solitamente collocato in un **armadio rack**⁹⁴ dotato di adeguati **patch panel**⁹⁵. Il permutatore intermedio (**IDF**) ha lo scopo di ricevere le linee provenienti dal centro stella di comprensorio, **attestarle**⁹⁶ e consentire il collegamento a tutto l'edificio, costituendo la **dorsale di edificio (intrabuilding backbone)**. Inoltre potrà anche realizzare il cablaggio orizzontale del piano su cui è collocato, oltre a quello verticale di edificio.

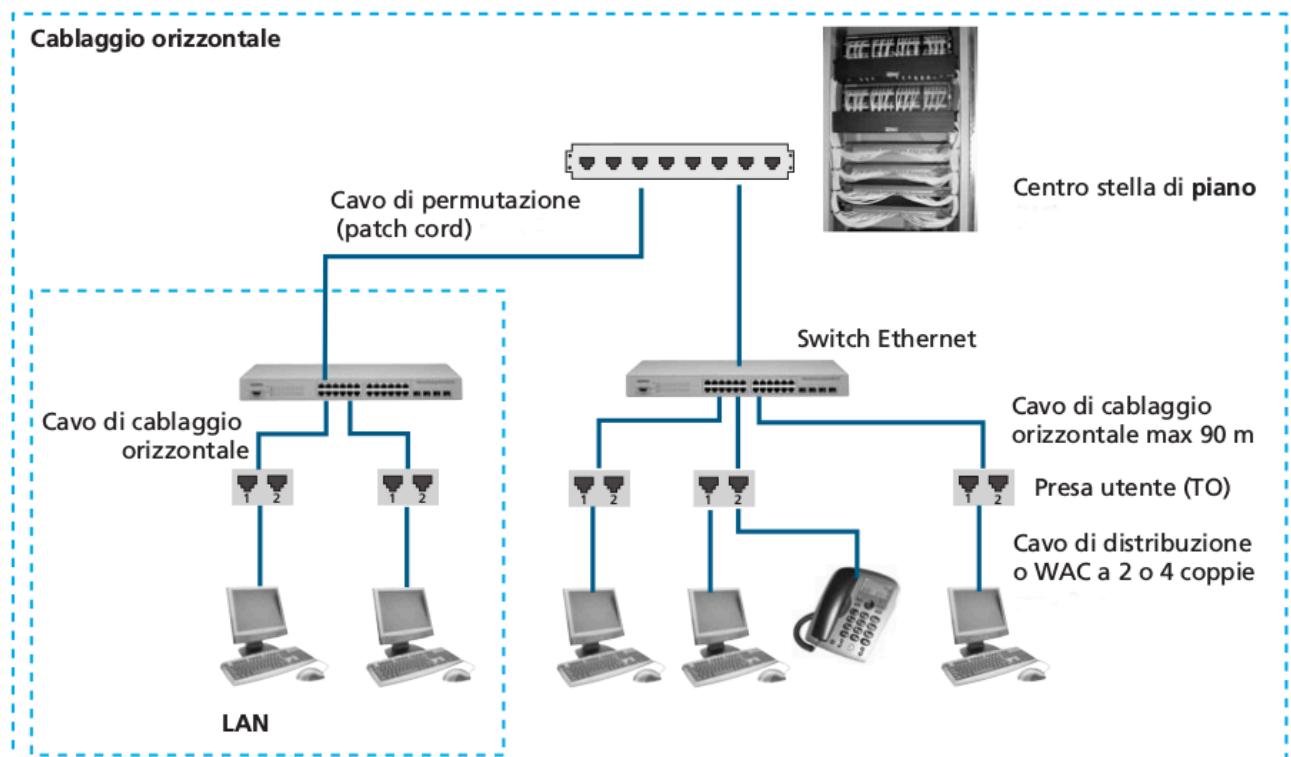
⁹⁴ Un **armadio rack** è un armadio a scaffali destinato a contenere server o apparati di rete.

⁹⁵ Il **patch panel** (pannello di permutazione) è un sistema che interfaccia e raccorda tra loro sezioni diverse di un cablaggio.

⁹⁶ **Attestare** un cavo significa terminarlo e fissarne una estremità.



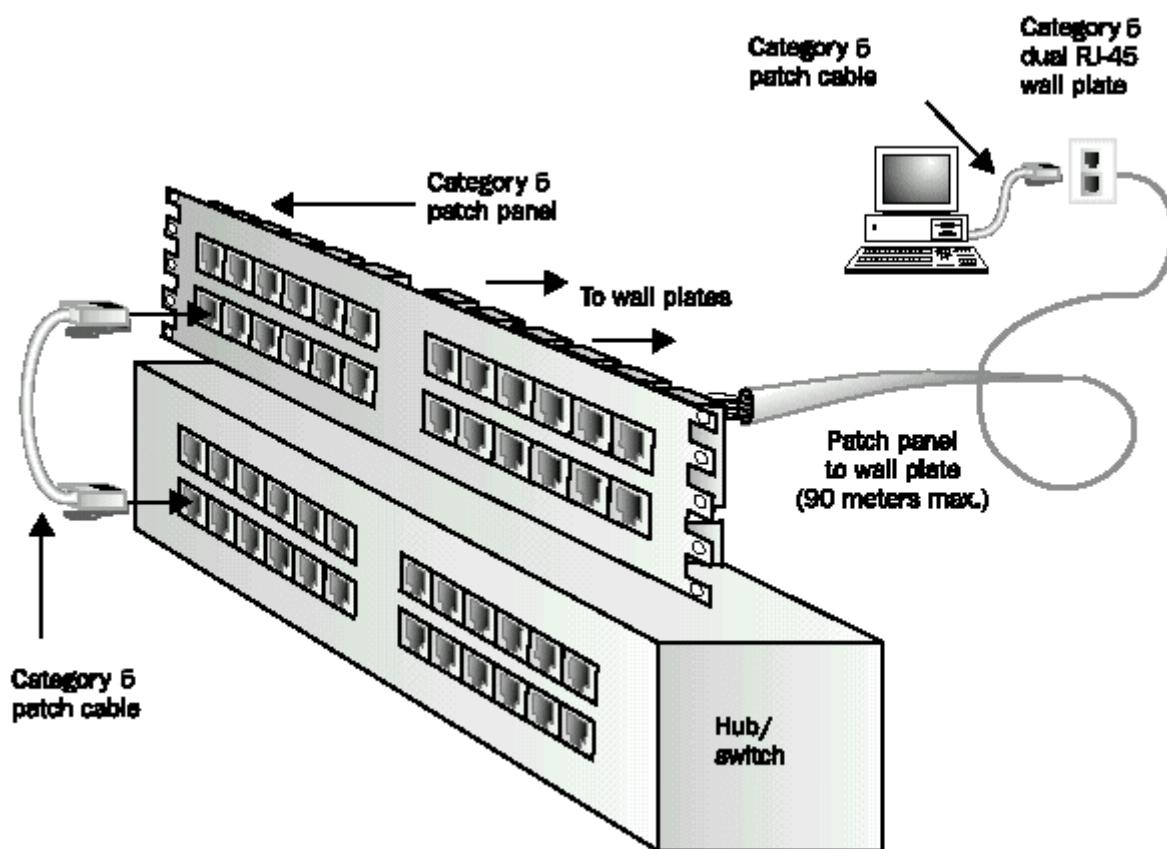
Il **centro stella di piano (FD)**, che rappresenta il terzo livello nella gerarchia, pur non costituendo una dorsale, è costituito normalmente da un **armadio rack** di medie dimensioni, a pavimento o a parete, da cui **parte il cablaggio orizzontale verso le prese utente (TO - Telecommunication Outlet)**.



Nell'**armadio rack** vi è un **patch panel** dotato di tante prese quante sono almeno le postazioni degli utenti.

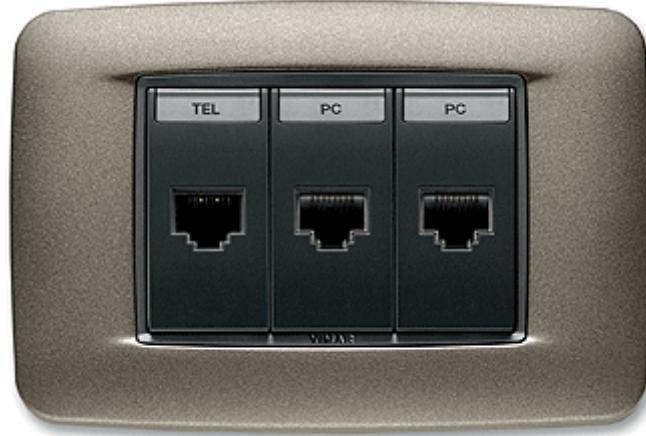


Dal **patch panel** è possibile effettuare un **collegamento** all'apparato di rete attivo (**switch**) mediante delle bretelle (**patch cord**). I **patch panel** possono essere anche molteplici se sul piano sono presenti diversi apparati di rete che dovranno essere attestati al **centro stella di piano (FD)**.



Per raggiungere le **prese utente (TO - Telecommunication Outlet)** dal **patch panel** normalmente si utilizzano cavi di interconnessione che sono parte integrante del cablaggio orizzontale.

Attraverso la **presa utente (TO - Telecommunication Outlet)** si raggiunge la **postazione di lavoro (WA - Work Area)** che viene collegata alla presa utente attraverso un **cavo di interconnessione (WAC - Work Area Cable)**.



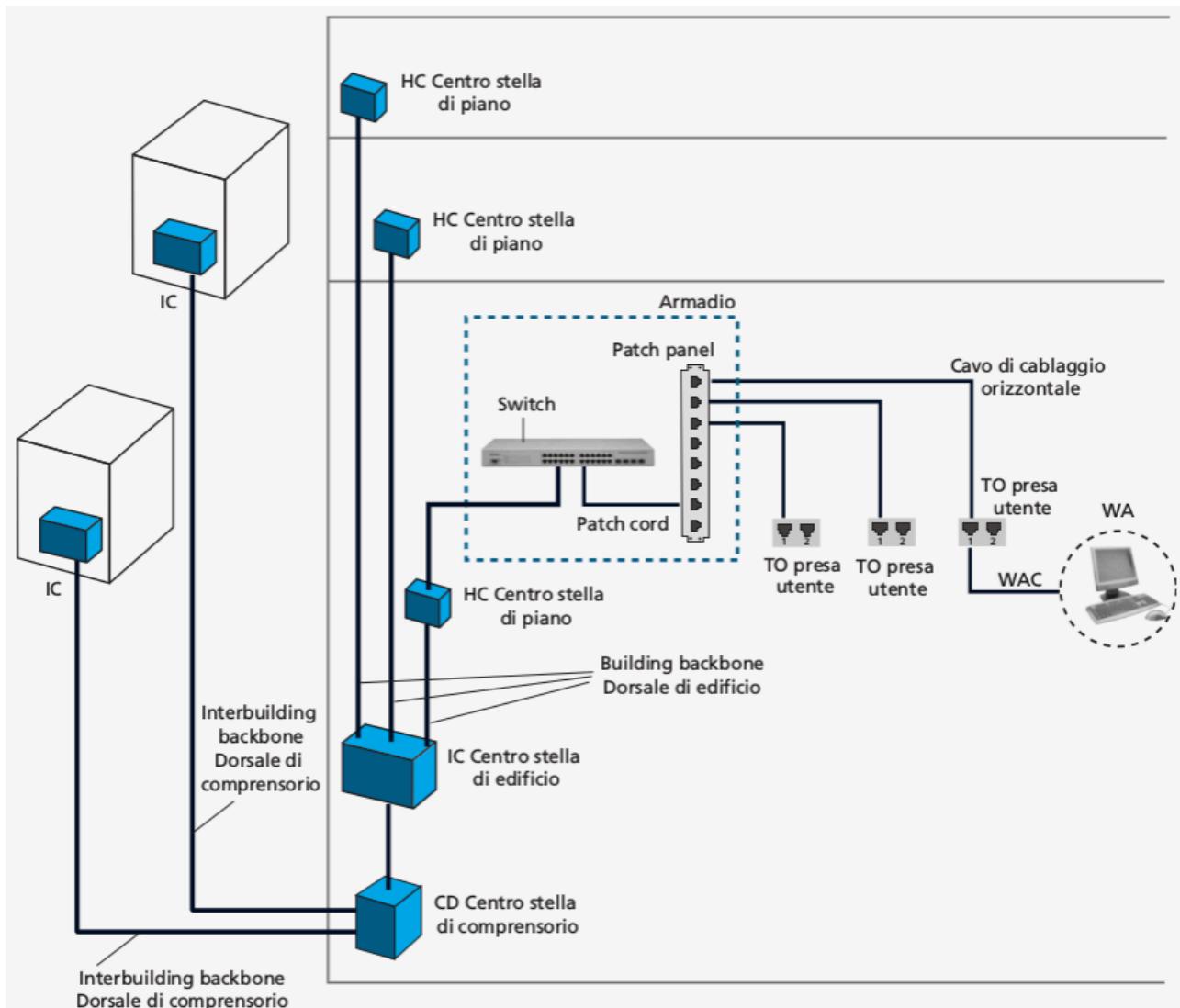
La presa **TO** è una presa telematica, su supporto modulare, che può essere attivata per erogare un qualsiasi servizio, e il cui design deve essere compatibile alle placche dell'impianto elettrico. Ogni modulo deve essere identificato opportunamente.

In sede di progettazione si devono **distribuire le postazioni di lavoro (WA) in base alla superficie utilizzabile** cercando di ottenere una elevata densità di **TO** con l'obiettivo di aumentare la **capacità del cablaggio di adattarsi ai cambiamenti** futuri. L'unità di **densità** universalmente riconosciuta dei **posti di lavoro** è di **uno ogni 10m²** dedicati agli uffici, come specificato nella normativa EIA.

Nella normativa EN ogni **WA** deve essere servita da almeno due prese utente (**TO**), tipicamente dati e fonia, per il collegamento di apparati attivi, utilizzando una presa con 8 contatti (RJ45 per doppino di categoria 6 o superiore), oppure una [presa SC o LC](#) per cavo in fibra, simplex o duplex.

Infine il **WAC**, pur essendo utilizzato per collegare i terminali, **non è considerato parte del sottosistema di cablaggio** essendo specifico di una applicazione. Il **contributo prestazionale di questo cavo di lavoro deve comunque essere considerato** nella realizzazione della struttura di cablaggio e può essere costituito da un doppino con plug RJ45 ad entrambe le estremità, o da un cavo bifibra con terminazione SC-D o LC-D ad entrambe le estremità.

La seguente immagine schematizza un cablaggio con tutti gli elementi descritti.



Si osservi che le connessioni tra sottosistemi, rappresentati dai distributori, possono essere costituiti sia da apparecchiature attive, come gli switch, sia da elementi esclusivamente passivi come connettori.

Nel caso di utilizzo di apparati attivi questi si posizionano alle fine di ogni sottosistema di cablaggio, quindi in ogni distributore può essere presente un apparato attivo (**EQP - Equipment**) mentre questo non è possibile in un punto di transizione **CP**.

Gli **apparecchi attivi** possono fare parte del distributore, ma **non sono parte della normativa del cablaggio strutturato**, questo significa che le prestazioni dell'impianto di cablaggio strutturato non tengono conto delle performance dei dispositivi.

Dimensionamento WA

Il **dimensionamento** consiste nello stabilire il **numero di postazioni di lavoro (WA)**, **di prese utente (TO)** e **di cavi** dell'area di lavoro (WAC), **in base alle superfici da servire** e alle eventuale indicazioni sul **numero dei servizi** da attivare.

È possibile applicare le seguenti formule per ottenere il **numero di postazioni di lavoro (WA_n)** e il **numero di cavi per area di lavoro (WAC_n)** che corrisponde anche al **numero di prese utente (TO_n)**, data la **superficie totale (SupT)**, la **superficie dell'area di lavoro (SupWA = 10m² se non diversamente indicato)** e il **numero di servizi per area di lavoro (S_n = 2 se non diversamente indicato)**:

$$\mathbf{WA_n = SupT / SupWA}$$

$$TO_n = WA_n * S_n$$

$$WAC_n = TO_n$$

da cui si ricava che

$$\mathbf{WAC_n = TO_n = (SupT / SupWA) * S_n}$$

Si consideri il seguente esempio.

Un edificio è costituito di 3 piani le cui aree di lavoro sono rispettivamente di

- 100m² per il primo;
- 120m² per il secondo;
- 80m² per il terzo piano.

Utilizzando le formule, con i valori di default, si ottengono:

$$SupT = 100m^2 + 120m^2 + 80m^2 = 300m^2$$

$$\mathbf{WA_n = SupT / SupWA = 300m^2 / 10m^2 = 30}$$

$$TO_n = WA_n * S_n = 30 * 2 = 60$$

$$\mathbf{WAC_n = TO_n = 60}$$

Quindi nell'edificio sarà possibile posizionare **30 posti di lavoro (WA_n)** con **60 prese utente (TO_n)** e **60 cavi di collegamento (WAC_n)** tra le prese e i terminali (30 computer e 30 telefoni).

Un calcolo più preciso dovrebbe **considerare le aree dei singoli piani** invece di considerare in modo indifferenziato la superficie totale dell'edificio. Inoltre **non si può prescindere dalla piantina dell'edificio** da cui è possibile **individuare strutture architettoniche che** di fatto **possono variare il numero grezzo delle postazioni**; alcune di queste possono non essere di fatto utilizzabili, in quanto collocate, ad esempio, a ridosso di un termosifone o di una apertura di passaggio.

Fonia e dati

Il **patch panel** e i **cavi volanti** sono i due componenti che **permettono** la **permutazione**, cioè la possibilità di commutare le tratte di cablaggio per assegnare le applicazioni ai **TO** (*Terminal Outlet*) senza opere murarie e/o elettriche aggiuntive.

La linea telefonica permette di comprendere e valorizzare la flessibilità e l'efficacia di un impianto strutturato. Per esempio un **TO** assegnato ad un computer può essere riassegnata ad un telefono semplicemente spostando un cavo volante sul distributore, oppure un dipendente può cambiare ufficio all'interno dell'azienda, mantenendo il proprio numero solo spostando un cavo volante sul distributore.

Grazie alle permutazioni il cablaggio strutturato permette una rapida configurazione o riconfigurazione della distribuzione di dati e fonia, o di altri segnali, all'interno dell'edificio.

Il cablaggio strutturato permette un unico cablaggio per dati e fonia, ma a livello apparati e applicativo i servizi sono amministrati generalmente in modo autonomo. La voce è per lo più governata da un **PABX** (*Private Branch Exchange*), comunemente noto come centralino telefonico, mentre le comunicazioni dati sono trasmesse mediante apparati di networking.



Il **PABX** è la centralina privata di commutazione ed è una centralina telefonica digitale o analogica che fa da interfaccia tra la rete telefonica interna e la rete telefonica esterna; nasce quindi con lo scopo di collegare alla rete pubblica un certo numero di apparecchi telefonici mediante un centralino privato.

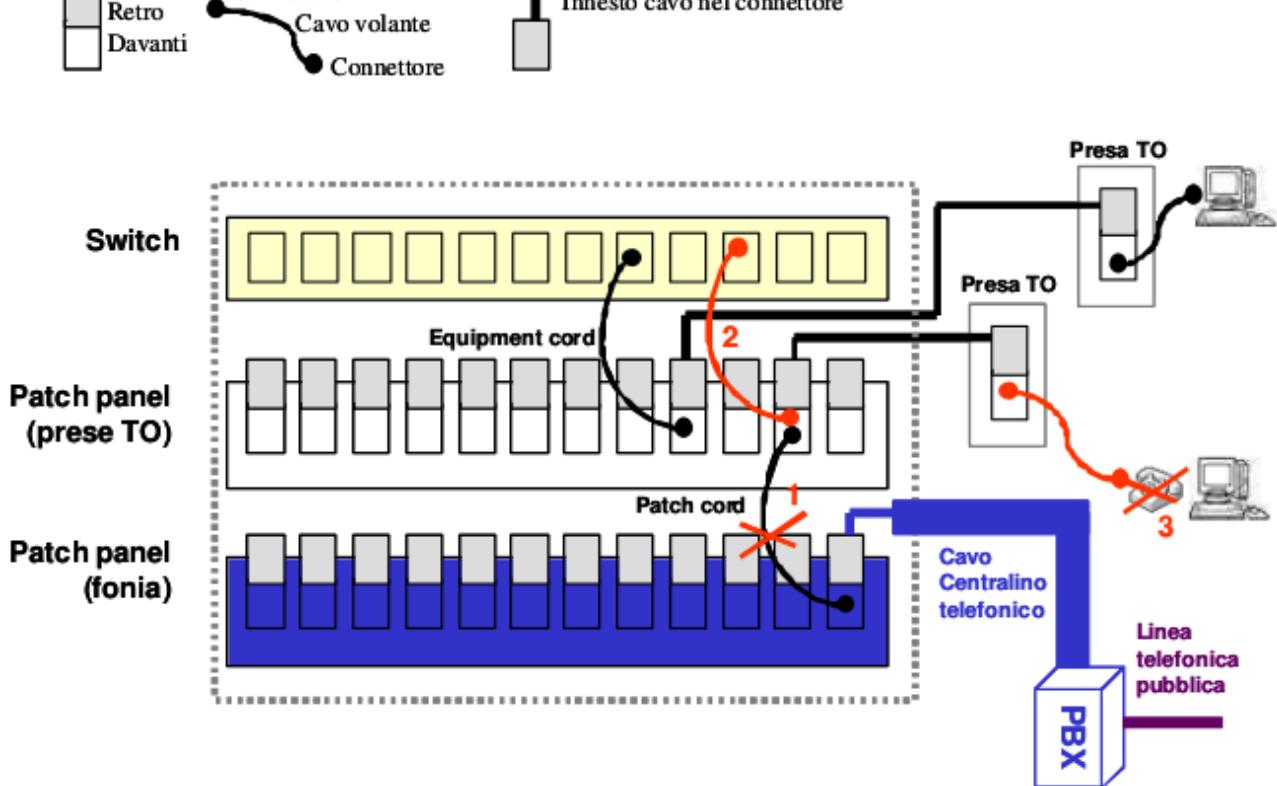
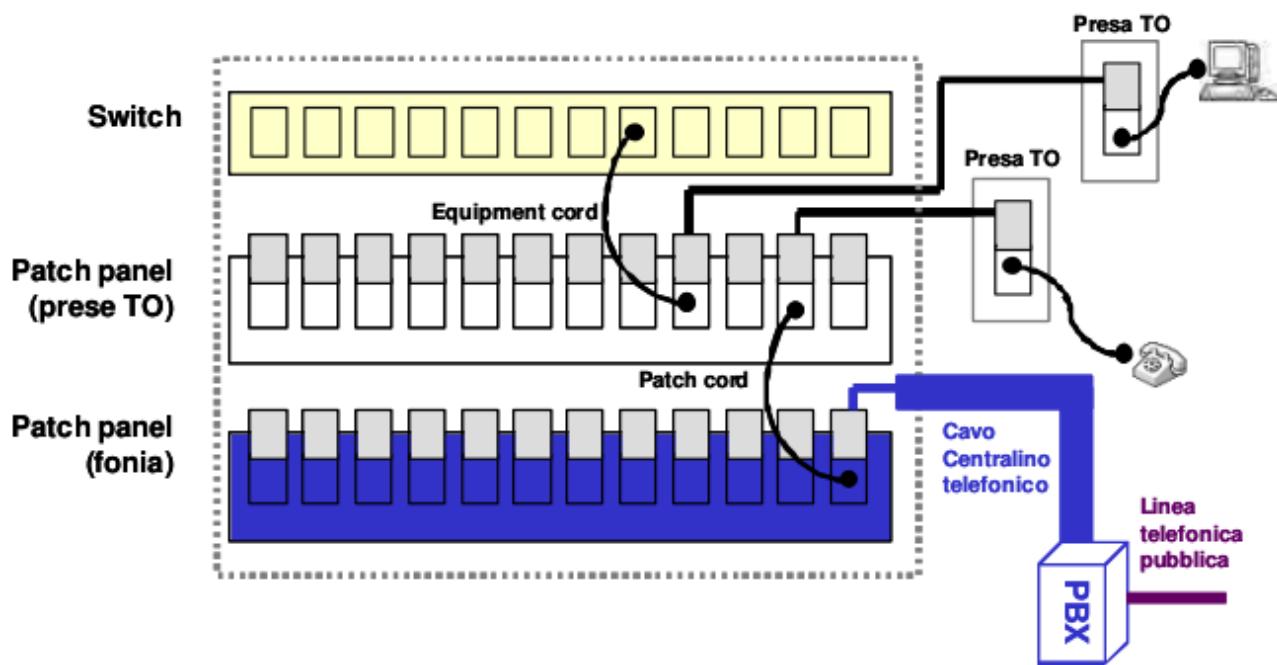
In un sistema di cablaggio per servizi dati e servizi fonia, il cablaggio orizzontale dal distributore di piano (**FD**) alle prese **TO** è unico, mentre il cablaggio di dorsale prevede due cablaggi separati, uno per i dati e uno per la fonia, e la connessione tra orizzontale e dorsale avviene con una interconnessione indiretta. Si utilizzano infatti uno o più patch panel per le prese **TO** e un patch panel per la fonia (RJ45 o sistema 110) detti di appoggio, in cui entra il doppino che può essere un cavo multicoppia⁹⁷ proveniente da un altro distributore (livello superiore) oppure direttamente dal **PABX**.

I patch panel e i cavi necessari sono:

- patch panel per le TO, cioè per il cablaggio delle WA degli utenti;
- patch panel di appoggio di fonia per le linee provenienti dal centralino telefonico, con un numero di connettori pari a metà del numero di TO;
- cavo EC (*Equipment Connector*) per i dati;
- cavo PC (*Permutation Cable*) per la fonia;
- cavo WAC per i dati;
- cavo WAC per la fonia.

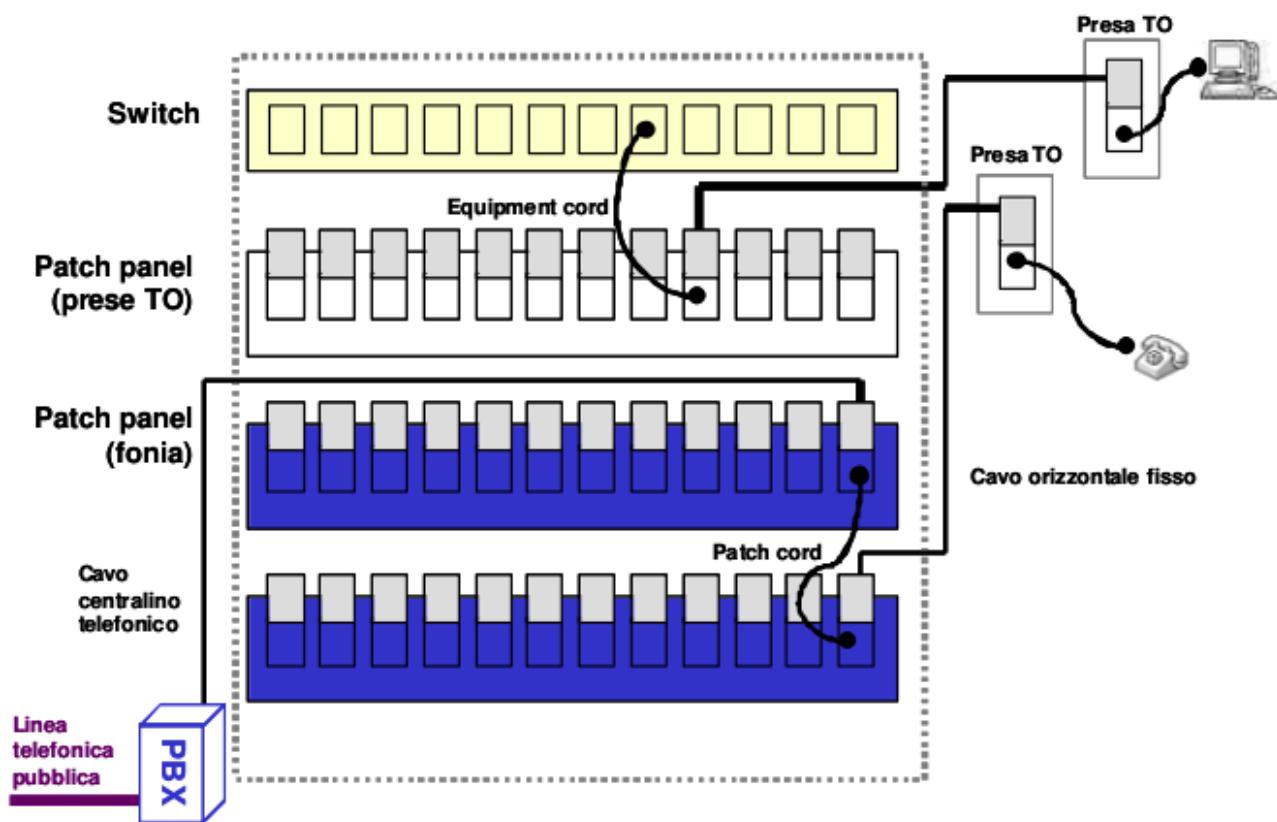
La flessibilità del sistema è data dall'utilizzo dei permutatori e dei cavi volanti, infatti l'assegnazione di una presa TO in precedenza assegnata ad un apparecchio di fonia si ottiene agendo sul distributore, spostando i cavi volanti (operazione 1 e 2 nelle figure sottostanti). Le figure seguenti illustrano l'esempio di un distributore di piano FD che prevede per ogni WA una presa per i dati e una presa per la fonia.

⁹⁷ Per la fonia vengono usati i pin 4 e 5.



In alternativa allo schema precedente basato su un permutatore di appoggio per la fonia si possono utilizzare due permutatori di appoggio con lo scopo di separare il permutatore dati da quello di fonia. In questo caso il permutatore dati collega metà

TO e il cablaggio è tutto su doppino a 4 coppie che permette una maggiore flessibilità del cablaggio stesso.



Caratteristica dei mezzi trasmissivi

Il mezzo trasmissivo è il canale fisico su cui veicolare i segnali e da cui dipendono le caratteristiche stesse dell'intero cablaggio strutturato.

La scelta del tipo di mezzo trasmissivo deve permettere di ottenere le prestazioni richieste, supportare gli standard attuali e futuri, garantire un'affidabilità prolungata nel tempo e garantire le dovute protezioni nell'ambiente di utilizzo.

I **tipi di mezzi** utilizzabili nelle normative sono il **rame** e la **fibra ottica**, in forma di cavo fisso (rigido o posato) o flessibile (anche detto volante).

Le normative stabiliscono le caratteristiche di un cavo certificato, cioè i requisiti minimi che il cavo deve soddisfare come limite superiore o inferiore di alcune misurazioni sui parametri fondamentali del cavo, come l'attenuazione, Next, ACR e altri.

Le **distanze ammesse** per le **dorsali variano** a seconda dei **mezzi trasmissivi** utilizzati e delle interconnessioni da effettuare, oltre che dalla **banda passante**, cioè la quantità di dati che può passare attraverso il cavo ogni secondo.

Si faccia riferimento alla [sezione 4.2](#) del corso CCNA: *Introduction to Networks* della CISCO™ per le caratteristiche dei cavi in rame e in fibra e dei relativi connettori. Nella sezione vengono toccate anche le caratteristiche del Wireless che deve comunque essere studiare, anche se non facente parte del cablaggio strutturato.

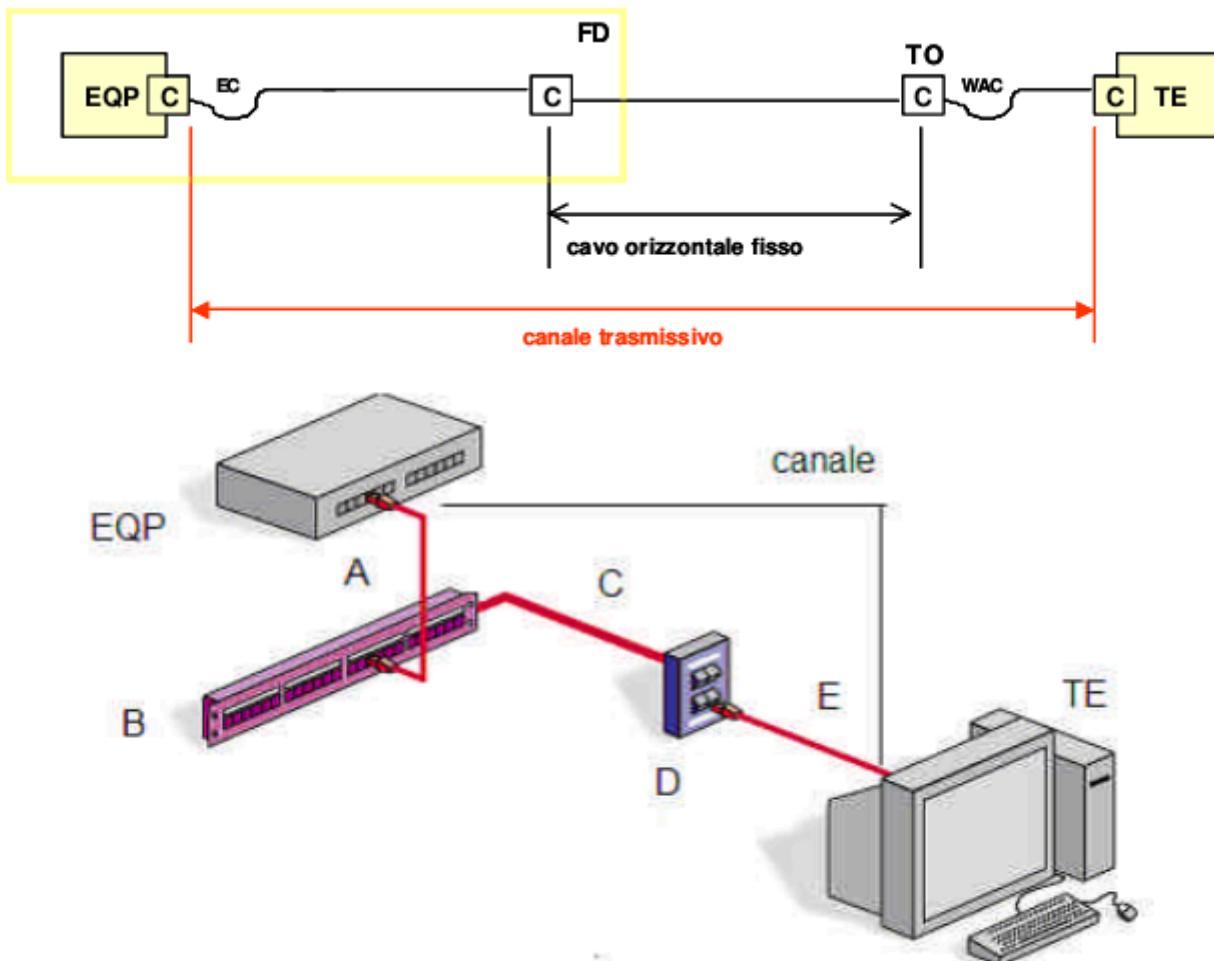
A questo [link](#) è possibile consultare uno schema riassuntivo delle diverse tipologie di cavo e delle relative caratteristiche e distanze.

Canale

Lo scopo del cablaggio è realizzare una infrastruttura passiva per creare dei percorsi di trasmissione tra una apparecchiatura attiva (**EQP**) e una apparecchiatura terminale (**TE**), come ad esempio tra uno switch (**EQP**) sul distributore di piano e un computer (**TE**) sull'area di lavoro dell'utente.

Nella **normativa EN** questo percorso si definisce come **canale** e questo gioca un ruolo fondamentale dato che l'applicazione trasportabile con il cablaggio dipende esclusivamente dalla prestazione del canale, cioè dalla capacità di trasportare il segnale senza significative perdite di capacità.

Il **canale**, individuato in rosso nella figura sottostante, è **composto** esclusivamente da **elementi passivi**, cioè dalle tratte passive del cavo e dalle connessioni alle estremità, senza considerare le connessioni delle apparecchiature.



C è il connettore – **EQP** è l'apparecchiatura di trasmissione – **TE** è l'apparecchiatura terminale
EC cavo apparecchiatura - **WAC** cavo apparecchiatura area lavoro
Equipment cable (A) – Patch panel (B) – Cavo fisso (C) – TO (D) - Work area cable (E)

In generale i canali sono realizzati utilizzando solo il cablaggio orizzontale, solo il cablaggio di dorsale di edificio, solo il cablaggio di dorsale di campus o una combinazione di tutti i precedenti.

Le **prestazioni del canale** non devono essere inferiori ai requisiti minimi delle prescrizioni previste dalla normativa e, in generale, tali prestazioni dipendono da diversi fattori:

- dalla lunghezza del cavo;
- dalle prestazioni dei componenti del canale che sono date dai parametri di attenuazione, Next, ecc;
- dal numero di connessioni, infatti più sono le connessioni, i connettori e i cavi flessibili utilizzati e peggiore è la prestazione del canale;
- dalla lunghezza complessiva di tutti i cavi flessibili, infatti maggiore è la lunghezza del cavo flessibile, minore è la prestazione del canale a causa di una maggiore attenuazione introdotta dei cavi flessibili;
- dalle tecniche di installazione utilizzate, fattore spesso trascurato.

I canali si distinguono a seconda del tipo di mezzo usato (rame e fibra) potendo anche creare canali misti, tipicamente fibra per le dorsali e rame per il cablaggio orizzontale.

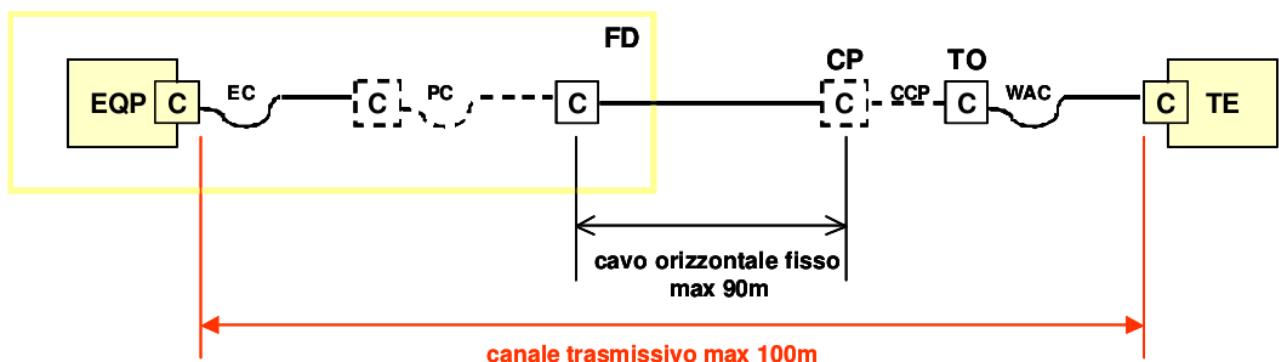
Nel caso di utilizzo di **canale misto** il collegamento tra i due canali avviene attraverso un convertitore *fibra ottica-cavo rame* costituito da un'apparecchiatura con porte optoelettroniche e porte rame, come ad esempio gli **switch fibra-rame** con porte in fibra native oppure inserite mediante moduli (GBIC o altro).



Un parametro fondamentale di un **canale** è la sua **distanza massima**⁹⁸:

- **canale orizzontale o distribuzione di piano: 100 metri;**
- **canale orizzontale + dorsale di campus + dorsale di edificio: 2000 metri.**

È da notare che non tutte le applicazioni sono supportate per le lunghezze massime indicate utilizzando un unico tipo di cavo, per esempio su un doppino non è possibile avere Fast Ethernet per 1000 metri.



C è il connettore – EQP è l'apparecchiatura di trasmissione – TE è l'apparecchiatura terminale
EC cavo apparecchiatura – PC cavo permutatore – WAC cavo apparecchiatura area lavoro
WA area di lavoro – CP consolidazione punto - CCP cavo CP - Tratteggio elemento opzionale

I modelli di canale orizzontale

Per il **canale orizzontale** la normativa EN stabilisce le seguenti restrizioni:

- la lunghezza fisica del canale non può superare i 100 metri;
- la lunghezza fisica del cavo orizzontale (L2) fisso non deve superare i 90 metri;
- la lunghezza del cavo PC (L1) non deve superare i 5 metri;
- se si utilizza un MUTO il cavo WAC non deve superare i 20 metri;

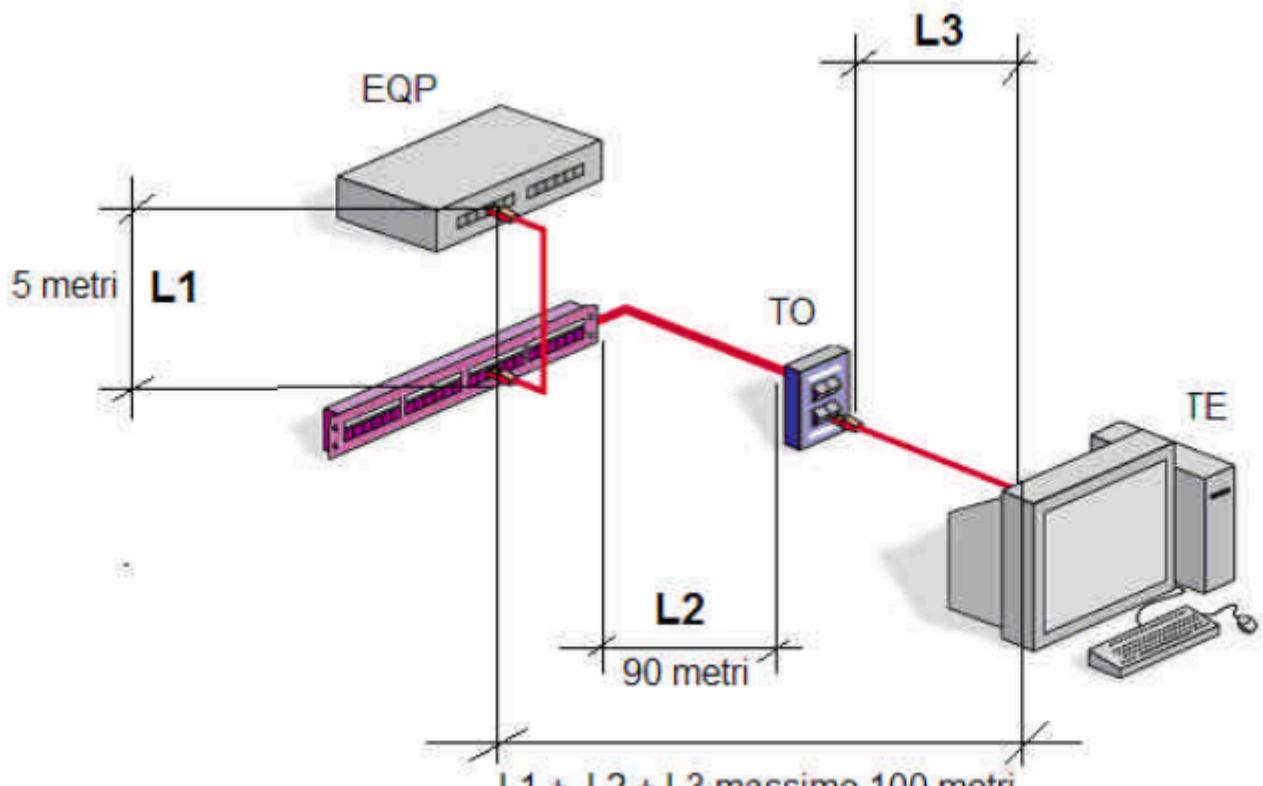
⁹⁸ Le lunghezze sono ottenute da equazioni per il canale sulla base di modelli di canale, considerando il numero massimo di connessioni, cioè il caso peggiore.

- se si utilizza un CP il cavo dal distributore di piano al CP, detto cavo CP, dovrebbe essere posto ad almeno 15 metri dal distributore di piano;
- la lunghezza massima dei cavi WAC (L3) ed EC è la differenza tra la massima lunghezza di canale (100 metri) e la lunghezza dei cavi PC.

Si ricava che la lunghezza massima del cavo orizzontale fisso dipende dalla lunghezza totale del cavo CP e dei vari cavi volanti da inserire nel canale, oltre che dalle caratteristiche dei cavi, e ne consegue l'importanza fondamentale della riduzione della lunghezza totale ($L_1 + L_2$) dei cavi flessibili.

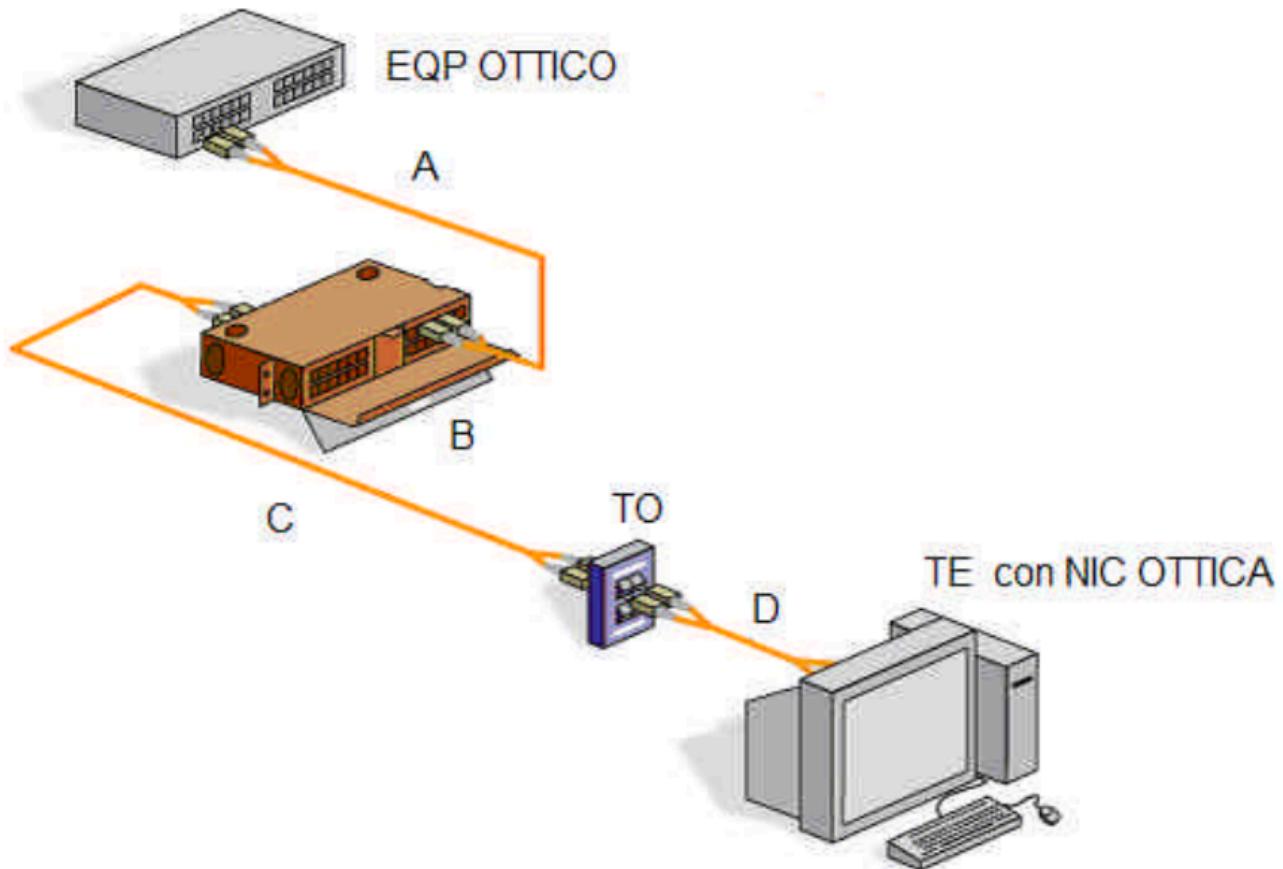
Se il cavo fisso è pari alla lunghezza massima allora la somma dei cavi volanti non deve superare i 10 metri, cioè **EC+PC+WAC = 10 metri** e si ricava anche che non tutte le prese utente possono essere servite all'interno di una zona.

È comunque opportuno per le prestazioni del canale che la lunghezza dei cavi flessibili, in particolare i cavi dell'area di lavoro, sia la minore possibile.



Lunghezza massima del canale orizzontale

Il canale orizzontale può essere su doppino o su fibra e le distanze da rispettare sono sempre le stesse.



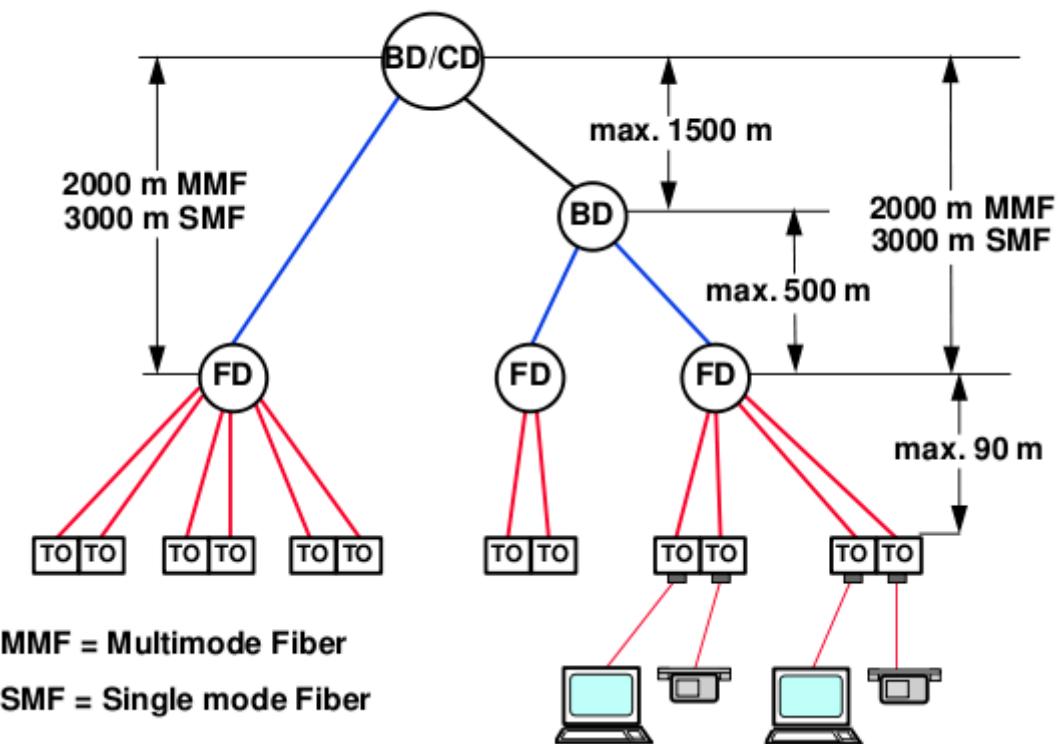
Bretella EC (A) – Cassetto (B) – Cavo fisso fibra (C) – Bretella WAC (D)

Canale orizzontale in fibra ottica

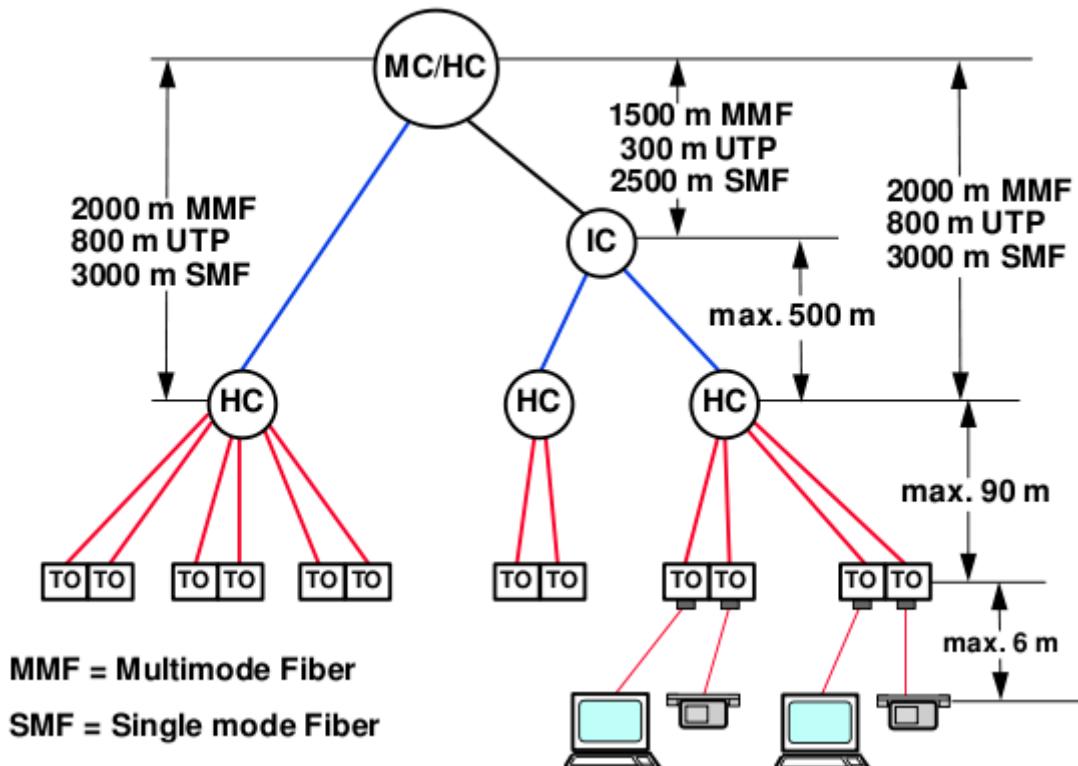
Il **canale di dorsale** è costituito dal sottosistema di dorsale, dai cavi di apparecchiature EC e cavi di permutazione PC.

La **lunghezza massima del cavo di dorsale dipende da molti fattori**, quali la lunghezza totale dei cavi volanti da inserire nel canale di dorsale, la categoria dei cavi, la classe del canale, il tipo di applicazione e il tipo di dorsale, e comunque **non deve superare i 2000 metri**.

Limiti di distanze ISO/IEC IS 11801



Limiti di distanze TIA/EIA 568A



Disposizione degli elementi funzionali

La normativa EN 50174-2 specifica la disposizione e l'alloggiamento degli elementi funzionali del cablaggio strutturato.

In linea generale gli **elementi** vengono **alloggiati** in contenitori e in posizioni che devono **tener conto** degli **aspetti ambientali** dell'edificio (fisico, climatico, elettromagnetico), permettendo anche una **semplice installazione** e futura **manutenzione** (misure, riparazioni, espansioni) entrambe in **condizioni di sicurezza**.

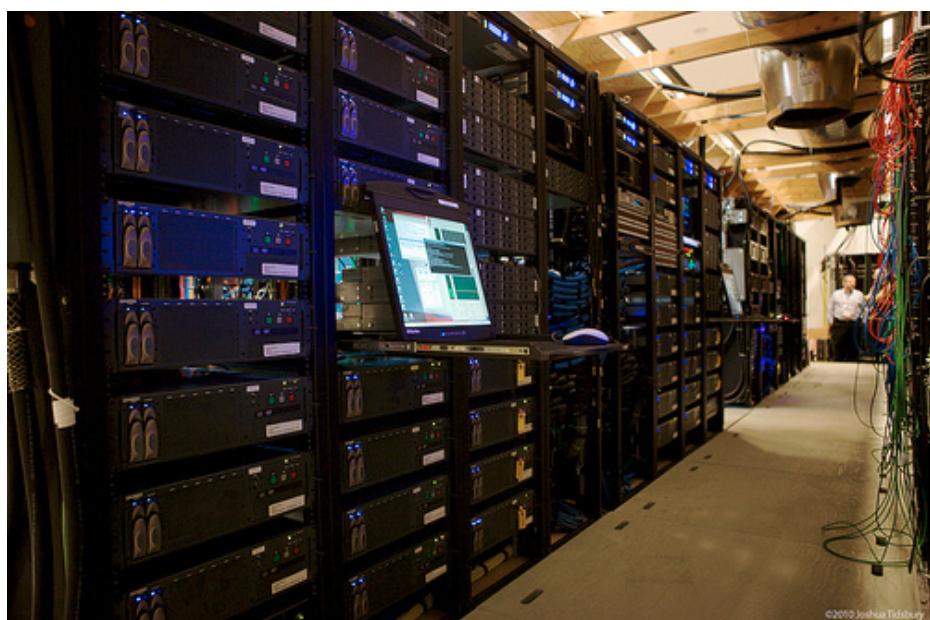
La normativa consiglia anche la **regola di buon senso**, e cioè di prevedere nel dimensionamento degli elementi funzionali, degli alloggiamenti e della loro disposizione, anche dell'espansione futura. Questo obiettivo può essere raggiunto pianificando, ad esempio, nelle canalizzazioni uno spazio libero per futuri cavi, nei contenitori ulteriore spazio per altri elementi per i distributori, e così via.

Distributori

I **distributori di edificio e di campus** dovrebbero essere disposti in apposite **sale apparati** (*Equipment Room - ER*).

La sala apparati **ER** è un particolare vano tecnico che dovrebbe essere appositamente progettato e realizzato per contenere un distributore di tipo **CD** o **BD**, gli apparati attivi (server, switch, router), i sistemi di alimentazione ausiliari, i sistemi antincendio e il controllo degli accessi, e comunque devono essere degli spazi che permettano un facile accesso per l'installazione di ulteriori cablaggi.

Nel caso di dimensioni di una certa rilevanza (media e grande azienda o un call center) la stanza deve anche essere acclimatata per mantenere un ambiente idoneo per le apparecchiature.



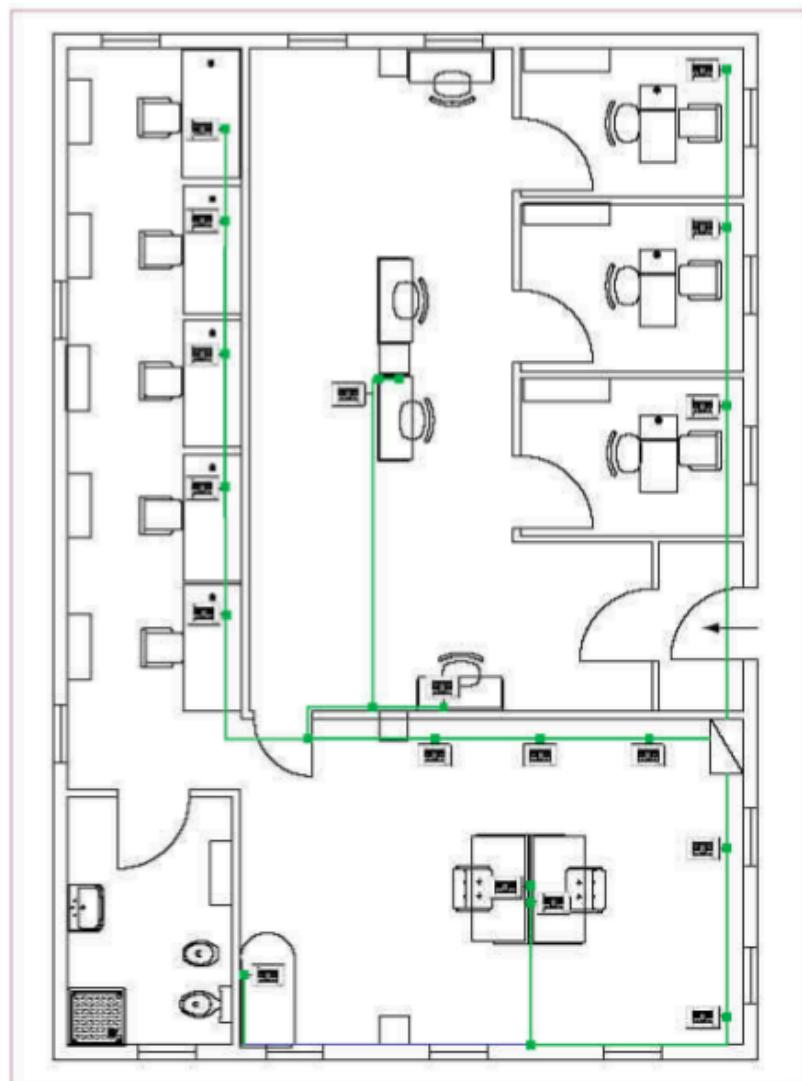
Prese utente

Le prese utente si trovano nelle aree di lavoro (**WA**), spazi destinati a una o più persone. La normativa generale non specifica dei limiti fisici all'estensione di queste aree, anche se alcuni regolamenti locali o nazionali possono stabilire delle limitazioni.

La disposizione delle prese utente (**TO**) dipende in primo luogo dalle prescrizioni di prestazione del canale che limitano la lunghezza dei cavi **WAC** tra la presa e l'apparato attivo, e comunque, in linea generale, le prese utente (**TO**) si dispongono con l'obiettivo di minimizzare la lunghezza dei cavi dell'area di lavoro.

Inoltre il posizionamento delle **TO** dipende dalla distribuzione delle aree di lavoro all'interno del piano o dei piani, cosa che spesso dipende dall'occupazione proposta dei locali e, in alcuni casi può dipendere da normative locali che specificano lo spazio minimo per ogni persona.

In mancanza di una proposta si può assumere come valore di riferimento quello già proposto di 10m² per area di lavoro (**WA**).



Layout cablaggio di una agenzia bancaria

Cavi

I cavi si posizionano nell'edificio considerando le limitazioni sulla lunghezza dei cavi stessi, quindi cercando di limitare la loro lunghezza per raggiungere i diversi elementi funzionali del cablaggio.

Il posizionamento deve tener conto di alcune zone che devono essere escluse, come il pozzo dell'ascensore e gli spazi per i parafulmini, evitando sorgenti di calore, di umidità o di vibrazione che aumentano il rischio di danneggiamenti del cavo o diminuiscono la prestazione del cavo stesso. Infine i cavi non si dovrebbero posizionare affiancati alle linee di potenza (energia elettrica) e quando questo non sia possibile devono essere collocati in alloggiamenti che permettono una separazione.



Canaline a parete con netta separazione tra energia e cablaggio strutturato

La normativa prevede per la distribuzione orizzontale di un distributore di piano (**FD**) la suddivisione dell'area del cablaggio orizzontale in zone adiacenti, in funzione dei limiti di distanze previste (100 metri per il canale e 90 metri di lunghezza massima del cavo fisso).

Ogni zona dovrebbe essere servita da un fascio di cavi unico (cavo multicoppia) e servito da un unico alloggiamento nel distributore allo scopo di permettere una più facile installazione e manutenzione, oltre alla riduzione della diafonia supplementare causata da cavi che scorrono in parallelo.

La progettazione del cablaggio strutturato

La progettazione del cablaggio strutturato dipende da molteplici fattori, quali:

- la disposizione topologica dello spazio da cablare: una stanza, un ufficio su un piano con più stanze, un edificio, un campus;
- dalla strategia dell'utilizzatore che può avere come obiettivo la riduzione dei costi o altre motivazioni;
- dall'applicazione di rete che la struttura di cablaggio deve trasportare, la quale richiede una determinata larghezza di banda, a sua volta dipendente dalle risorse hardware e software che si utilizzano (office automation, database, CAD, video digitale, ecc.);
- dall'ambiente, con la presenza o meno di problemi di **EMC** (*Electromagnetic compatibility*);
- dall'espansione futura della struttura, variazioni di impiego e/o potenziamento delle utenze, aumento della banda e innovazione tecnologica;
- dalle limitazioni di accesso agli armadi e al cablaggio al personale non autorizzato, conformemente alle indicazioni dell'utente finale.

Partendo dal presupposto che il concetto stesso di cablaggio strutturato prevede una predisposizione nelle opere edili, la progettazione della rete dovrà integrarsi con l'edificio, come qualsiasi altro impianto (elettrico, idraulico, ecc.), e nel caso di nuova costruzione, questo lavoro risulta indubbiamente semplificato.

Il progettista dell'impianto di cablaggio strutturato deve effettuare la scelta migliore, considerando i criteri di progettazione, all'interno di un dominio di scelta che agli estremi può prevedere un cablaggio tutto in rame (doppino) o un cablaggio tutto in fibra.

Spesso la soluzione scelta è quella ibrida:

- dorsali in fibra;
- server con necessità di banda su fibra;
- cablaggio orizzontale in doppino.

La progettazione è una procedura in più passi che in generale deve prevedere:

1. la scelta dello standard di riferimento;
2. la scelta del modello di cablaggio;
3. la definizione del cablaggio orizzontale di piano e di quello verticale di edificio e di campus rispettando tutti i limiti sulle distanze e la planimetria fornita;
4. la scelta dei componenti e dei cavi in funzione delle prestazioni richieste e delle condizioni ambientali;
5. la scelta dei sistemi di contenimento dei cavi e degli altri elementi;
6. la definizione del sistema di fonia;
7. un adeguato sistema di identificazione dei dispositivi tramite etichettatura e colorazione;
8. la predisposizione del documento di progetto.

In ogni caso, **in mancanza di specifiche di progetto** ci si attenga alle seguenti disposizioni minime:

1. un armadio ogni 1000m²;
2. al massimo 100 WA per ogni armadio;
3. una WA ogni 10m²;
4. posare 2 cavi per ogni WA (dati e fonia).

Assegnazione degli indirizzi IP

Pur non essendo una fase del cablaggio strutturato che rimane confinato al solo livello fisico, nelle reti che si andranno a progettare sarà comunque richiesta l'assegnazione di un indirizzo IP ad ogni dispositivo attivo, e quindi si dovrà predisporre un adeguato piano di indirizzamento, eventualmente utilizzando anche le VLAN.

Gli indirizzi assegnati potranno essere visibili solo all'interno della rete con un eventuale indirizzo esposto all'esterno, magari implementando un servizio di NAT o PAT.

Ovviamente dovranno essere previsti tutti i sistemi di sicurezza necessari per garantire un servizio continuativo e mirato ai soli utenti autorizzati (DMZ, firewall, VPN, sistemi di autenticazione, sistemi di cifratura).

Esempio di cablaggio strutturato

Prendiamo in esame il caso di una scuola di medie dimensioni in cui si voglia realizzare un cablaggio strutturato che raggiunga ogni laboratorio, ufficio e classe.

L'edificio che si vuole cablare è costituito da tre piani:

- al primo piano sono presenti gli uffici amministrativi costituiti da 3 stanze, la sala professori, la biblioteca e l'ufficio di presidenza;
- al secondo piano sono presenti 4 laboratori e 5 aule;
- al terzo piano sono presenti 9 aule.

Nelle aule dovrà essere presente una presa a muro con due connettori RJ45. Si potrà così utilizzare la presa di rete con un portatile o con un carrello dotato di PC e apparecchiature multimediali con la funzione di stazione mobile per le lezioni.

Si prevede la connessione a Internet da parte di ogni partizione di lavoro.

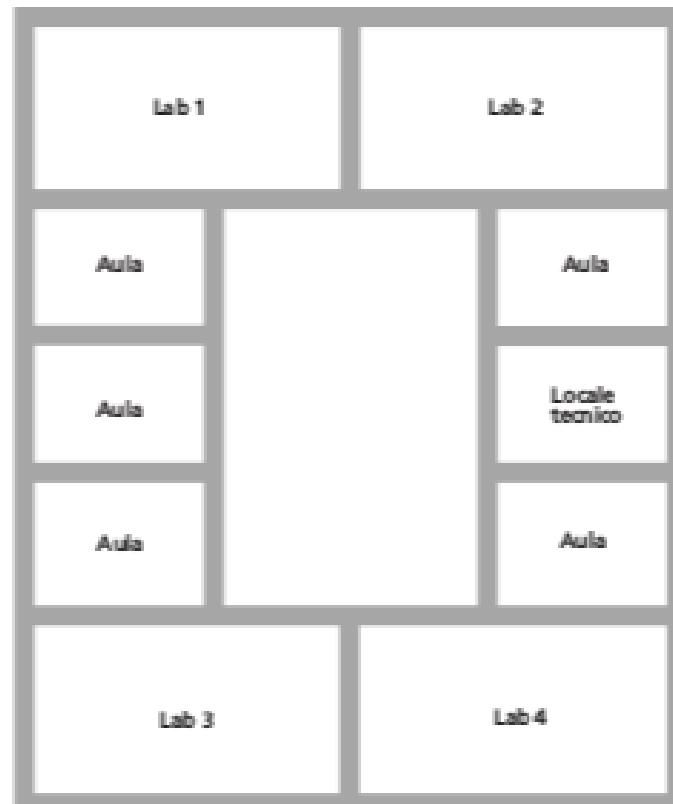
La planimetria dei tre piani è illustrata nelle figure seguenti.

La soluzione proposta è solo una delle possibili, e si lascia come esercizio il trovarne di alternative riorganizzando la progettazione del cablaggio strutturato o sviluppando ulteriormente la proposta utilizzando tecnologie alternative e inserendo ulteriori considerazioni relative soprattutto alla sicurezza complessiva del sistema.

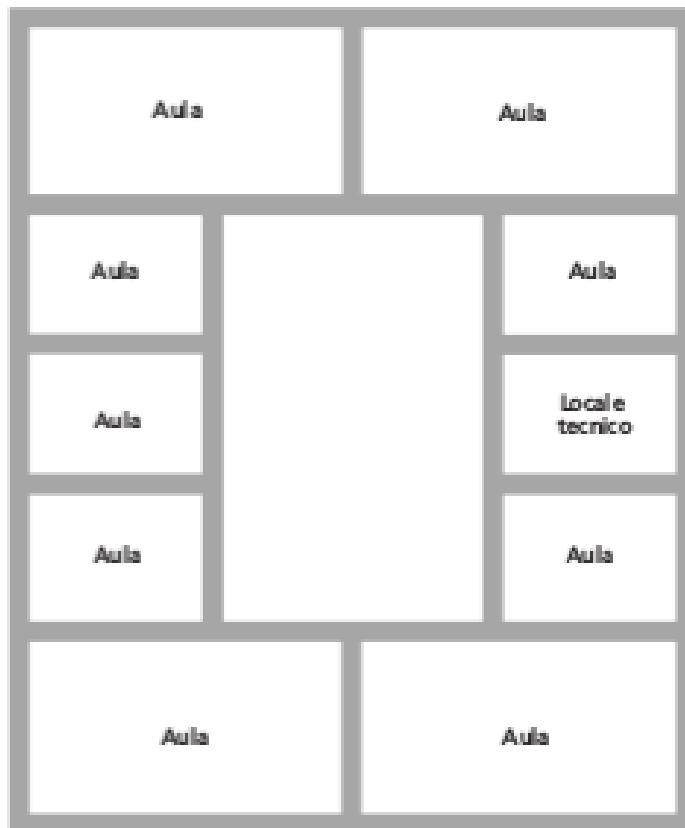
Primo piano



Secondo piano



Terzo piano

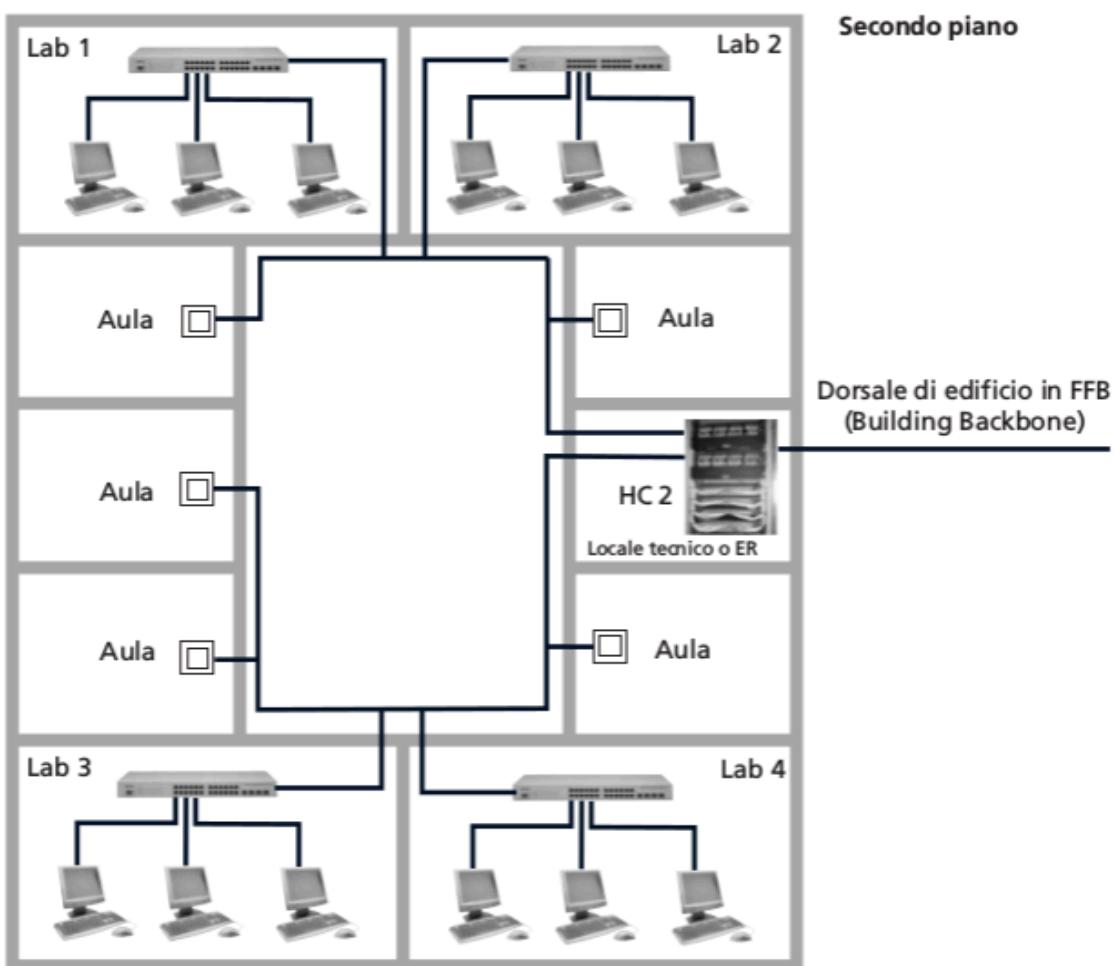
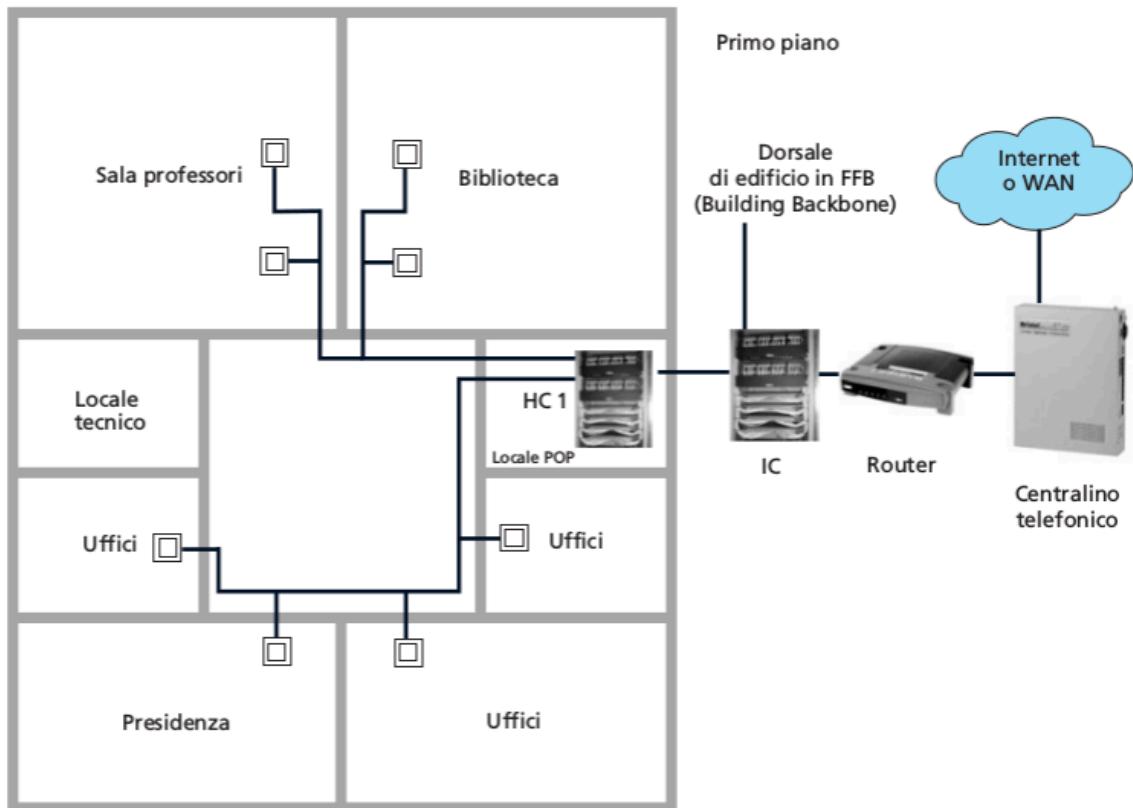


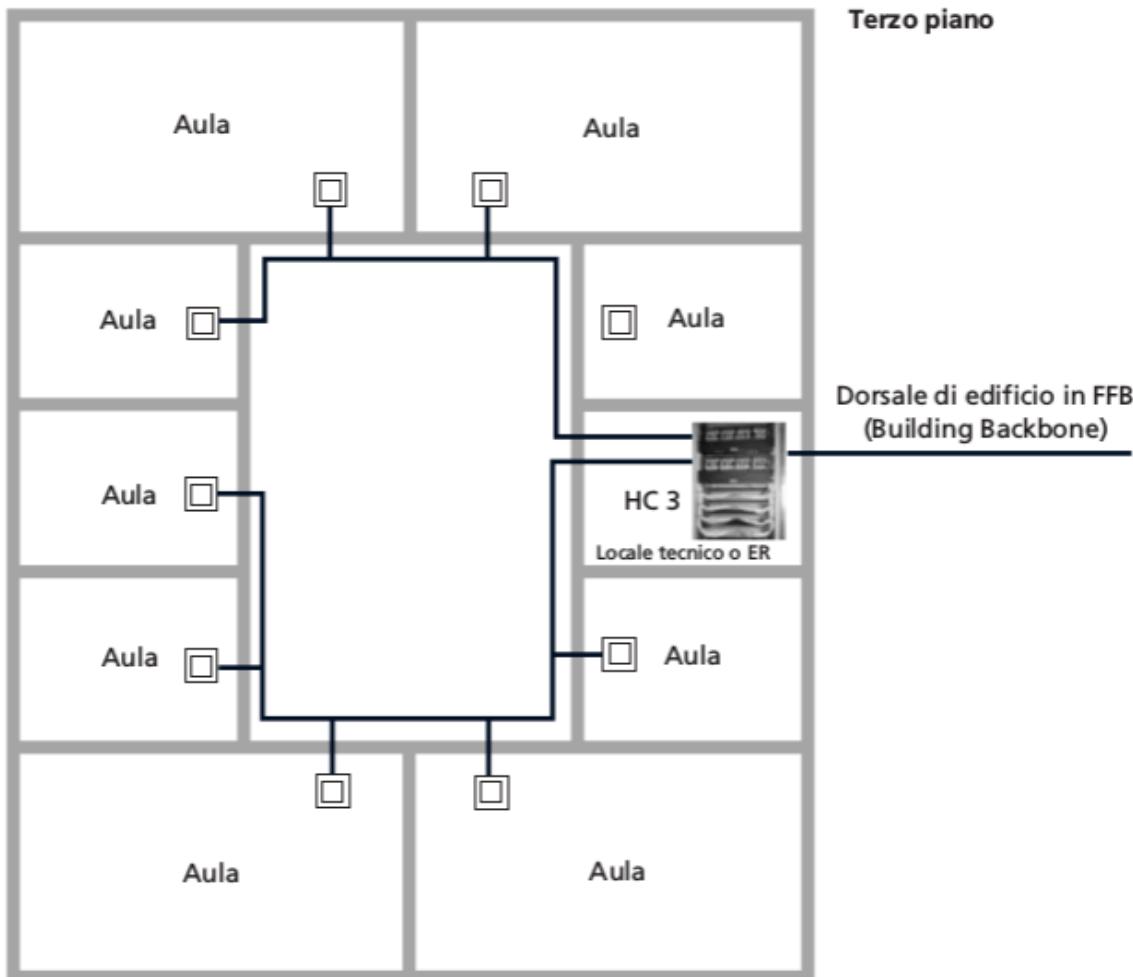
Operazioni preliminari, scelte tecniche e posizionamento degli apparati

In base a un sopralluogo e all'analisi della piantina della scuola si è deciso:

- il numero e le posizioni delle prese a muro in base alla funzione svolta dai singoli locali;
- il percorso dei cavi (e quindi delle canaline) e la loro lunghezza;
- la posizione dove installare gli armadi di permutazione.

Le piantine di seguito illustrate mostrano la dislocazione di tali elementi.





I canali

Per i canali si utilizzerà:

- fibra ottica multimodale per il cablaggio verticale;
- cavo UTP CAT6 per il cablaggio orizzontale dagli HC (FD) alle WA.

Si lascia come esercizio la scelta delle velocità dei cavi in base ai costi odierni.

I mezzi trasmissivi

Si useranno le seguenti tipologie di cavi:

- fibre ottiche multimodali 62.5/125 µm;
- cavi UTP a 4 coppie.

Elementi del sistema di cablaggio

Ogni piano avrà un armadio di permutazione che chiameremo HC1, HC2, HC3.

Al piano terra nel locale identificato come POP inseriremo il centro stella di edificio IC (BD).

Per le stazioni di lavoro presenti nei laboratori potrà essere conveniente avere a disposizione degli switch locali onde evitare di portare decine di cavi verso l'armadio di concentrazione. In questo caso sarà sufficiente avere una presa di rete nel laboratorio a cui collegare la porta di uplink dello switch. Un cablaggio strutturato permetterà di predisporre tutte le prese di rete della scuola anche senza che si debbano attivare tutte fin da subito.

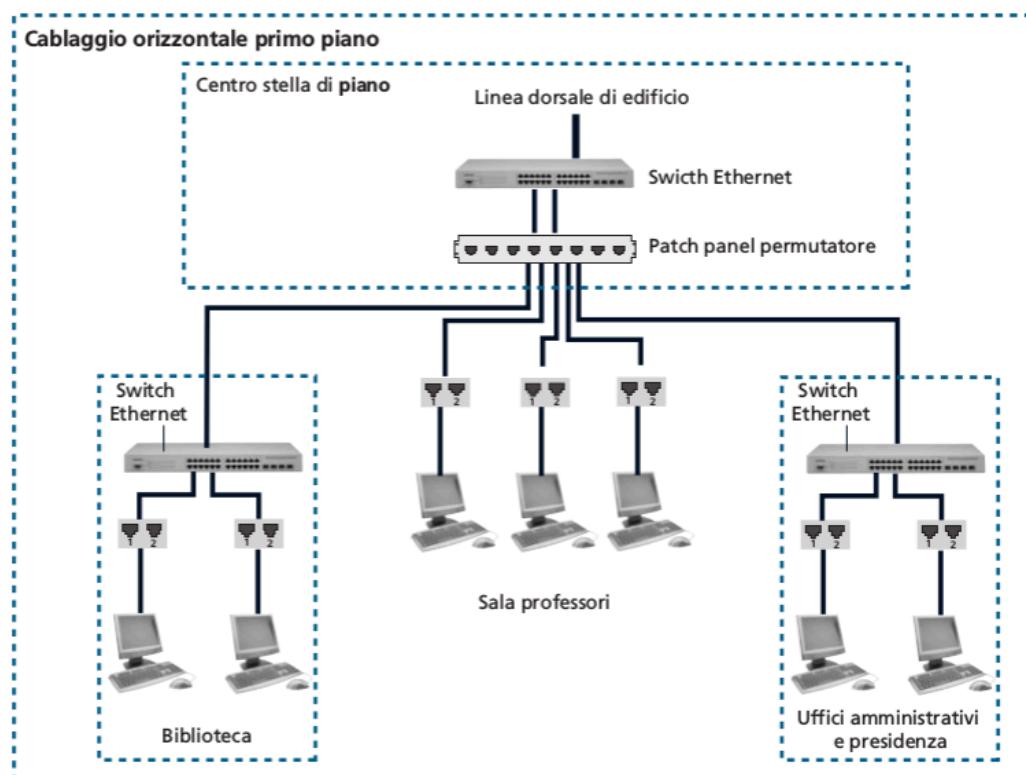
Nelle aule si potrà utilizzare la presa di rete con un portatile o con un carrello dotato di PC e apparecchiature multimediali con la funzione di stazione mobile per le lezioni.

Si utilizzeranno i seguenti componenti attivi:

- un UTM⁹⁹ (*Unified Threat Management*) che svolge sia la funzione di router, sia la funzione di firewall da posizionare nel locale POP (la tipologia di UTM dipenderà dal budget disponibile);
- uno switch Gigabit per ogni piano, con 16 porte Ethernet e connessione in fibra multimodale 62,5/125;
- uno switch Gigabit Ethernet con 24 porte per ogni laboratorio.

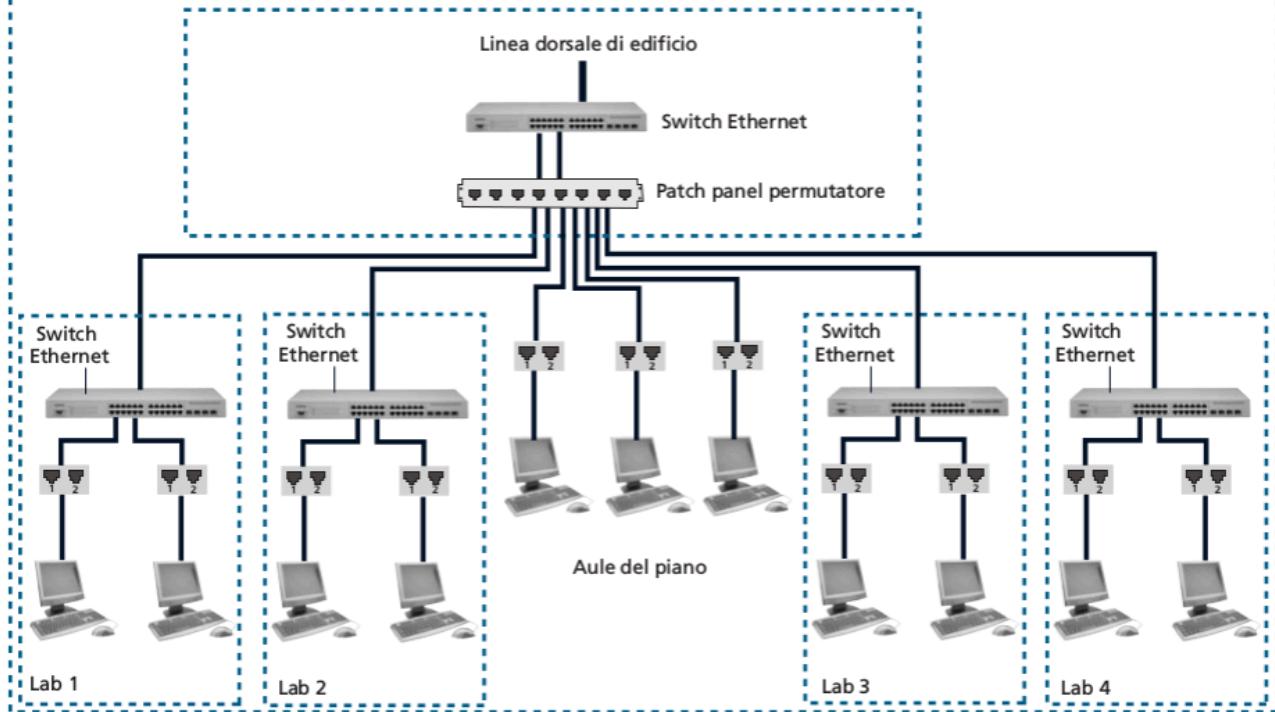
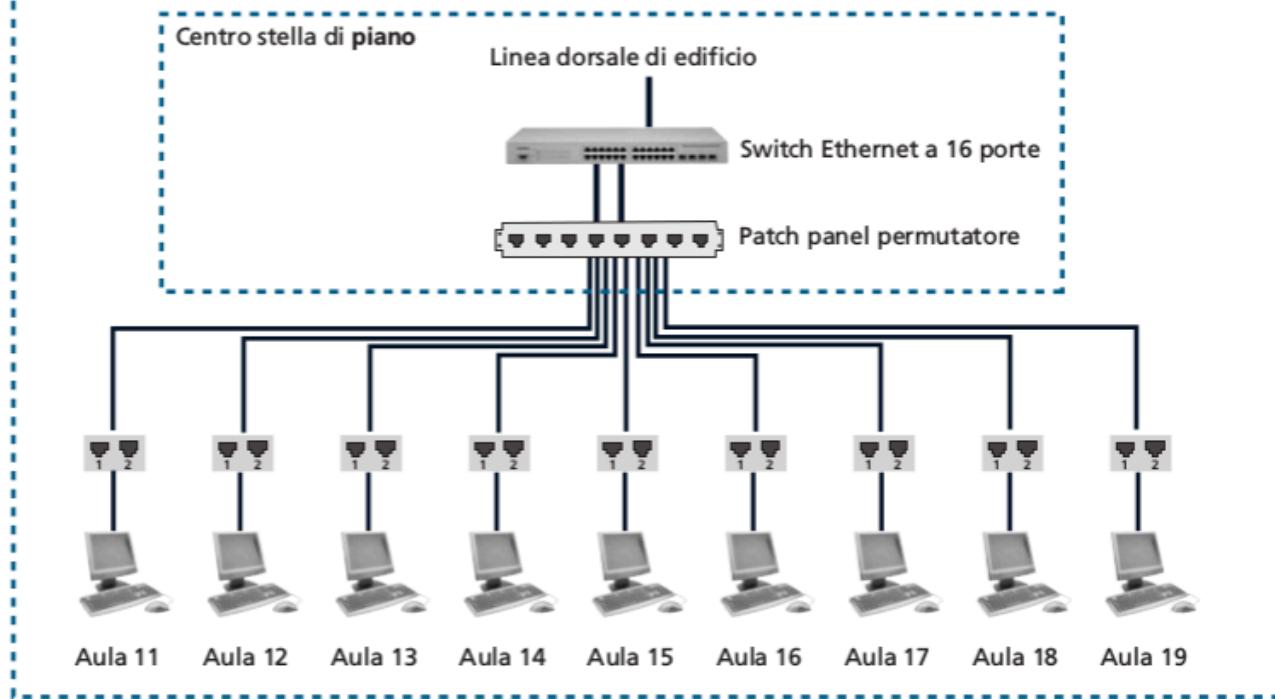
Il cablaggio orizzontale

Vediamo i cablaggi orizzontali relativi ai tre piani.



⁹⁹ Si vedano i seguenti link per comprendere quali funzioni sono offerte da un UTM:

- [Unified Threat Management \(UTM\) della Fortinet](#)
- [FortiOS™ Handbook - Advanced Routing](#)

Cablaggio orizzontale terzo piano**Cablaggio orizzontale terzo piano**

Assegnare gli indirizzi IP

Si potrebbero assegnare indirizzi di classe B o C, poiché il numero totale di computer è inferiore a 254. Sceglieremo comunque di assegnare indirizzi IP di classe A del tipo 10.0.0.0 (privati) perché in questo modo si potrà suddividere la rete in modo gerarchico, attribuendo uno specifico significato a ogni byte, ad esempio:

- il primo byte indica la rete di classe A e vale 10;
- al secondo byte è attribuito il numero del piano: 1 per il primo piano, 2 per il secondo, 3 per il terzo (quindi 3 sottoreti, ma sarebbe possibile creare 254);
- al terzo byte il numero di laboratorio o di aula, ad esempio:
 - ai laboratori saranno assegnati numeri a partire da 1, e quindi 1 per il Lab 1, 2 per il Lab 2, 3 per i Lab 3, 4 per il Lab 4;
 - alle aule saranno assegnati numeri sopra il 10, e quindi 11 per la prima aula 12 per la seconda aula 13 per la terza ecc;
 - al quarto byte il numero progressivo della WA (computer o stampante o altro dispositivo di rete) presente nell'aula o nel laboratorio, partendo da 1 fino a 254.

Seguendo queste indicazioni l'indirizzo 10.2.3.41 individuerà un dispositivo che si trova al secondo piano, terzo laboratorio (Lab 3), caratterizzato dal numero progressivo 41.

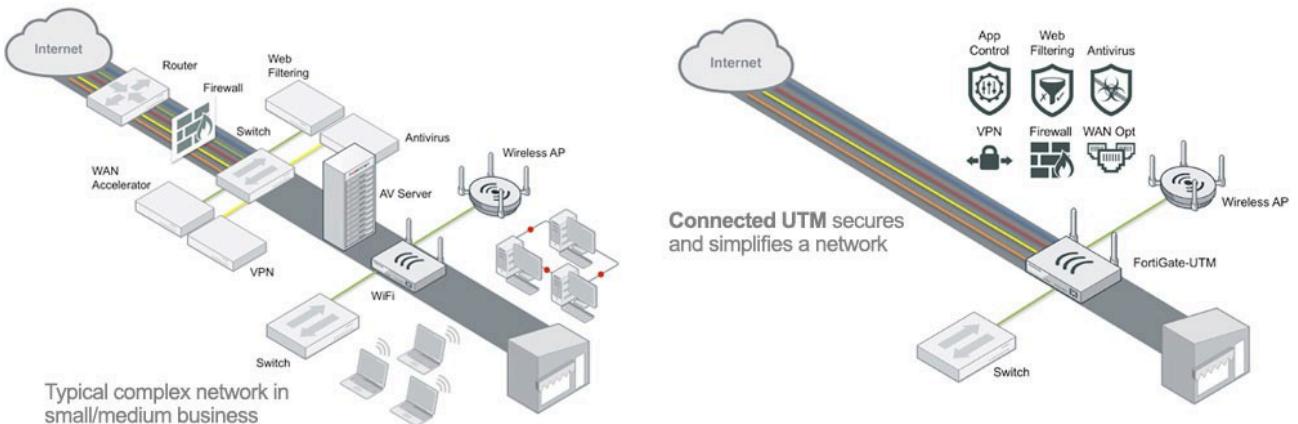
Si lascia come esercizio la predisposizione del piano di indirizzamento supponendo che in ogni laboratorio saranno installabili al massimo 24 PC.

Programmazione degli apparati attivi

Per il collegamento a Internet si sceglierà di utilizzare un UTM la cui funzione di routing implementerà un protocollo NAT/PAT che permetta a ogni nodo della rete di accedere a Internet mediante un solo indirizzo pubblico assegnato all'interfaccia pubblica dello UTM stesso.

L'UTM svolge primariamente la funzione di firewall che può essere impostato su diversi criteri, come ad esempio sugli indirizzi IP, sul protocollo, sul nome host, ecc.

Impostazioni meno recenti utilizzavano le ACL (*Access Control List*) le quali permettono di specificare, tramite operatori logici, filtri inclusivi in cui venivano specificati solo gli indirizzi permessi, e filtri esclusivi in cui venivano specificati solo gli indirizzi non consentiti. Questa tecnica però è sempre meno utilizzata in quanto le ACL non permettono di gestire automaticamente le risposte alle sessioni aperte.



Collegamenti wireless

Le reti wireless (WLAN) sono reti di computer che utilizzano onde radio (RF) o raggi infrarossi (IR) per realizzare il collegamento tra le stazioni di lavoro.

Il collegamento tramite onde radio è il più utilizzato, avendo un raggio di azione maggiore, oltre a una banda e una copertura più ampia.

La velocità di trasmissione delle reti wireless è in funzione degli standard adottati e fra quelli attualmente più utilizzati c'è lo standard [IEEE 802.11n](#) che opera sia nella banda di frequenza dei 2.4 GHz, sia in quella dei 5 GHz e la velocità reale dovrebbe essere di 300 Mbit/sec.

Le reti wireless offrono molti vantaggi e vengono utilizzate quando si vuole avere libertà di movimento per la propria stazione di lavoro. Inoltre permette di superare eventuali vincoli fisici o di tutela artistica di un edificio non essendoci la necessità di far passare tracce e cavi attraverso i muri o i soffitti. Permette anche di limitare i costi del cablaggio e di realizzare reti temporanee o urgenti in poco tempo.

Il mondo **wireless** costituisce quindi un'**alternativa al cablaggio strutturato** specialmente in tutti quei casi in cui non è richiesta una altissima affidabilità e una banda ampia quanto quella di un sistema wired, anche se risulta più affidabile avere un sistema misto che utilizzi il wireless in determinati ambienti, ma si faccia affidamento al cablaggio strutturato almeno per le connessioni di piano.

Per allestire una rete wireless è necessario disporre di un apparato centrale detto **Access Point**, e di **schede di rete wireless** da inserire sulle singole stazioni di lavoro.

L'**Access Point** è un trasmettitore radio, operante alle frequenze di 2.4 GHz e di 5 GHz, in grado comunicare con tutti gli adattatori di rete che si trovano nella sua zona di copertura. Viene solitamente collegato alla rete locale, con una porta RJ45, per fare da ponte tra la rete wireless e la rete cablata. La potenza di trasmissione è limitata, per legge, a 10 mvolt (*microvolt*).

La **scheda wireless** può essere una scheda interna da inserire sul bus PCI del computer oppure un dispositivo USB. Le schede wireless possono comunicare con l'Access Point (reti wireless strutturate) o direttamente tra loro, se le distanze e il numero dei nodi sono limitati (reti wireless ad hoc).



Le installazioni odierne offrono una copertura radio che può arrivare anche a 300 metri in orizzontale, mentre in verticale riescono a superare un solaio coprendo quindi il piano inferiore e il piano superiore rispetto a quello in cui è posto l'Access Point. Questi valori sono ovviamente da considerarsi solo indicativi in quanto dipendono dagli

ostacoli architettonici degli edifici. Infatti lo spessore dei muri e la consistenza dei solai incidono fortemente sulla ricezione del segnale radio, ed è quindi sempre opportuno fare un sopralluogo preliminare con un sistema wireless funzionante per capire se questo soddisfa le esigenze.

Per superare gli ostacoli architettonici ed offrire una buona copertura indipendentemente dalla dimensione dell'edificio è possibile utilizzare più Access Point per creare differenti zone di copertura che funzionano come le "celle" dei telefonini, garantendo anche il lavoro ad un computer che viene spostato da una area all'altra.

È inoltre possibile realizzare ponti radio per estendere la rete locale su più edifici disponendo di due Access Point che vengono collegati ad antenne direttive (parabole o yagi), allestendo ponti di trasmissione con una portata fino a 2-3 chilometri, purché esista la visibilità ottica delle due antenne.

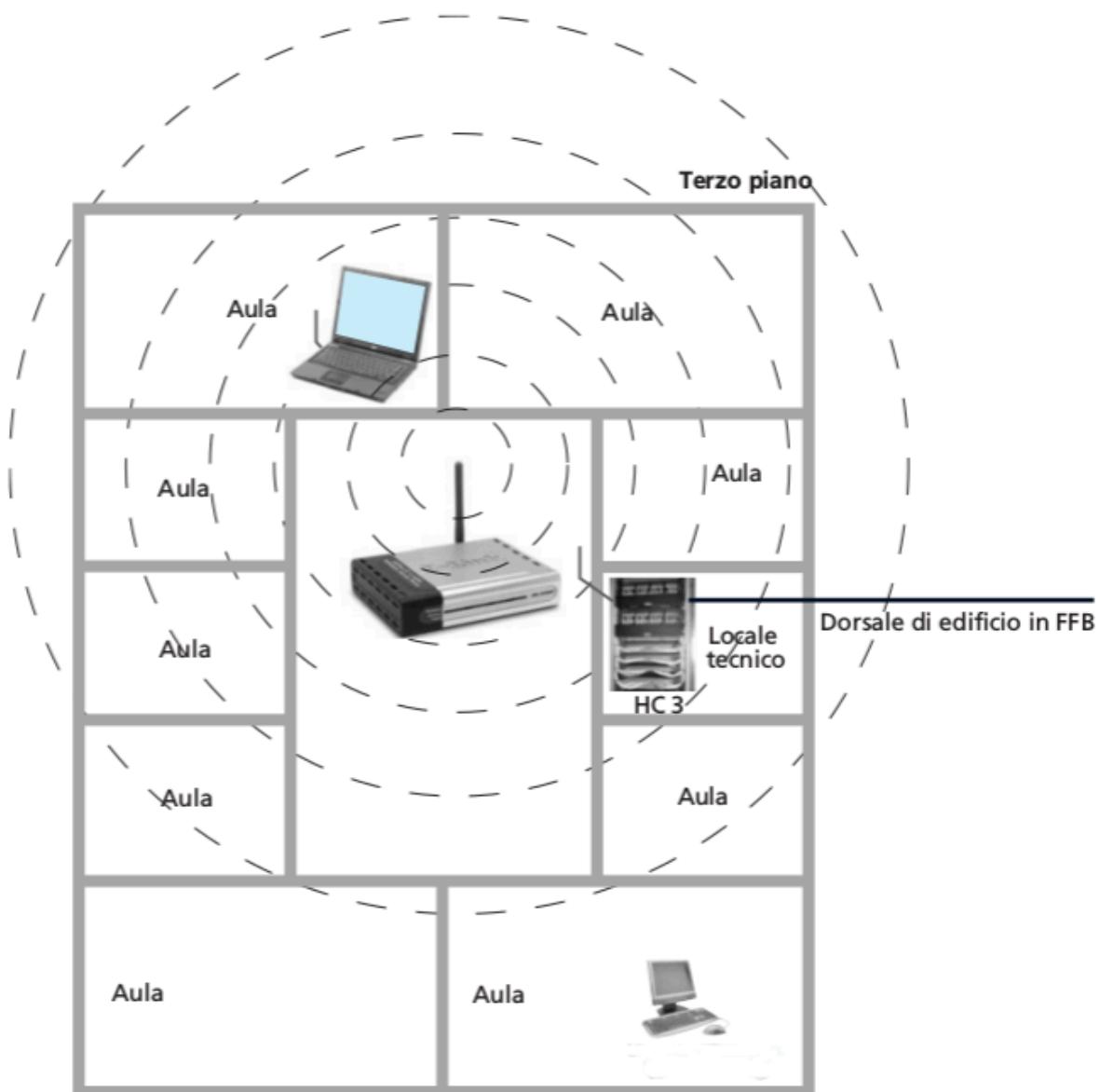
Un esempio di cablaggio con postazioni wireless

L'esempio precedente potrebbe essere modificato introducendo dei collegamenti wireless.

La situazione dei piani può rimanere invariata, ma si può supporre che al terzo piano si faccia uso di una soluzione wireless anziché di cablaggio strutturato.

Si dovranno quindi aggiungere i seguenti dispositivi che sostituiranno il cablaggio orizzontale:

- un Access Point che sarà posizionato in modo tale da coprire tutto il piano (le distanze coperte rientrano nel raggio di azione dell'apparato);
- tante schede di rete quanti sono i computer destinati a essere trasportati nelle aule (si ipotizzino quattro portatili);
- un permutatore di piano al quale collegare l'access point affinché possa collegarsi con il resto della rete della scuola.



Bibliografia e sitografia

S. Anelli, P. Macchi, G. Angiani, G. Zicchieri, *Gateway - Sistemi e reti*, vol. 3, ed. Petrini, 2016

F. Pecoraro, *Appunti di Reti di Calcolatori*,
<http://www.giordanicaserta.it/frapec/sistemi/Commutazione.html>, ultimo accesso maggio 2017

Computer Networking. Cap. 3. Autori: James F. Kurose (University of Massachusetts, Amherst) Keith W. Ross (Polytechnic Institute of NYU)

CISCO, CCNA R&S: *Introduction to Networks*, <http://netacad.com>, ultimo accesso agosto 2017

S. Empson, CCNA - *Guida rapida ai comandi*, 2a ed., Cisco Press, Pearson Informatica, marzo 2009

S. Danesino, canale YouTube [sdanesino](#) e [Sophia Danesino](#)

D. Evans, [Applied Cryptography](#), [Udacity](#), ultimo accesso settembre 2017

B. Schneier, *Applied Cryptography*,
<https://mrajacse.files.wordpress.com/2012/01/applied-cryptography-2nd-ed-b-schneier.pdf>, ultimo accesso settembre 2017

NetPing, <http://www.netpingdevice.com/Blog>, ultima visita settembre 2017

Dans Courses, canale YouTube [danscourses](#), sito Web [danscourses.com](#), ultima visita novembre 2017

[Netsetup.it](#), ultimo accesso ottobre 2017

[Udacity](#), [Applied Cryptography](#), ultimo accesso novembre 2017

[Learn Cryptography](#), [Frequency Analysis](#), ultimo accesso novembre 2017

[Khan Academy YouTube channel](#), [The Caesar cipher | Journey into cryptography | Computer Science | Khan Academy](#), ultimo accesso novembre 2017

[CS50](#), [Vigenère Cipher](#), ultimo accesso novembre 2017

[Wikipedia](#), [Transposition cipher](#), ultimo accesso novembre 2017

[internet-class](#), [Security](#)

[Math Explorers' Club](#), [Number Theory and Cryptography - Primes, Modular Arithmetic, and Public Key Cryptography](#), [Cornell University - Department of Mathematics](#), April 15, 2004

P. Corn, A. Ellinor, E. The Head, M. Jain, [Diffie-Hellman](#), [Brilliant](#), ultima accesso dicembre 2017

Prof. [S. Gordon](#), [Security and Cryptography 2014](#), [CQ University Australia](#), ultimo accesso dicembre 2017

[Wikiversità](#), [Funzioni di hash](#), ultimo accesso dicembre 2017

[Wikipedia](#), [Secure Hash Algorithm](#), ultimo accesso gennaio 2018

[Wikipedia](#), [Funzione crittografica di hash](#), ultimo accesso gennaio 2018

E. Baldino, R. Rondano, A. Spano, C. Iacobelli, *Internetworking - Sistemi e reti*, quinto anno, ed. Juvenilia, 2016

[NetworkLessons.com](#), [IPsec \(Internet Protocol Security\)](#), ultimo accesso febbraio 2018

Prof. [L. Alcuri](#), [Sistemi di cablaggio strutturato](#), Università di Palermo, Dipartimento di Ingegneria Elettrica, Elettronica e delle Telecomunicazioni, di Tecnologie Chimiche, Automatica e Modelli Matematici, Gruppo Telecomunicazioni e Teoria dell'Informazione

P. Ollari, *Corso di sistemi e reti per Informatica - Applicazioni e sicurezza in rete*, vol. 3, ed. Zanichelli, 2013

[Cisco Packet Tracer Labs](#), ultimo accesso maggio 2018

[Ing. Alfredo Centinaro](#), ultimo accesso marzo 2019

[Antoine Aflalo](#), [Elastic Stack with TLS](#), ultimo accesso marzo 2019

For the future

ACL on MAC address:

[https://www.techrepublic.com/article/cisco-administration-101-understanding-etherne t-mac-addresses/](https://www.techrepublic.com/article/cisco-administration-101-understanding-ethernet-mac-addresses/)