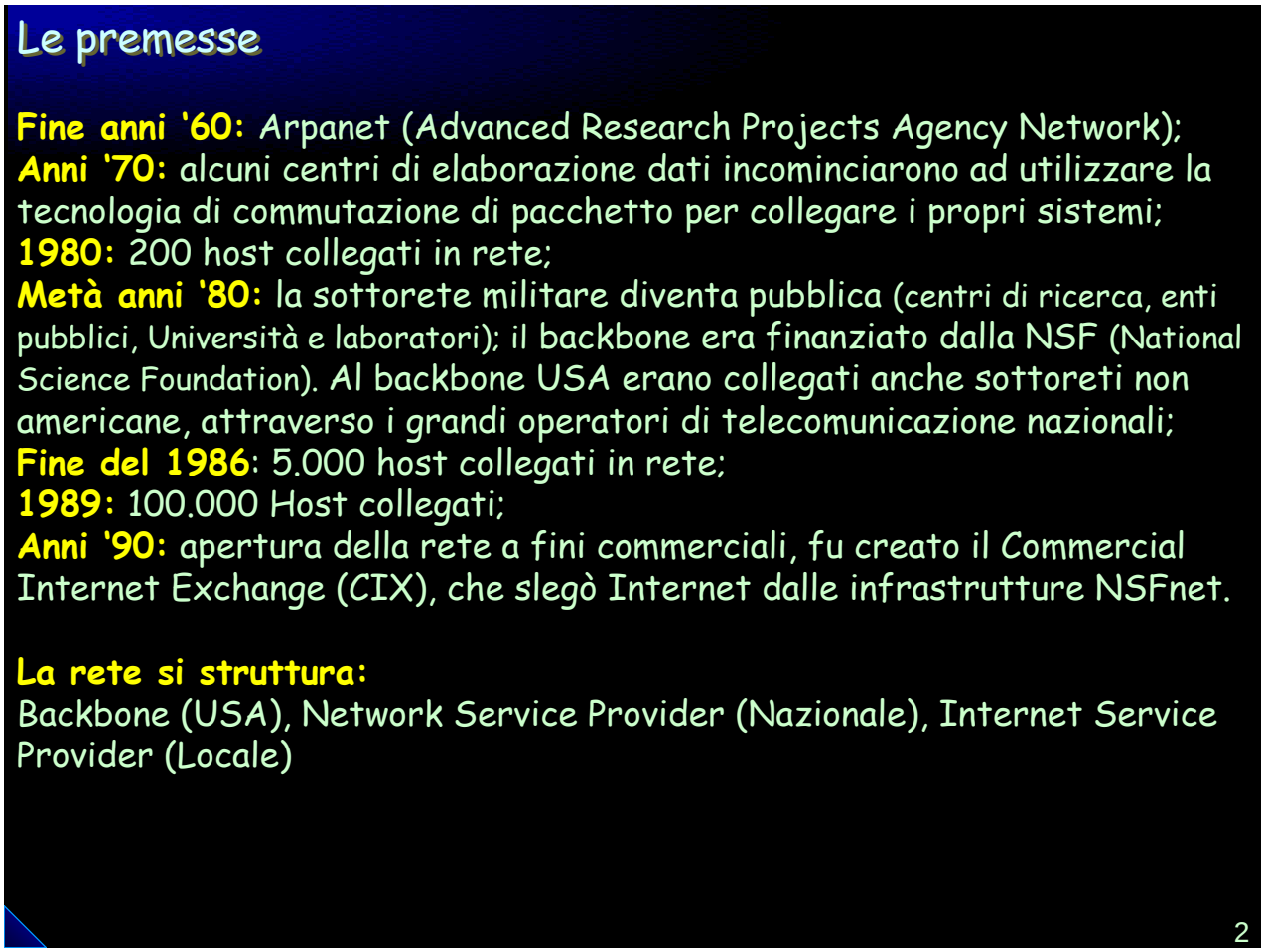
2. TCP/IP

- Le premesse
- Il modello di riferimento dei protocolli
- Cenni sull'Internet Protocol Suite
- IP Address
- IP Header
- Dimensioni delle trame delle principali reti
- Funzioni di routing
- Aspetti di gestione della rete
- Protocolli del livello Data Link: SLIP e PPP
- Address Resolution Protocol: ARP e RARP
- Protocolli del livello Trasporto: TCP e UDP
- Livelli Applicativi
- Domain Name System: DNS

1

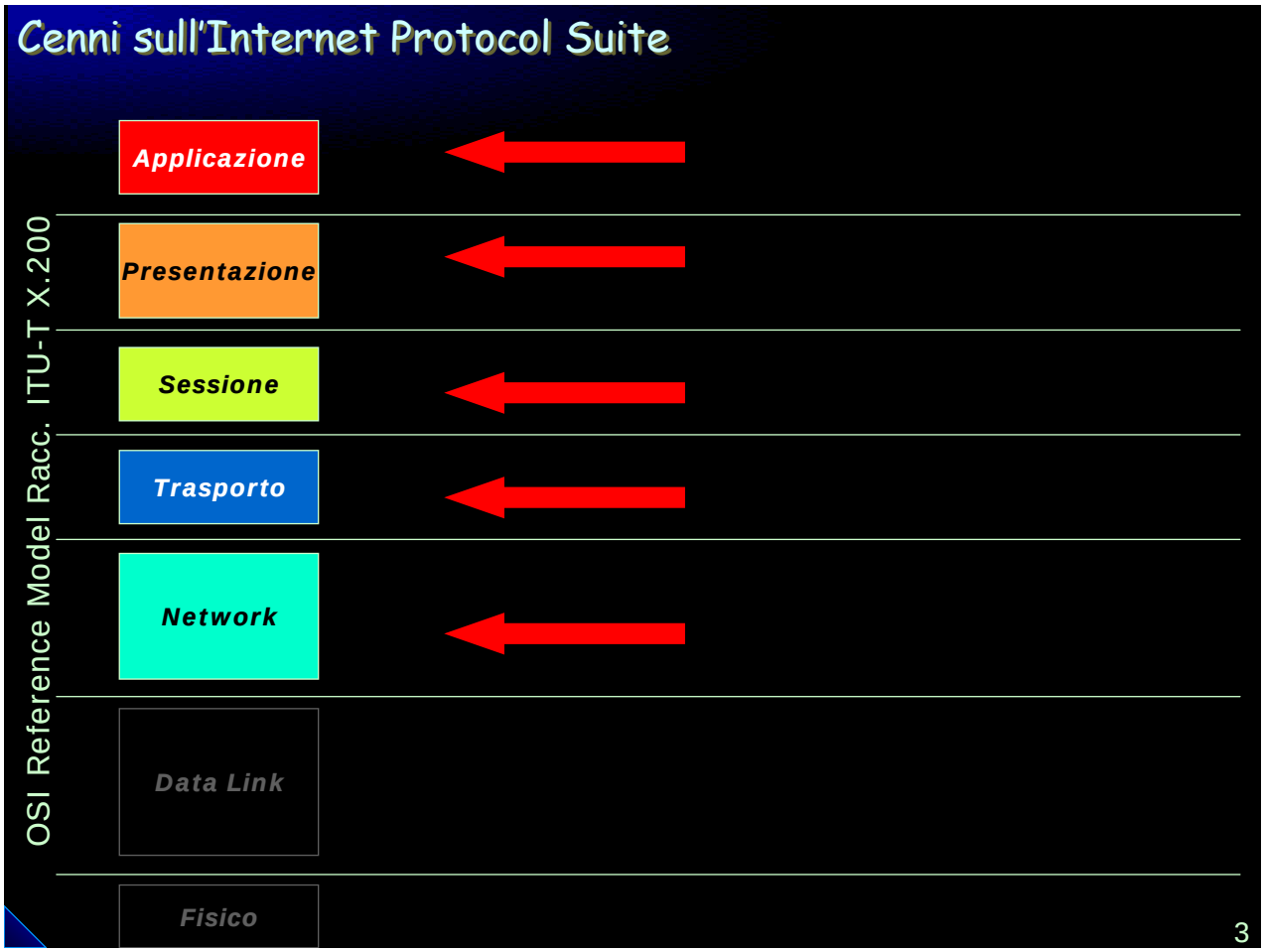
Note



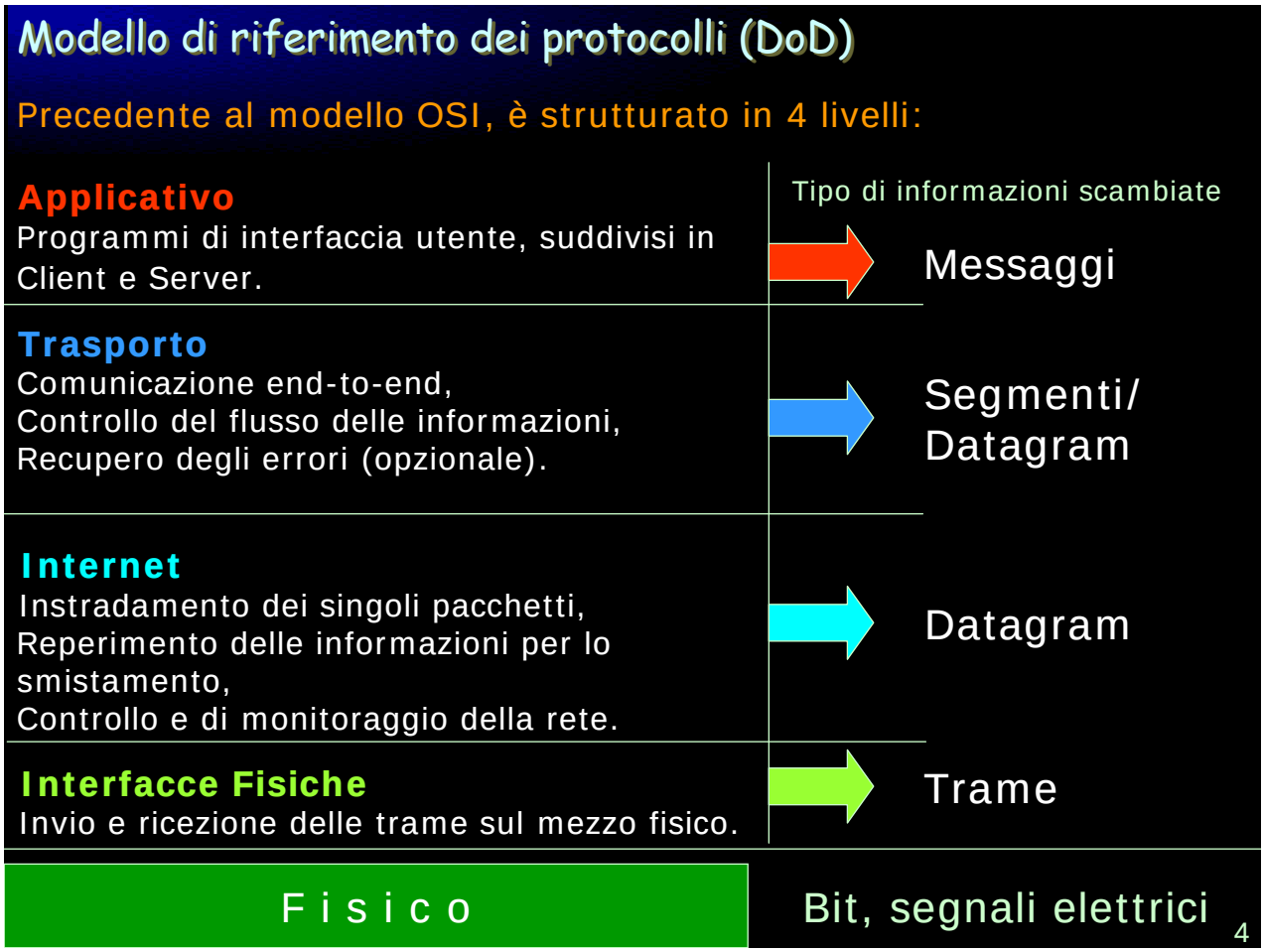
Internet è l'evoluzione della rete Arpanet, sviluppata alla fine degli anni sessanta per esigenze militari, quando il Dipartimento della Difesa statunitense incaricò l'ARPA (Advanced Research Projects Agency) di studiare un sistema telematico capace di resistere ad un possibile attacco nucleare, un sistema capace cioè di indirizzare automaticamente i pacchetti di dati all'interno della rete, anche qualora uno o più calcolatori fossero inutilizzabili o distrutti. Nei primi anni settanta alcuni centri di elaborazione dati incominciarono ad utilizzare la tecnologia di commutazione di pacchetto per collegare i propri sistemi e nel 1980 vi erano circa 200 host collegati in rete. Verso la metà degli anni '80 la sottorete militare venne scorporata e resa indipendente (MilNet). La rete Arpanet fu così utilizzata per collegare centri di ricerca, enti pubblici, Università e laboratori; la dorsale principale di comunicazione (backbone) era comunque ancora finanziata dalle autorità statutinensi attraverso la NSF (National Science Foundation) e vi erano limitazioni all'uso della rete per fini commerciali. Alla dorsale principale statunitense erano collegati anche alcune sottoreti non americane, attraverso i grandi operatori di telecomunicazione nazionali. Alla fine del 1986 vi erano 5.000 host collegati in rete, e tre anni più tardi il numero raggiunse le 100.000 unità. Per consentire l'utilizzo anche a fini commerciali, nei primi anni novanta fu creato il Commercial Internet Exchange (CIX), che slegò Internet dalle infrastrutture NSFnet e, quindi, dalle restrizioni imposte dalle autorità statunitensi.

Lo sviluppo della rete e l'ingresso di operatori commerciali ha portato alla creazione di una struttura gerarchica prima assente. Internet oggi si appoggia su alcune grandi reti nazionali, gestite dai grandi operatori di telecomunicazioni, detti anche Network Service Provider (NSP). A loro volta, le reti nazionali sono collegate da una parte alle grandi dorsali di comunicazione (quelle principali operano alla velocità di 45 Mb/s) e dall'altro alle sottoreti create dagli Internet Service Provider, che offrono l'accesso ad aziende e privati; a tale scopo i Service Provider dispongono di punti di accesso locali detti POP (Point of Presence).

Nonostante il grande sviluppo della rete Internet essa non è posseduta né gestita da una singola autorità, anche se esistono diversi enti ed associazioni volontari che assicurano il mantenimento e lo sviluppo degli standard necessari al funzionamento del sistema.


Cenni sull'Internet Protocol Suite
OSI Reference Model Racc. ITU-T X.200
Applicazione
Presentazione
Sessione
Trasporto
Network
Data Link
Fisico
3

Note



Il modello a strati usato dai protocolli TCP/IP e' precedente ad OSI, e viene comunemente chiamato modello Department of Defense (DoD) (Dipartimento della Difesa Americano).

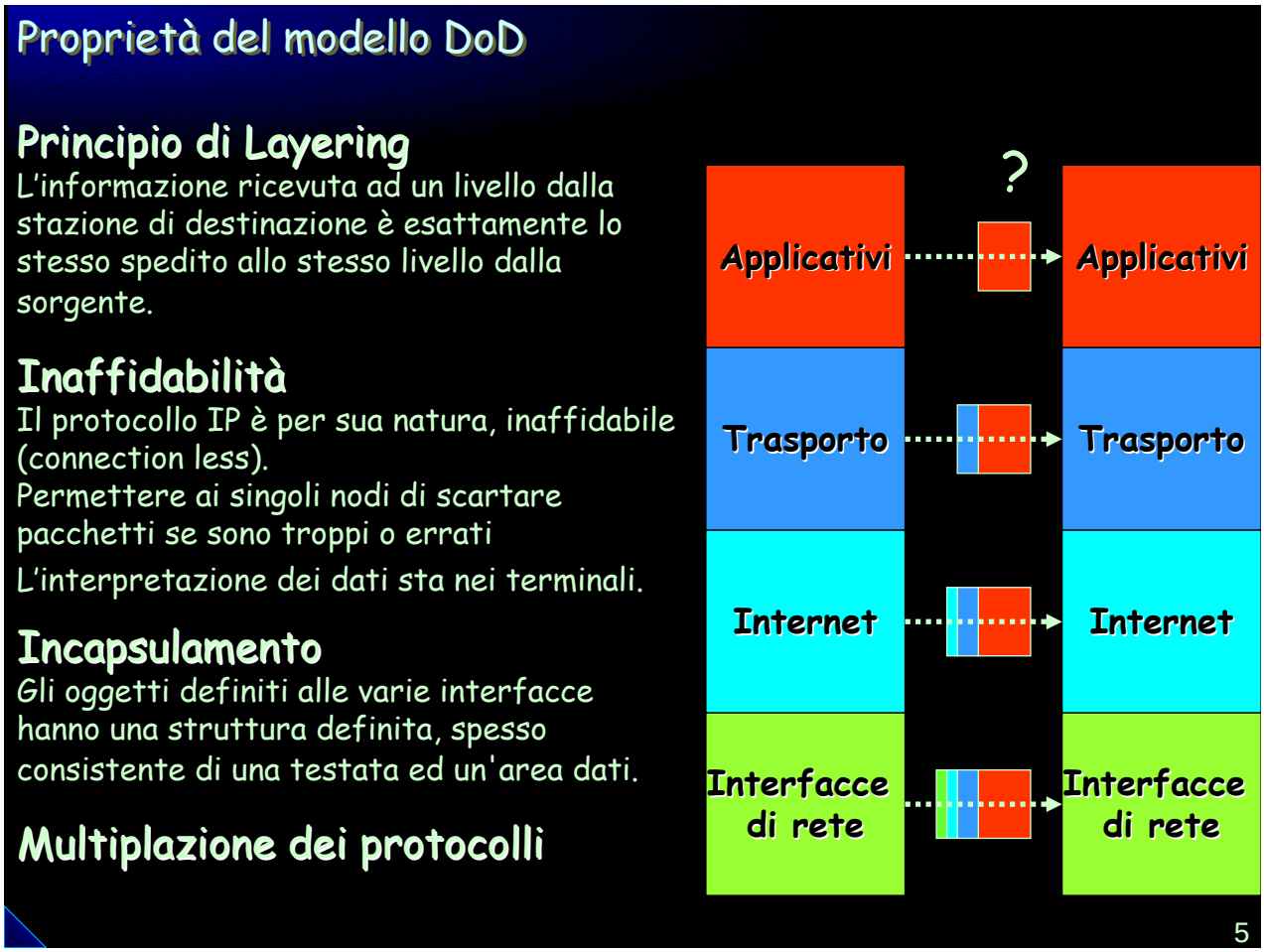
I quattro strati che compongono il modello DoD sono:

- Applicativo, fornisce i programmi di interfaccia utente, suddivisi in Client e Server;

- Trasporto, fornisce la comunicazione tra le due stazioni terminali portanti gli applicativi (end-to-end communication), regola il flusso delle informazioni, e puo' fornire un trasporto affidabile, cioe' con recupero errori;

- Internet, si occupa dello smistamento dei singoli pacchetti su una rete complessa e interconnessa, del reperimento delle informazioni necessarie allo smistamento, dello scambio di messaggi di controllo e di monitoraggio rete,

- Interfacce Fisiche, è responsabile dell'interfacciamento con il mezzo fisico e dell'invio e ricezione su di esso dei pacchetti. E' un livello che non viene ben specificato in quanto demanda alle reti già disponibili il compito di trasferire le trame contenenti i pacchetti IP. Le soluzioni normalmente adottate sono: in ambito geografico HDLC, SLIP, PPP, X.25, Frame Relay, ATM; in ambito locale Ethernet/IEEE802.3, Token Ring, FDDI.



Principio di Layering

L'informazione ricevuta ad un livello dalla stazione di destinazione è esattamente la stesso spedito allo stesso livello dalla sorgente.

Inaffidabilità

La filosofia architetturale di TCP/IP e' semplice: costruire una rete che possa sopportare il carico in transito, ma permettere ai singoli nodi di scartare pacchetti se il carico e' temporaneamente eccessivo, o se risultano errati o non recapitabili. L'incarico di rendere il recapito pacchetti affidabile non spetta allo strato di Internet, ma a strati software superiori. Si dice che il protocollo IP e' per sua natura, inaffidabile.

In generale e' lo strato trasporto che si occupa del controllo di flusso e del recupero errori. Infatti la sede principale di "intelligenza" della rete è a livello trasporto o superiore. Le singole stazioni collegate alla rete non fungono soltanto da punti terminali di comunicazione, ma possono anche assumere il ruolo di router, per l'interscambio di pacchetti da una rete ad un'altra.

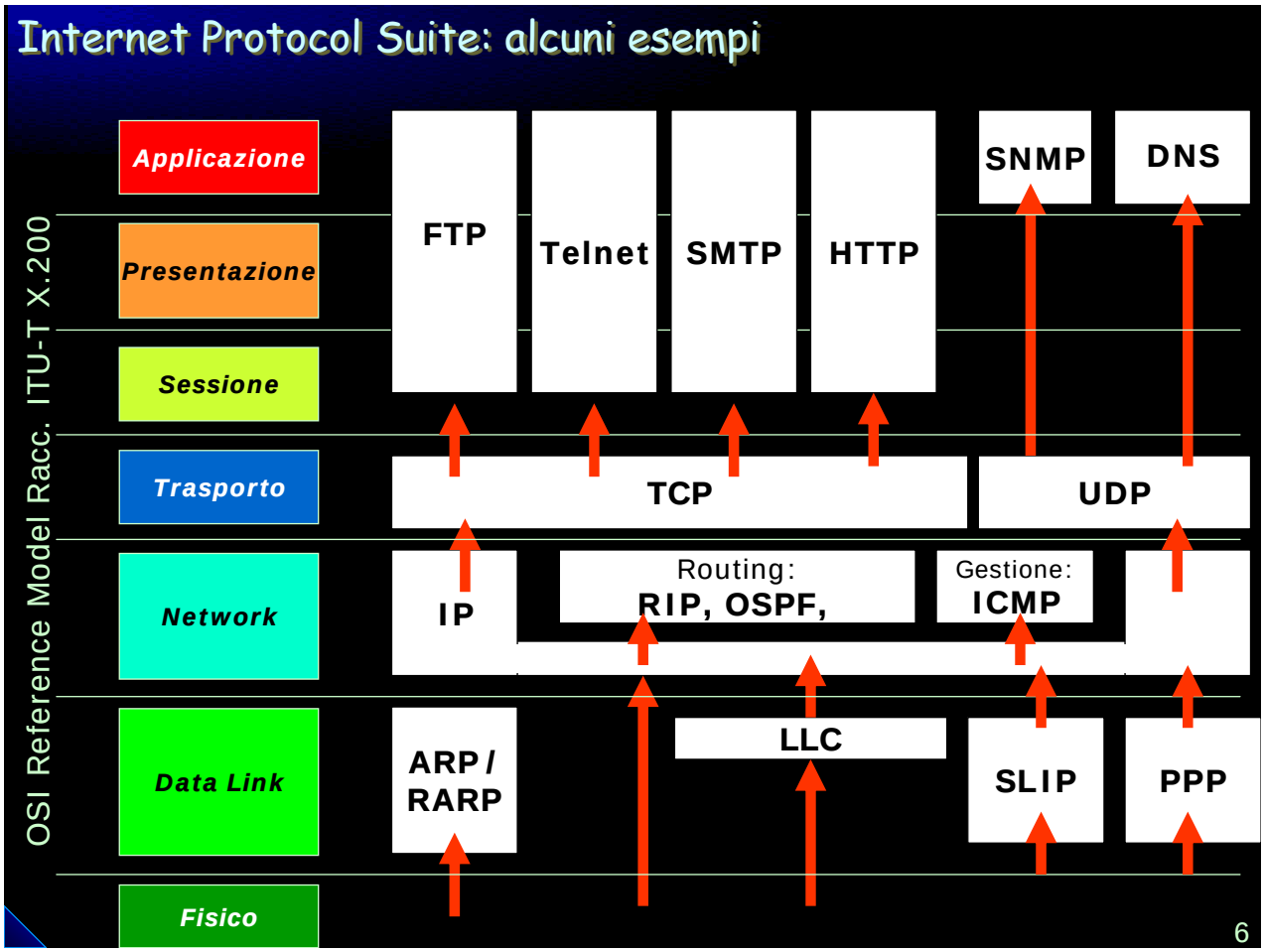
Incapsulamento

Gli oggetti definiti alle varie interfacce hanno una struttura definita, spesso consistente di una testata ed un'area dati.

Multiplazione dei protocolli

Più protocolli di trasporto interfacciano simultaneamente il protocollo IP. Più protocolli a livello Internet simultaneamente inviano trame tramite il livello Intefacce Fisiche.

In generale le informazioni a livello superiore vengono multiplate a livello inferiore incapsulandole come dati ed inserendo in un campo della testata a livello inferiore la informazione di come far avvenire la demultiplazione.



I servizi Internet sono supportati da una metodologia di comunicazione basata sulla trasmissione di pacchetti di dati gestita dai protocolli TCP/IP.

Viene comunemente identificata con il nome TCP/IP tutta una serie di protocolli di rete che si sono andati via via sviluppando a partire dal 1969 ad oggi. Vi sono letteralmente molte decine di tali protocolli.

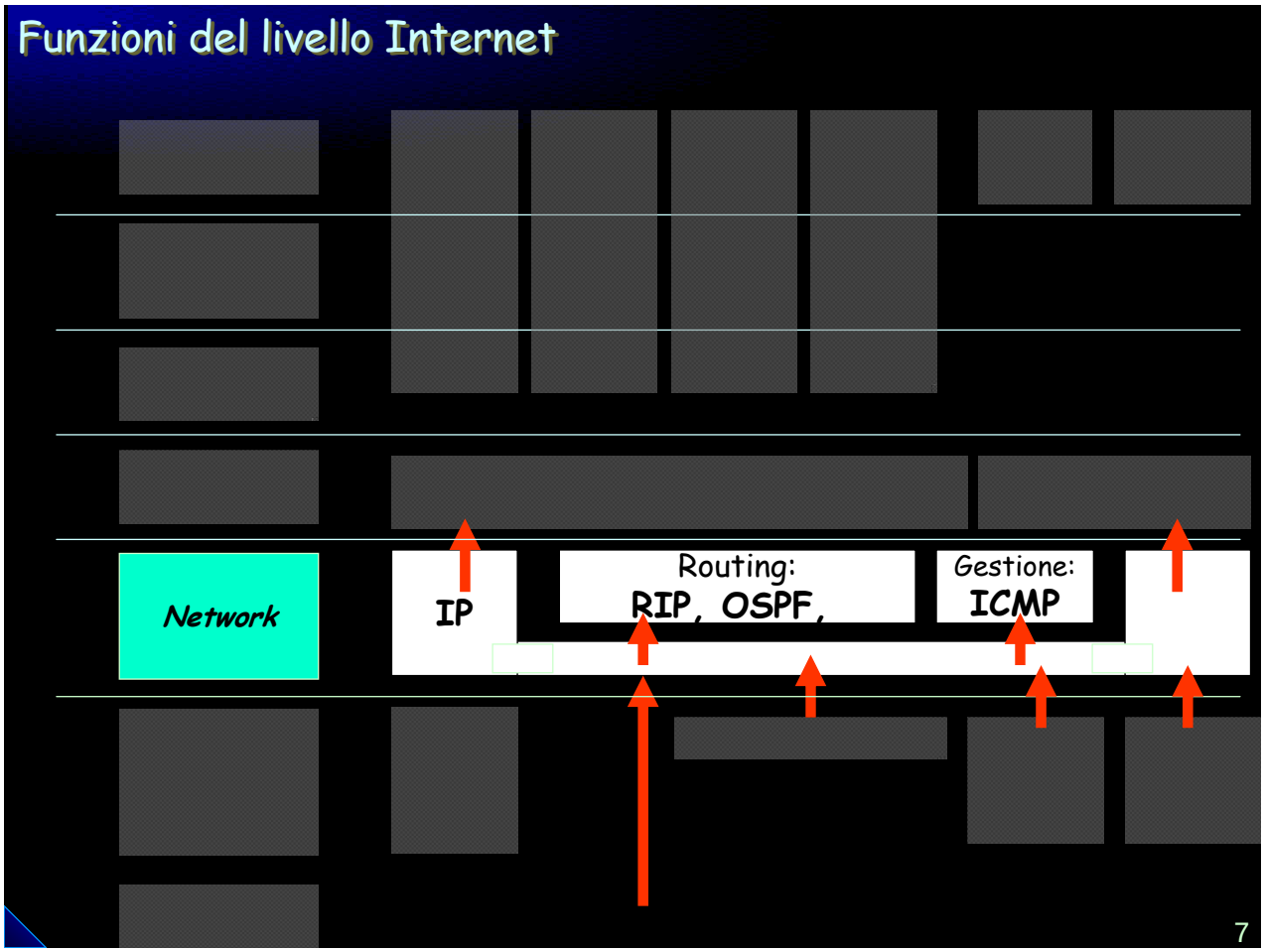
Non tutti i protocolli di rete della serie TCP/IP sono necessari o supportati; molti sono obsoleti, alcuni solo sperimentali in quanto la situazione e' in continua evoluzione.

Le caratteristiche salienti, che stanno alla base della filosofia tradizionale dei protocolli TCP/IP, e che hanno condizionato finora il design dei protocolli sono:

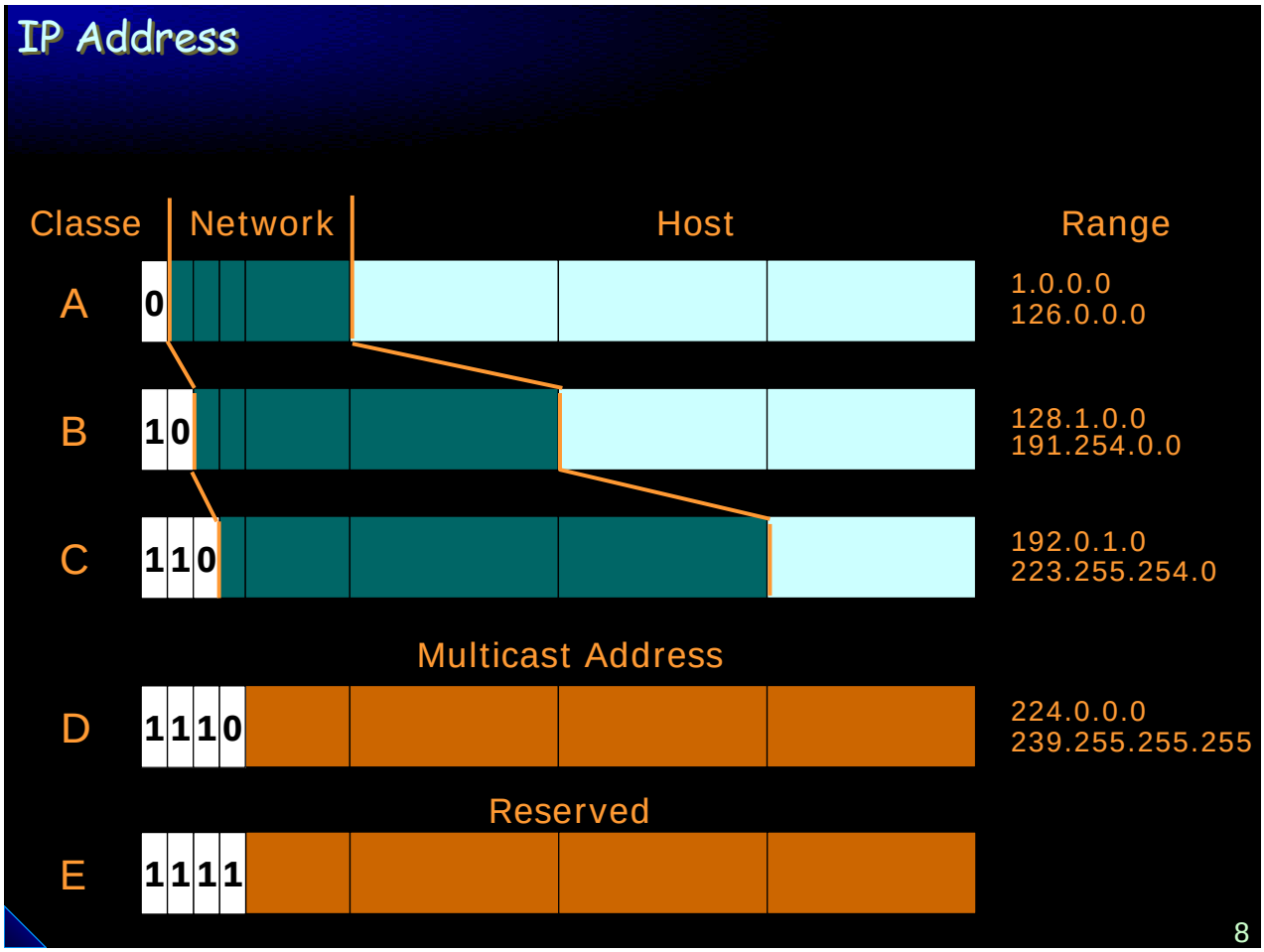
♦ inaffidabilità della rete trasmissiva (si parla degli anni 70). Spetta ai terminali della comunicazione di rete la responsabilità di verificare la conformità dei messaggi ricevuti. Vengono pertanto inventati numerosi protocolli ausiliari che non hanno lo scopo diretto di comunicare dati, ma compiono il controllo, la manutenzione e l'amministrazione della rete.

♦ spirito di cooperazione tra i vari protocolli, dovuto al fatto che le risorse di rete sono poco costose e gli utenti amichevoli.

I principi guida originali sono oggi in crisi in quanto l'utilizzo di Internet è notevolmente cambiato (da una rete militare ad una rete civile e di diffusione culturale e commerciale). Comunque essendo ormai massivamente diffuso si impone come standard di fatto.


Funzioni del livello Internet
Network
IP
Routing:
RIP, OSPF,
Gestione:
ICMP
7

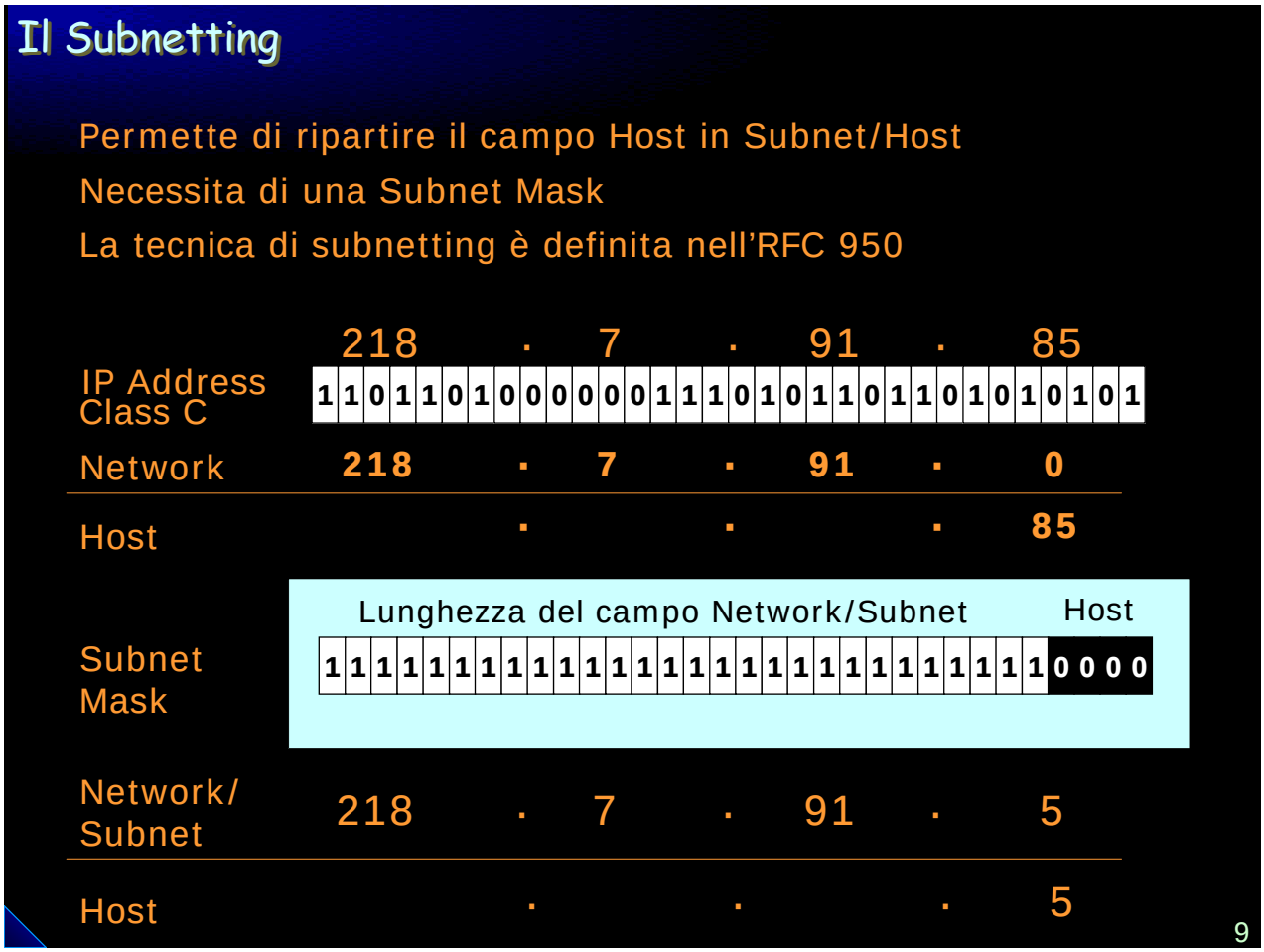
Note:



Gli indirizzi di rete nel modo Internet vengono definiti IP Address. Vale la pena ricordare che questi sono indirizzi univoci sulla rete, pertanto costituiscono una risorsa che va razionalizzata. Purtroppo l'assegnazione di tali indirizzi fu inizialmente non troppo azzeccata e sui potenziali 4 miliardi di numeri che si possono ottenere (32 bit), una buona parte è stata sprecata.

La lunghezza dell'IP Address è di 32 bit suddivisi in 4 byte i quali compongono 2 campi (Network e Host). La dimensione dei campi Network e Host dipende dal tipo di classe (A, B, C, D, E).

- Classe A, poche reti di dimensioni grandissime.
- Classe B, intermedia.
- Classe C, molte reti di piccole dimensioni.
- Classe D, indirizzi per applicazioni multicast.
- Classe E, indirizzi riservati per usi futuri.



Mascheramento

La metodologia di assegnazione degli indirizzi è molto inefficiente in quanto viene assegnato un indirizzo di rete ad un richiedente, anche se questi non usa tutti gli indirizzi di host disponibili. L'espansione recente di Internet ha determinato l'esaurimento degli indirizzi in classe B.

Pertanto la gerarchia di indirizzamento a due livelli Network e Host, viene spesso estesa ad avere un terzo livello gerarchico Subnet. Una Subnet è una suddivisione del campo Host.

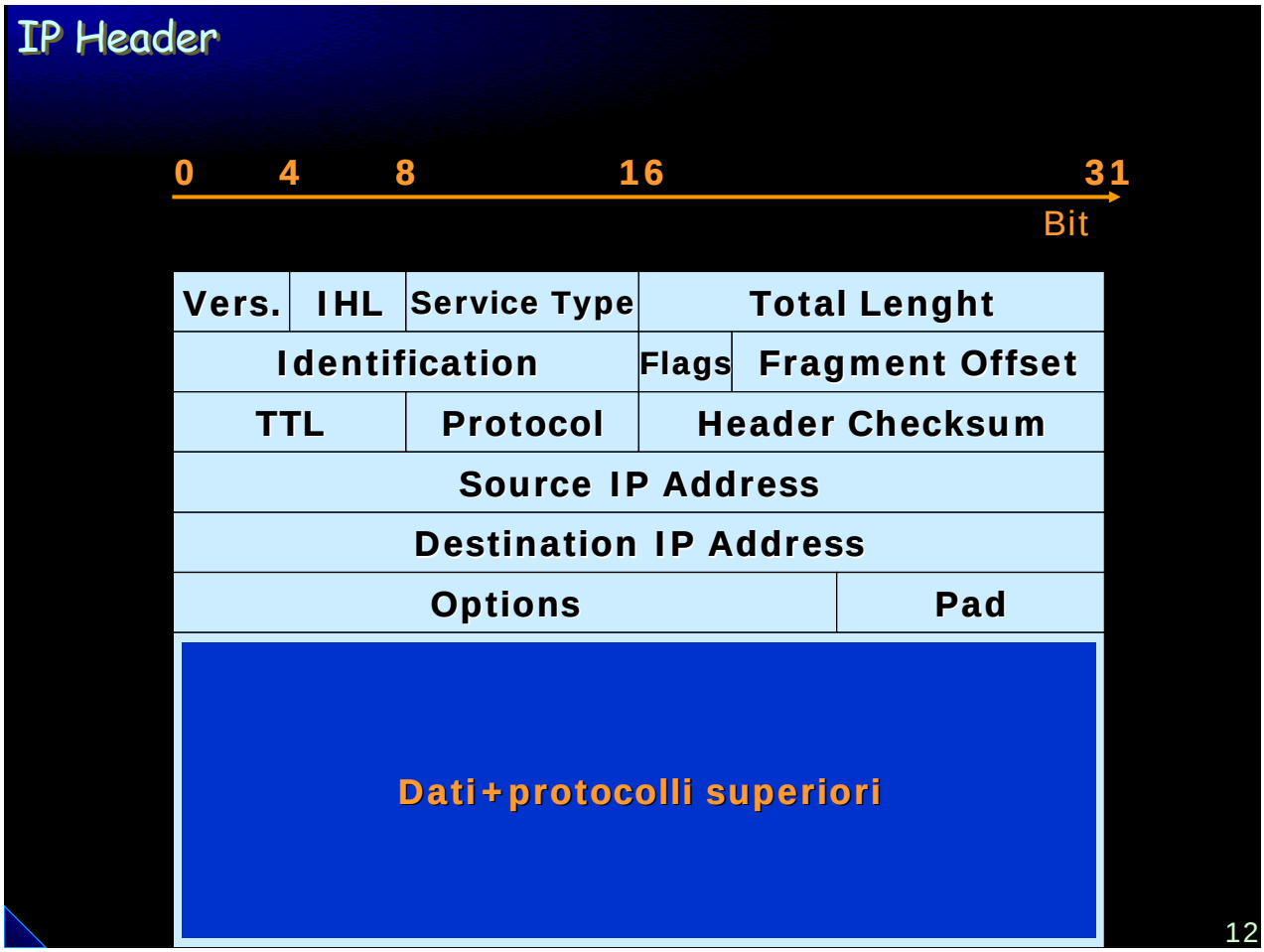
Il campo Network/Subnet che si viene a creare è specificato da una Subnet Mask a 32 bit. La subnet Mask contiene bit a 1 in corrispondenza dei campi Network e Subnet, mentre i bit valgono 0 in corrispondenza del campo Host. L'ampiezza dei campi Subnet e Host può essere definita in modo flessibile entro i limiti consentiti dalle classi di indirizzamento (classe A - 3 byte, classe B - 2 byte, classe C - 1 byte). Il riconoscimento dell'indirizzo di Subnet avviene con una operazione di AND logico tra l'IP Address e la Subnet Mask.

All'interno di una Network la Subnet Mask deve essere univoca e residente su tutti gli Host connessi.

Gli indirizzi internet non designano le stazioni di rete ma le interfacce di rete. Una stazione con piu' interfacce e' detta "multi-homed". Questo e' tipicamente, ma non solo, il caso dei router, che compiono lo smistamento tra due o piu' reti contigue.

Tutti gli host mantengono Tabelle di Routing che specificano come far arrivare un pacchetto ad una stazione remota, inviandolo ad un router locale che esegue poi lo smistamento. L'associazione di indirizzi internet ad interfacce permette di ottenere del routing (smistamento) di precisione, e permette ai router di scegliere il percorso migliore per una determinata destinazione.

L'header di un datagramma IP e' definita dallo RFC791.



L'IP (Internet Protocol) è il protocollo che si occupa di consegnare al computer remoto i singoli pacchetti, che vengono generati dai livelli superiori (ossia, dal TCP o, più raramente, da altri protocolli di controllo della trasmissione). A questo scopo, esso prende ogni singolo blocco di dati che arriva dal livello superiore e vi aggiunge una ulteriore intestazione di 20/60 byte.

L'IP è un protocollo di tipo non connesso (connectionless): questo significa che i dati vengono affidati alla rete, che poi, in maniera separata pacchetto per pacchetto, provvede a consegnarli; chi trasmette non attende alcun segnale di ricevuto dal destinatario, e non si ha controllo sul fatto che la trasmissione vada a buon fine, che la velocità di trasmissione sia appropriata, o che i pacchetti non si mescolino e non si duplichino. Ovviamente, non ci si può aspettare un risultato di qualità da un approccio di questo tipo: difatti, i protocolli di controllo della trasmissione, come il TCP, servono proprio a garantire un minimo di controllo sul corretto ricevimento dei dati.

Il formato dell'IP header è l'RFC 791 del 1981.

Il primo campo è la Versione, settato a 4 per la versione corrente di IP (IPv4). Questo campo varrà 6 nella prossima versione (IPv6).

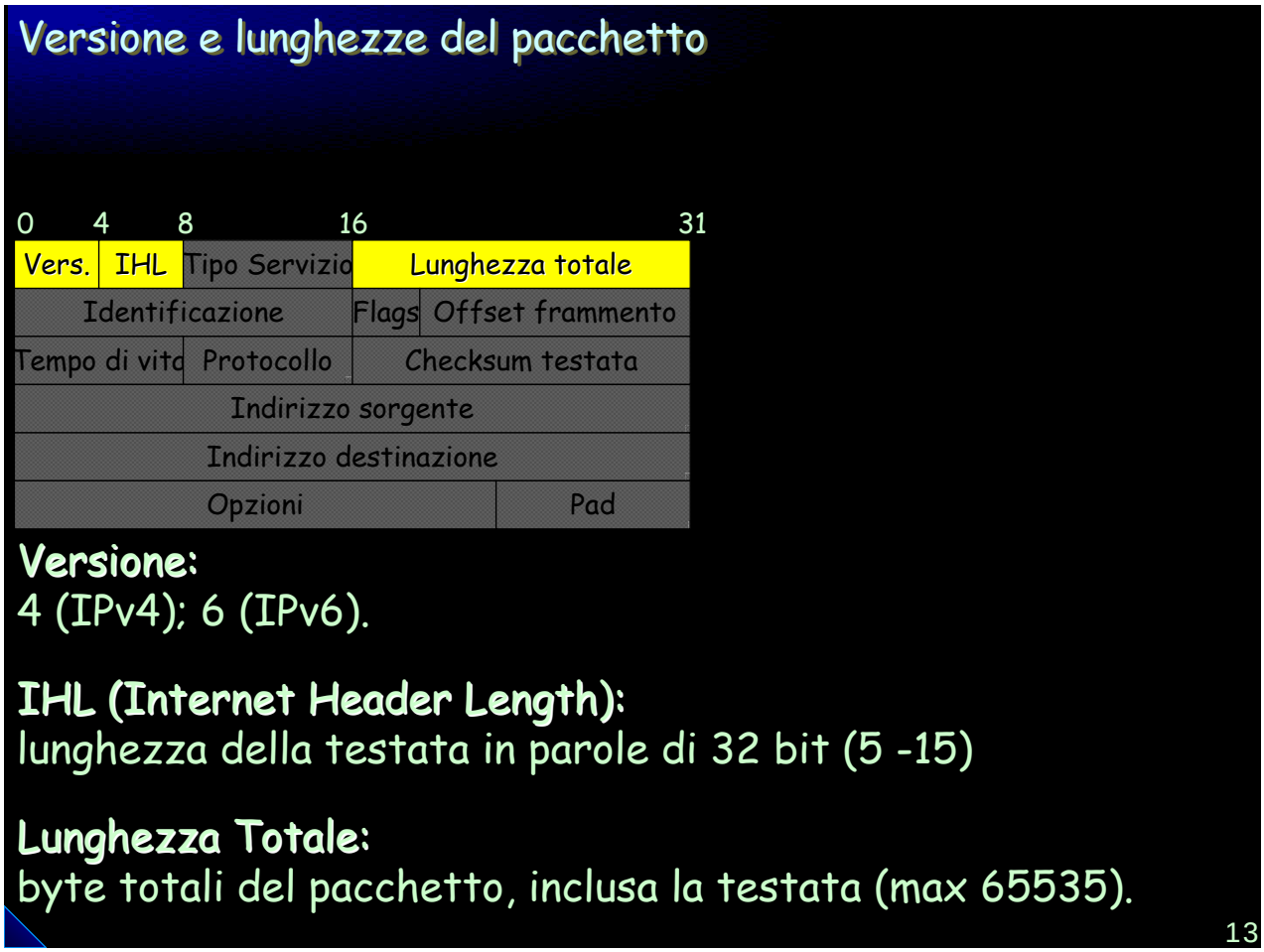
Il campo IHL è l'Internet Header Length, o lunghezza dell'header stessa, misurato in parole da 32 bit. IHL varia da 5, quando non vi sono opzioni, ad un massimo di 15, che permette un massimo di 40 byte per le opzioni.

Il Service Type definisce la precedenza del pacchetto ed il tipo di routing desiderato. E' composto da due sottocampi: Priorità e Routing.

La Total Lenght è il numero di byte totale del pacchetto, incluso l'header; la dimensione massima di un pacchetto IP e' di 65535 byte.

Il campo Identificativo esprime l'identità del pacchetto originale.

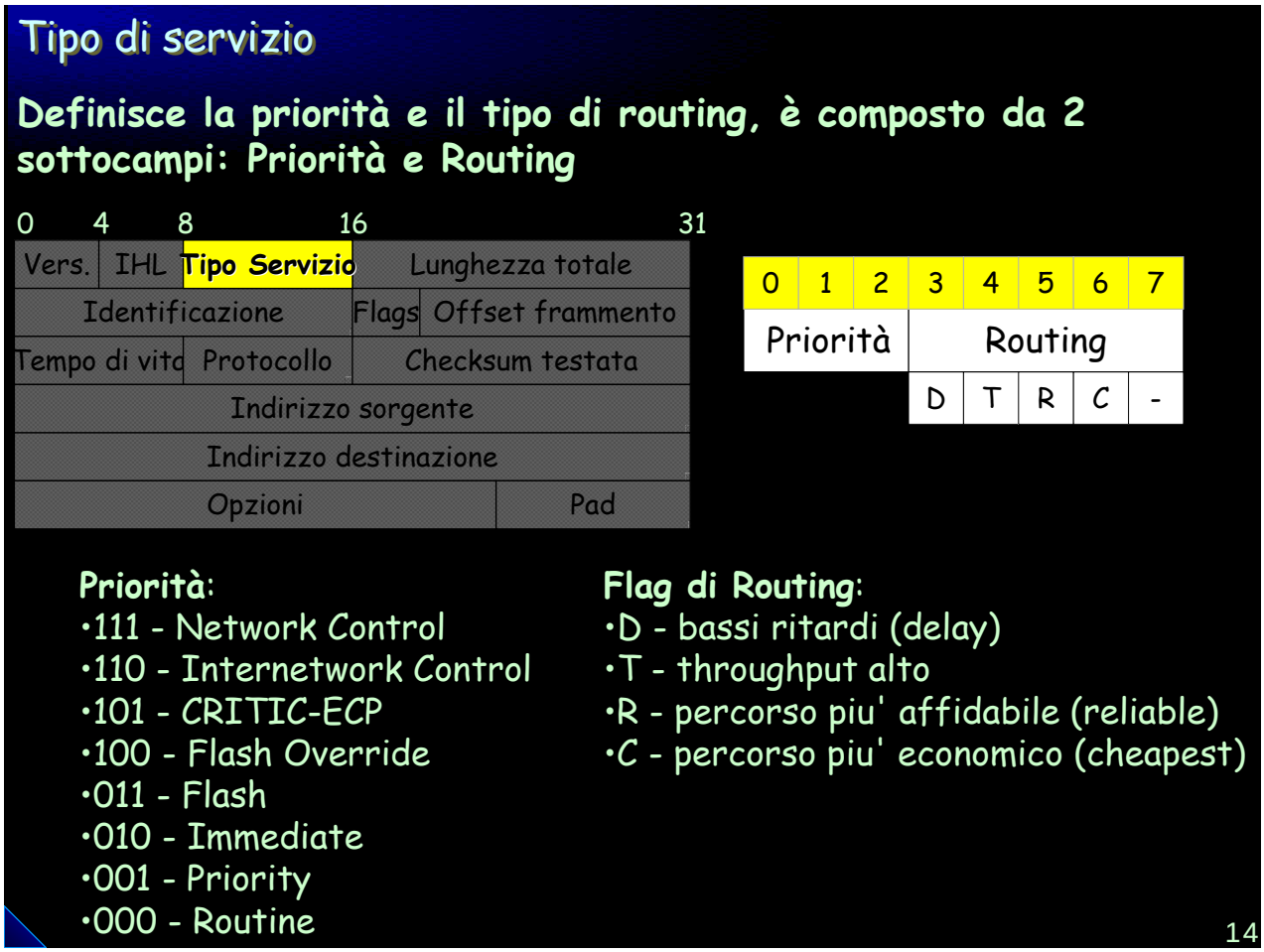
Il campo Protocollo identifica il programma a cui il pacchetto deve essere passato quando giunge a destinazione. Sono definiti vari numeri identificativi di protocollo.



Il primo campo e' la Versione, settato a 4 per la versione corrente di IP (IPv4). Questo campo varrà 6 nella prossima versione (IPv6).

Il campo IHL è l'Internet Header Length, o lunghezza della testata stessa, misurato in parole da 32 bit. IHL varia da 5, quando non vi sono opzioni, ad un massimo di 15, che permette un massimo di 40 byte per le opzioni.

La Lunghezza Totale e' il numero di byte totale del pacchetto, inclusa la testata: la dimensione massima di un pacchetto IP e' di 65535 byte.



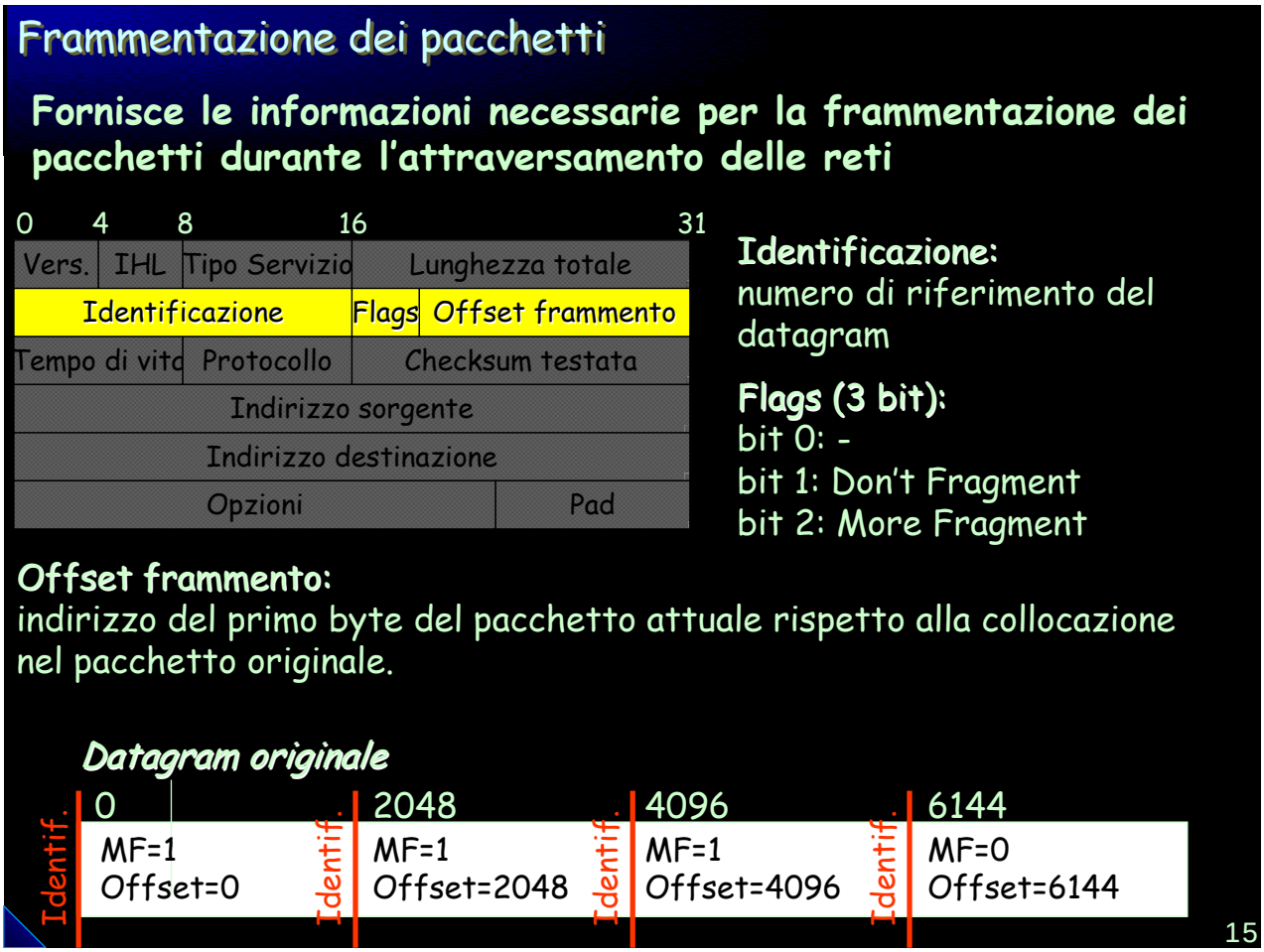
Il Tipo Servizio definisce la precedenza del pacchetto ed il tipo di routing desiderato.

Il campo Tipo di Servizio ha due sottocampi: Precedenza e Tipo di Servizio.

La **precedenza** e' un'indicazione della priorita' del pacchetto e ne determina il trattamento in una coda. Vi sono otto valori di precedenza:
•111 - Network Control
•110 - Internetwork Control
•101 - CRITIC-ECP
•100 - Flash Override
•011 - Flash
•010 - Immediate
•001 - Priority
•000 - Routine

Il documento RFC791 mette in guardia da possibili abusi del campo di sicurezza e suggerisce che tale campo debba essere gestito solo all'interno di una rete locale.

Il campo **Tipo di Routing** consiste di cinque bit di flag, di cui quattro definiti:
•D - bassi ritardi (delay)
•T - throughput (banda passante) alto
•R - percorso piu' affidabile (reliable)
•C - percorso piu' economico (cheapest)
Solo uno dei bit del campo puo' essere settato.



Segmenti diversi di rete supportano trame a livello 2 di lunghezze massime diverse. La lunghezza massima tipica di ogni segmento si chiama Maximum Transmission Unit (MTU). Pertanto il transito dei pacchetti su più reti determina problemi di incompatibilità della lunghezza dell'unità informativa. La risoluzione a tale aspetto viene data con la frammentazione (o segmentazione) dei dati.

Nell'header IP i campi Identificazione, Flag ed Offset di Frammento della testata IP gestiscono la frammentazione.

Il campo **Flags** e' composto da tre bit.

Il primo e' sempre zero. Il secondo e' il bit Don't Fragment (DF) ed indica che il pacchetto originale non deve essere frammentato. Se questo bit e' settato e vi e' necessita' di frammentazione, il pacchetto viene scartato, e un messaggio (ICMP) e' inviato al mittente.

Il bit More Fragments (MF) e' settato a 1 per tutti i frammenti del pacchetto originario tranne l'ultimo.

Il campo **Offset Frammento** e' l'indirizzo in byte che il primo byte del frammento corrente occupava nel datagram originario.

La frammentazione puo' avvenire in piu' punti del percorso in rete di un datagram, e puo' anche avvenire ricorsivamente. I campi Offset e MF sono sempre espressi relativi al pacchetto originale.

Il campo **Identificativo** esprime l'identita' del pacchetto originale.

Il riassemblaggio del pacchetto e' compiuto solo dalla destinazione finale, che deve ricomporre tutti i frammenti con lo stesso identificativo.

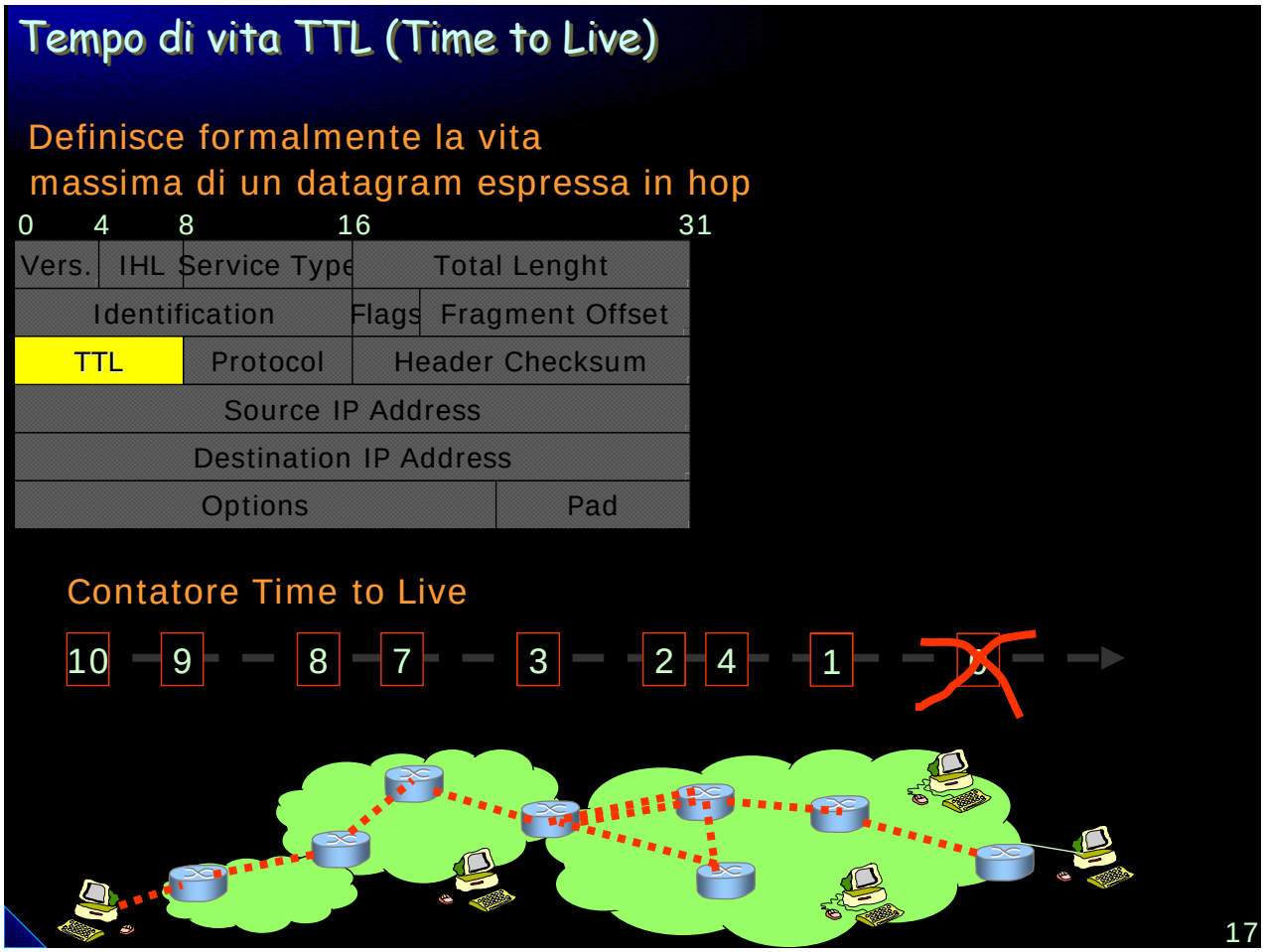
In caso di perdita in transito anche di un solo frammento, l'intero pacchetto viene scartato. Il ricevente ha un tempo di timeout per consentire a tutti i pacchetti di giungere a destinazione: il timeout e' implementato in modo naturale decrementando il Tempo di Vita del pacchetto ogni secondo e scartando il pacchetto se questo raggiunge lo zero.

Una caratteristica del livello Data Link è il limite massimo della trama (MTU) che può essere trasmessa. Questo limite dipende dai dettagli del protocollo specifico usato a livello data link, non e' una caratteristica solo o sempre solo del mezzo trasmissivo. Per esempio, CSMA/CD (IEEE 802.3) ha un limite di dati di 1500 byte.

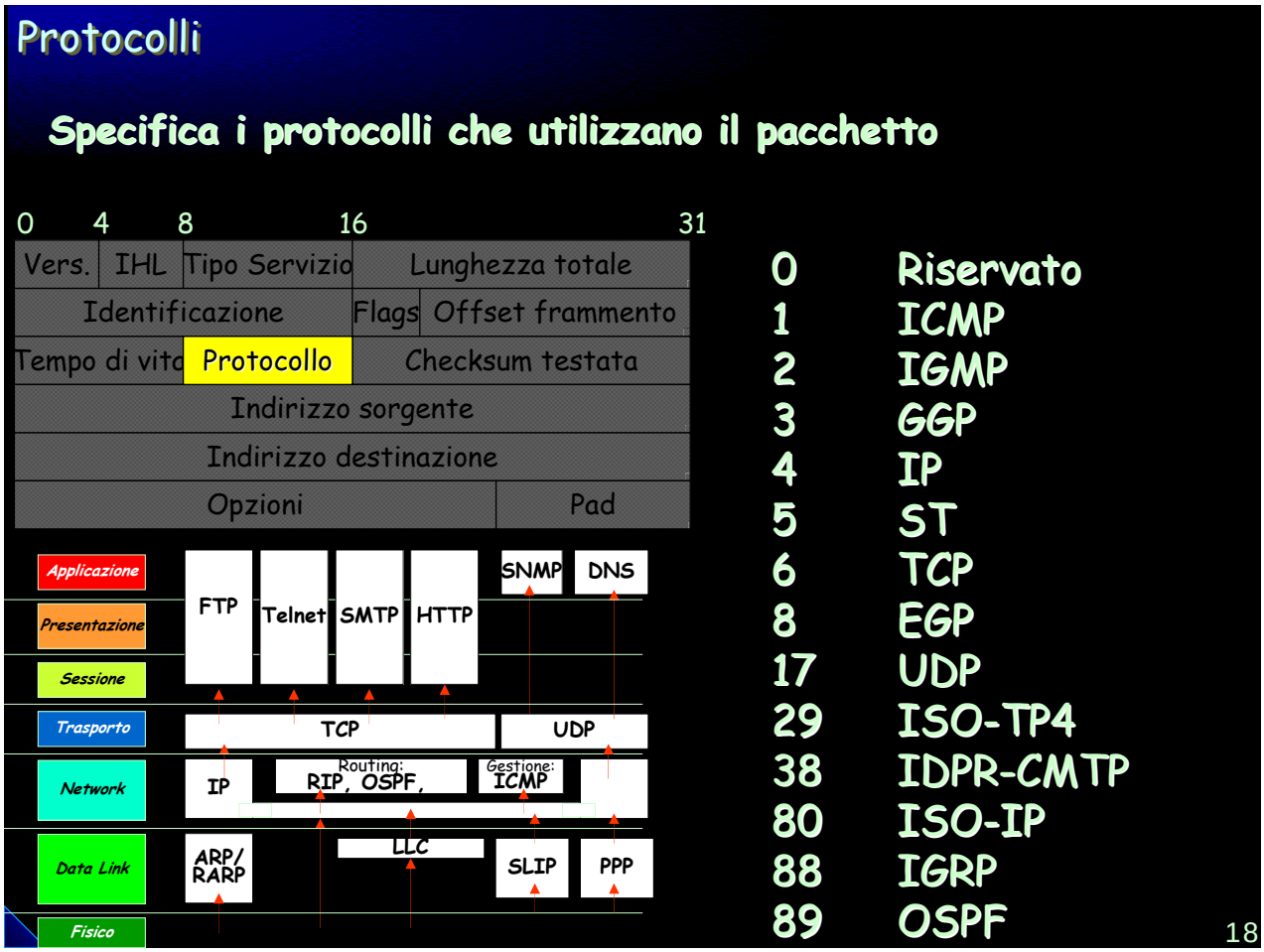
Nella comunicazione tra due stazioni attraverso molti reti diverse ha importanza la MTU minima dell'intero percorso, chiamata Path MTU. Questo valore determina la lunghezza massima di un pacchetto al di sopra della quale il pacchetto verra' certamente frammentato.

E' da notare che la Path MTU non è necessariamente simmetrica e può essere diversa nelle due direzioni di un percorso.

Il documento RFC1191 descrive un metodo per scoprire la Path MTU.

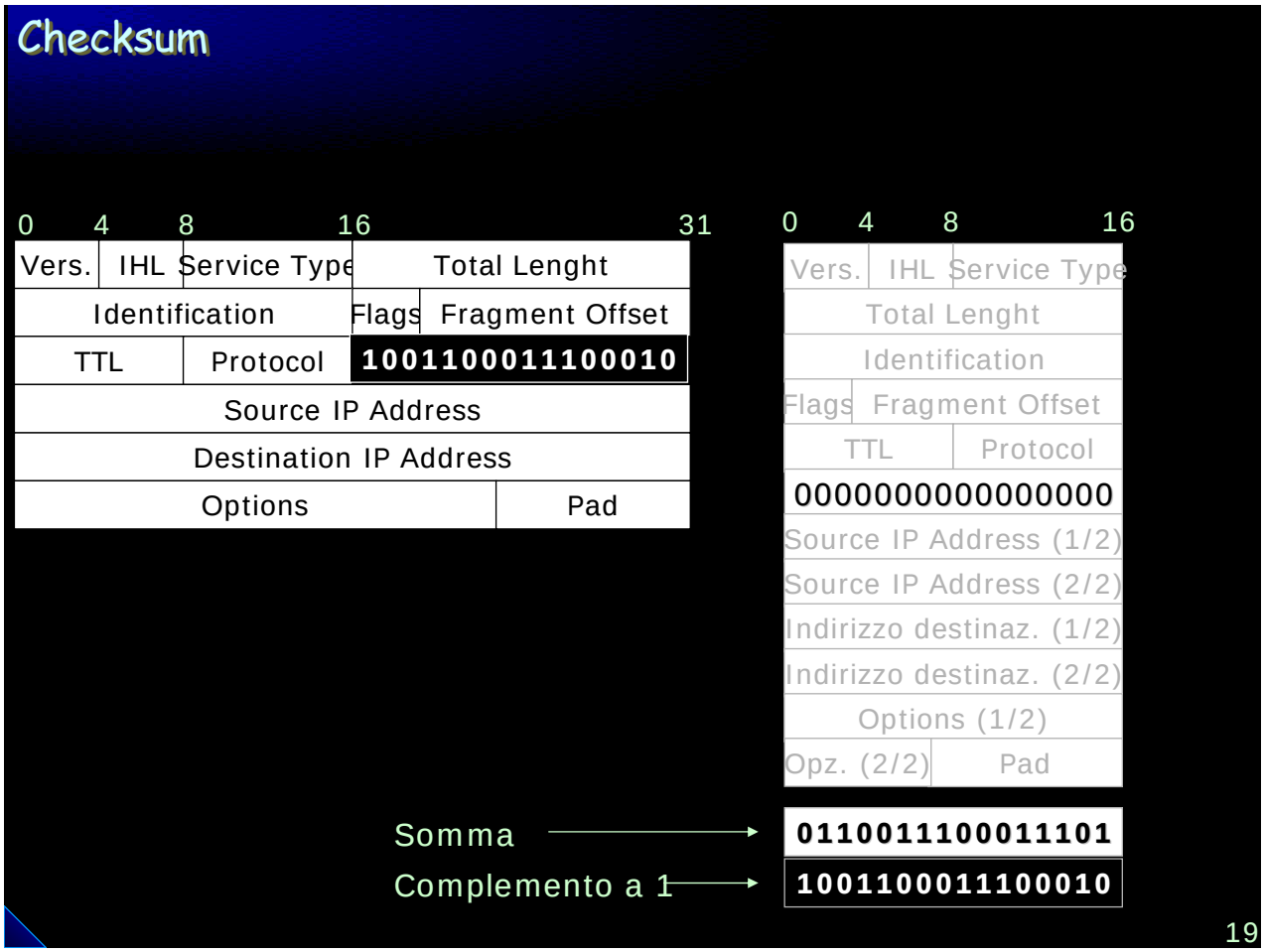


Il campo TTL (Time To Live - TTL) definisce formalmente la vita massima di un pacchetto espressa in secondi, allo scopo di impedire che alcuni pacchetti entrino in un loop infinito di routing e persistano per sempre nella rete. Questo campo è molto utile anche a livelli superiori. Per esempio, se TCP attende lo scadere del TTL dopo la chiusura di una connessione, ha la garanzia che non arrivino più pacchetti che appartenevano alla connessione chiusa. Il TTL come tempo in secondi è di difficile implementazione; in pratica i router decrementano di uno questo campo ogni volta che inviano un pacchetto in via di smistamento. Se il valore del TTL scende a zero, il pacchetto viene scartato.



Il campo Protocollo identifica il programma a cui il pacchetto deve essere passato quando giunge a destinazione. Sono definiti vari numeri identificativi di protocollo.

0	Riservato	
1	ICMP	Internet Control Message
2	IGMP	Internet Group Management
3	GGP	Gateway-to-gateway
4	IP	IP su IP (incapsulamento)
5	ST	Stream
6	TCP	Transmission Control
8	EGP	Exterior Gateway
17	UDP	User Datagram
29	ISO-TP4	ISO Transport Class 4
38	IDPR-CMTP	IDPR Control Message Transport
80	ISO-IP	
88	IGRP	
89	OSPF	Open Shortest Path Fast



Il campo Header Checksum serve a controllare l'integrità dell'header durante la trasmissione. Il checsum è definito come:

il complemento a 1 a 16 bit della somma di tutte le parole a 16 bit dell'header, dopo che il campo Checksum stesso è stato posto a zero.

Questo tipo di checksum è considerato un compromesso accettabile tra velocità di calcolo e capacità di rilevamento errori.

Il problema principale del livello Internet e' lo smistamento dei pacchetti dalla stazione sorgente a quella di destinazione. Lo smistamento si chiama routing e una macchina preposta allo smistamento si dice router.

Il routing avviene a vari livelli: entro la rete locale, entro un gruppo di reti locali interconnesse a far parte di un Sistema Autonomo, fino a livello dell'Internet globale. Si puo dividere il routing in due forme: diretto e indiretto.

Il routing diretto avviene tra due stazioni connesse direttamente allo stesso mezzo fisico. Piu' computer collegati allo steeso mezzo fisico sono nodi della stessa rete o sottorete. Il test di appartenenza alla sottorete e' eseguito semplicamente confrontando logicamente l'indirizzo IP di un nodo con la maschera di sottorete.

Il routing indiretto avviene quando i pacchetti devono essere fatti transitare almeno attraverso un router. Il sistema dei routers che usano i protocolli TCP/IP e' una struttura interconnessa e cooperativa. I pacchetti sono smistati da un router ad un altro finche' non giungono alla rete di destinazione.

Su ogni router è presente una tabella di routing che permette l'instradamento dei pacchetti verso la destinazione.

Il routing e' discusso nel documento RFC1009. Le regole di routing tra reti interconnesse sono discusse in RFC1124. Modelli di direttive di routing sono anche discussi in RFC1104, RFC1092 e RFC1102.

Il protocollo IP non costruisce la Tabella di Routing ma la usa secondo le seguenti modalità:

1. estrae l'indirizzo di destinazione dal pacchetto IP;
2. legge l'indirizzo di rete della rete di destinazione;
3. se l'indirizzo di rete finale coincide con una delle reti localmente connesse in modo diretto, allora il recapito e' diretto ed il pacchetto viene inviato direttamente ad un indirizzo fisico, tramite lo strato Data Link ;
4. altrimenti cerca l'indirizzo di destinazione specifico dell'host nella tabella di routing e lo fa' proseguire verso la destinazione successiva;
5. se non trova nessun riferimento, scarta il pacchetto e segnala l'errore.

Vi possono essere stazioni configurate come gateway ma non come router, per esempio i Firewall. Queste stazioni non eseguono il forwarding del pacchetto tramite il normale algoritmo IP, ma passano il pacchetto a particolari applicativi a livelli superiori, i quali eseguono la decisione se inviare o no il pacchetto. Si tratta quindi di una ricezione ed invio di pacchetto, non di uno smistamento.

L'Internet Control Message Protocol (ICMP) scambia messaggi di diagnostica e di errore, allo scopo di fornire feedback sulle operazioni di scambio pacchetti e di migliorare le prestazioni della rete.

La specifica del protocollo ICMP e' contenuta nel documento RFC792.

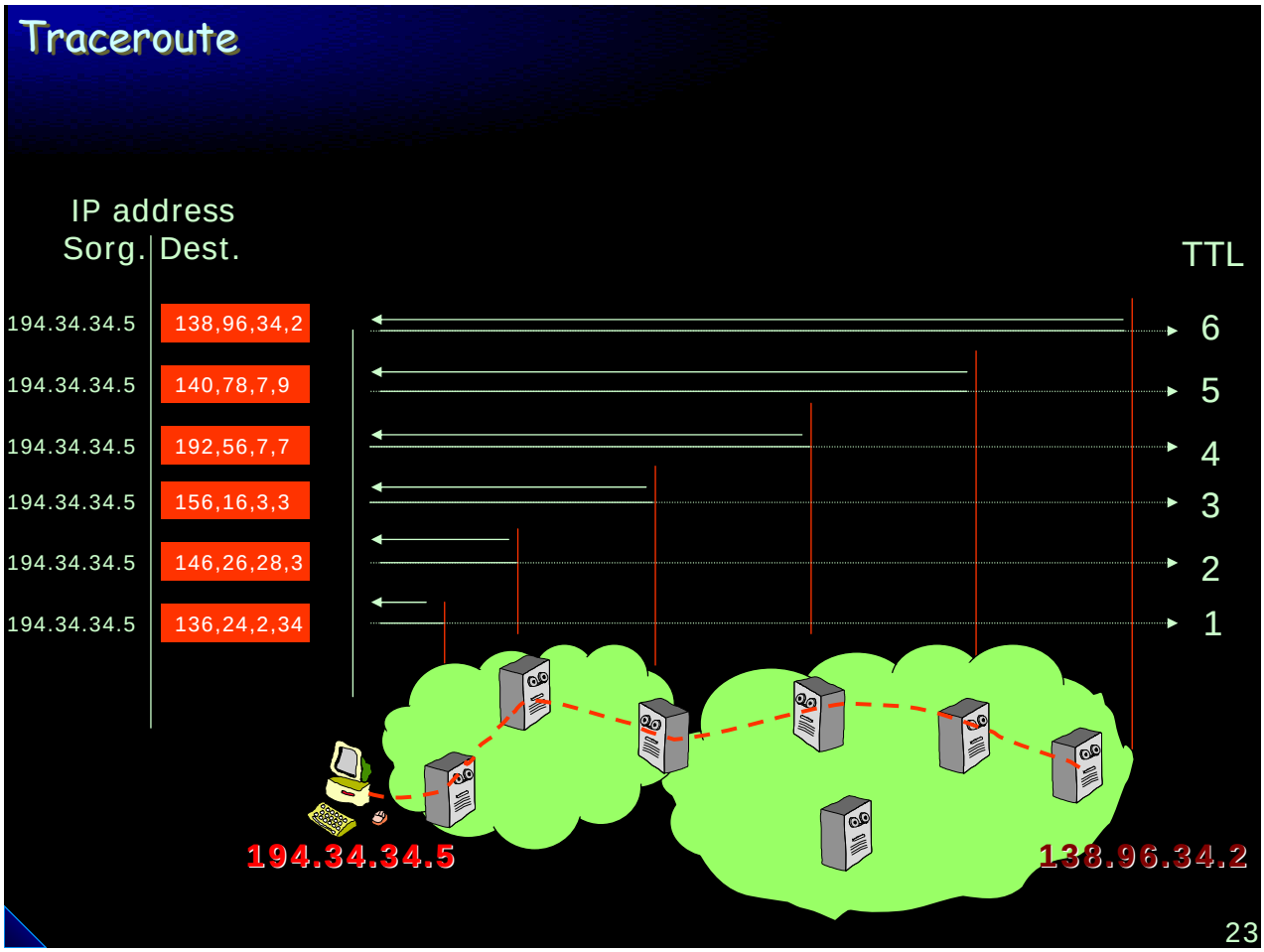
La maggior parte dei messaggi ICMP vengono trasmessi da un router intermedio di percorso alla stazione trasmittente per indicare i motivi dell'avvenuto scartamento di un pacchetto .

E' da notare che in caso di scartamento di un pacchetto ICMP stesso non viene inviato un altro messaggio ICMP, per evitare effetti valanga o loop infiniti. Perciò la presenza di ICMP non garantisce che le stazioni trasmittenti scoprano la perdita di tutti i pacchetti scartati, e il controllo di flusso e la ritrasmissione devono essere affidati a protocolli a livelli più alti.

ICMP è implementato direttamente sopra IP e viene considerato un protocollo necessario della serie protocolli TCP/IP.

Tutti i messaggi ICMP iniziano con un header comune a 32 bit.

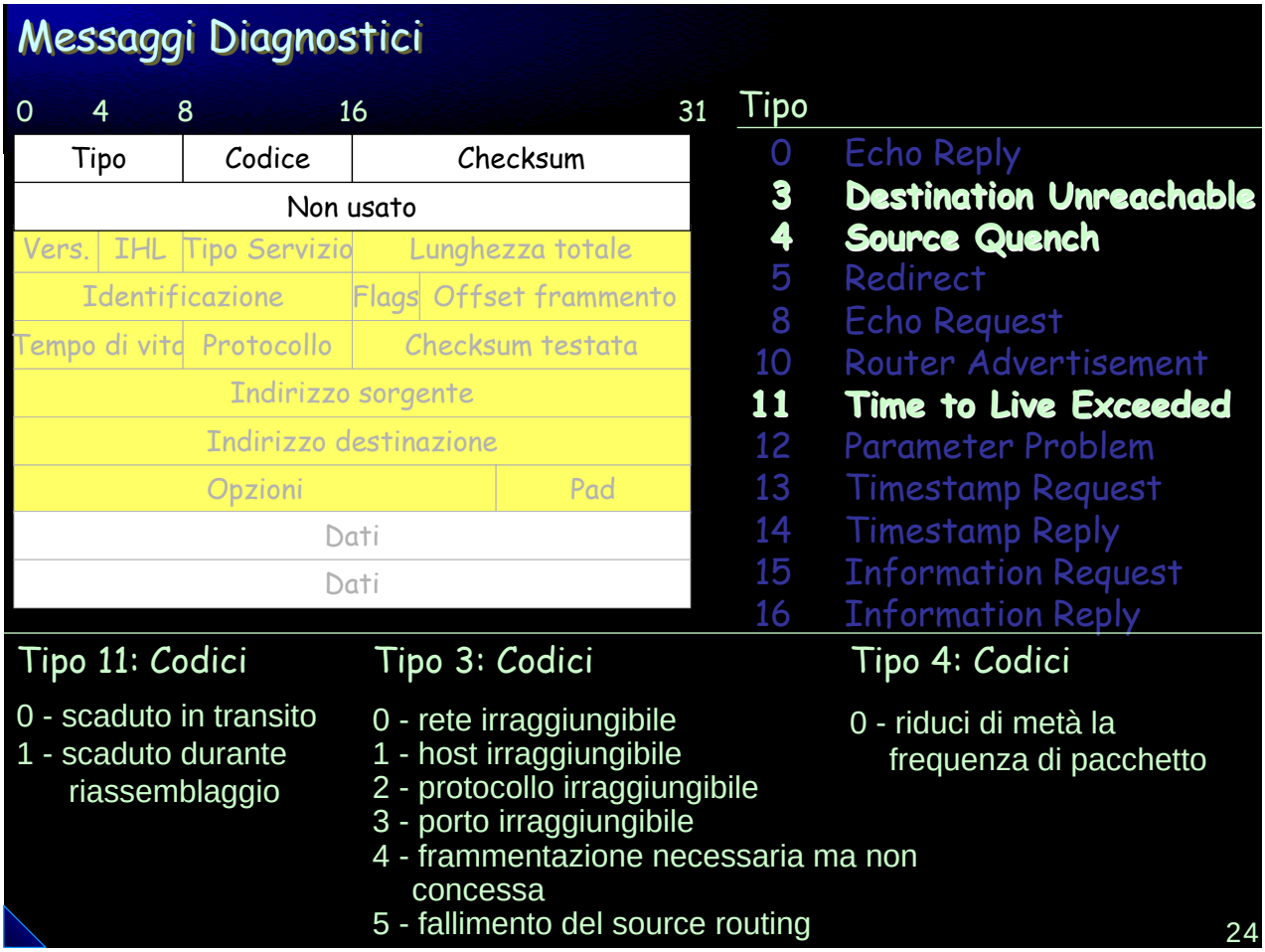
Il campo Checksum è calcolato con lo stesso algoritmo dell'header IP. Il campo Tipo esprime il tipo di messaggio, il campo Codice esprime una variante che dipende dal tipo. Sono definiti molti tipi di messaggi ICMP.



Con il programma traceroute si può realizzare un metodo, basato sui protocolli IP e ICMP che permette di determinare quali sono i router intermedi fra una stazione sorgente ed una stazione di destinazione.

Viene realizzato mediante mediante l'invio di una serie di pacchetti IP contenenti un segmento UDP (protocollo di livello superiore connection less) diretto ad un port non utilizzato, con valore iniziale TTL pari a 1e successivamente incrementato di 1 unità.

In questo modo i router intermedi scarteranno successivamente i pacchetti a causa del valore TTL scaduto (il primo pacchetto con TTL=1 sarà scartato dal primo router, il secondo pacchetto con TTL=2 dal secondo router, e così via) e restituiranno al mittente un pacchetto ICMP (TTL Exceeded). La stazione origine, mediante la registrazione degli indirizzi IP dei router che rispondono con ICMP, può tracciare delle statistiche sul routing.



I messaggi di tipo Time Exceeded, Destination Unreachable e Source Quench denotano tutti un problema operazionale risultante in un pacchetto scartato, e hanno lo stesso formato.

Dopo il campo Non Usato segue una porzione del pacchetto scartato includente la sua intera testata e i primi 64 bit del suo campo dati.

I messaggi **Destination Unreachable** vengono inviati da un router o host finale che non riesce a recapitare a destinazione il pacchetto. Il campo codice descrive l'esatto motivo:

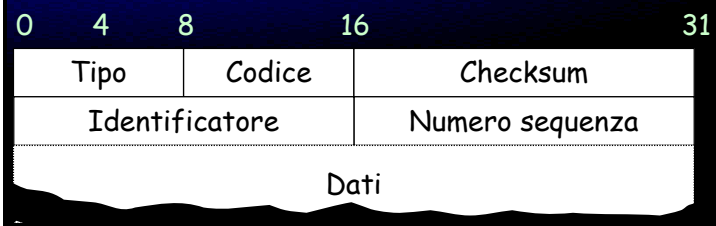
•0 -rete irraggiungibile •1 - host irraggiungibile •2 - protocollo irraggiungibile •3 - porto irraggiungibile •4 - frammentazione necessaria ma bit DF settato •5 - fallimento del source routing

I messaggi **Time Exceeded** vengono inviati quando il campo TTL di un pacchetto e' scaduto. Il codice indica il motivo preciso:

•0 - TTL scaduto in transito •1 - TTL scaduto durante riassemblaggio frammenti

I messaggi **Source Quench** sono inviati da un router in stato di congestione. Il campo Codice e' settato a zero. La stazione ricevente riduce subito (tipicamente alla meta') la velocita' di invio pacchetti, salvo incrementarla gradualmente in tempi successivi.

Ping




```
PING ds.internic.net:
108 bytes from 198.45.45.10: icmp-seq=2.     time=173.  ms
108 bytes from 198.45.45.10: icmp-seq=0.     time=5048. ms
108 bytes from 198.45.45.10: icmp-seq=1.     time=4410. ms
108 bytes from 198.45.45.10: icmp-seq=3.     time=2461. ms
108 bytes from 198.45.45.10: icmp-seq=8.     time=150.  ms
108 bytes from 198.45.45.10: icmp-seq=10.    time=191.  ms
108 bytes from 198.45.45.10: icmp-seq=11.    time=218.  ms
108 bytes from 198.45.45.10: icmp-seq=12.    time=210.  ms
108 bytes from 198.45.45.10: icmp-seq=13.    time=140.  ms
108 bytes from 198.45.45.10: icmp-seq=14.    time=270.  ms
108 bytes from 198.45.45.10: icmp-seq=15.    time=168.  ms
108 bytes from 198.45.45.10: icmp-seq=16.    time=152.  ms
108 bytes from 198.45.45.10: icmp-seq=17.    time=199.  ms
108 bytes from 198.45.45.10: icmp-seq=18.    time=220.  ms
```

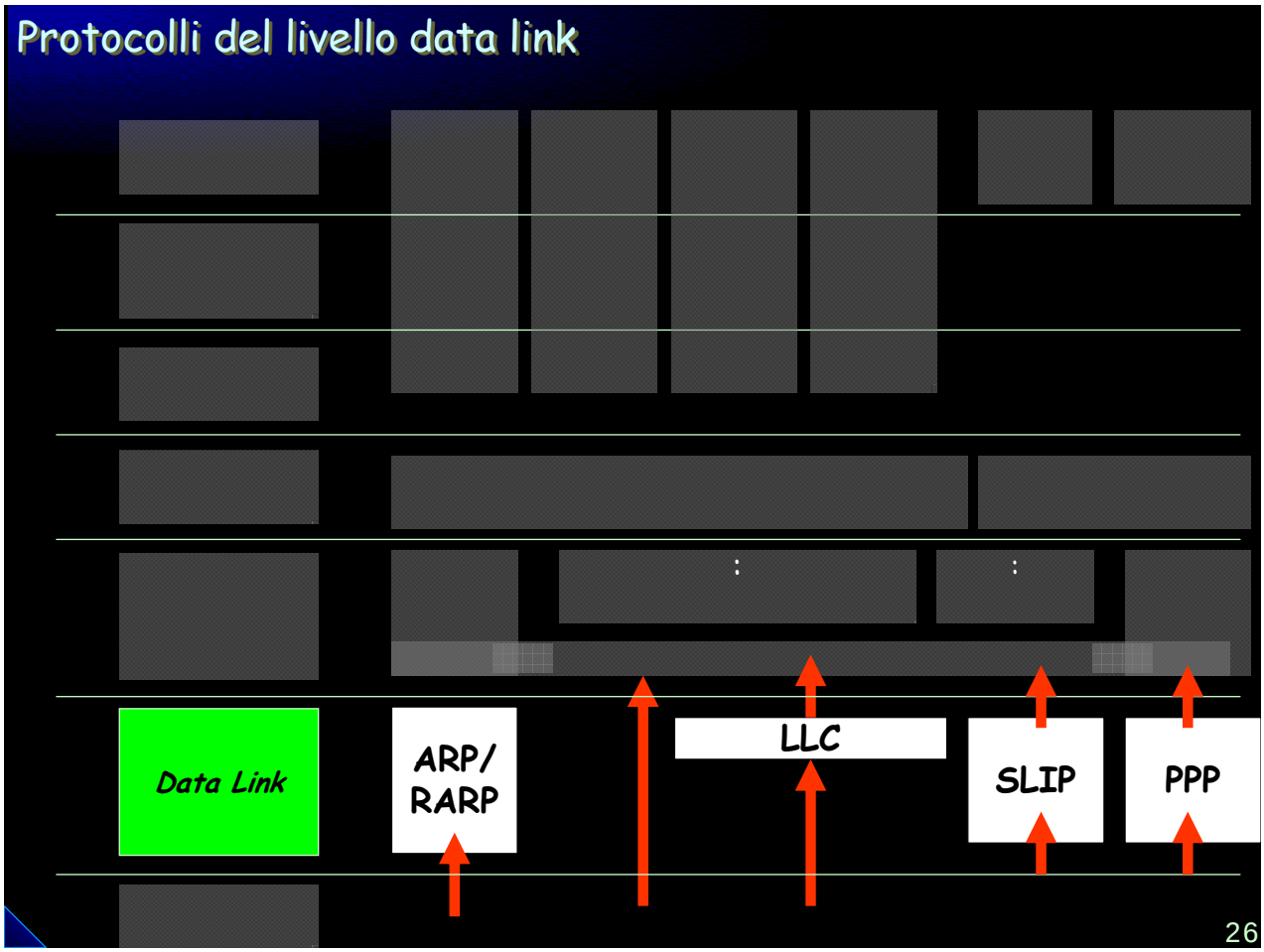
Tipo	
0	**Echo Reply**
3	Destination Unreachable
4	Source Quench
5	Redirect
8	**Echo Request**
10	Router Advertisement
11	Time to Live Exceeded
12	Parameter Problem
13	Timestamp Request
14	Timestamp Reply
15	Information Request
16	Information Reply

25

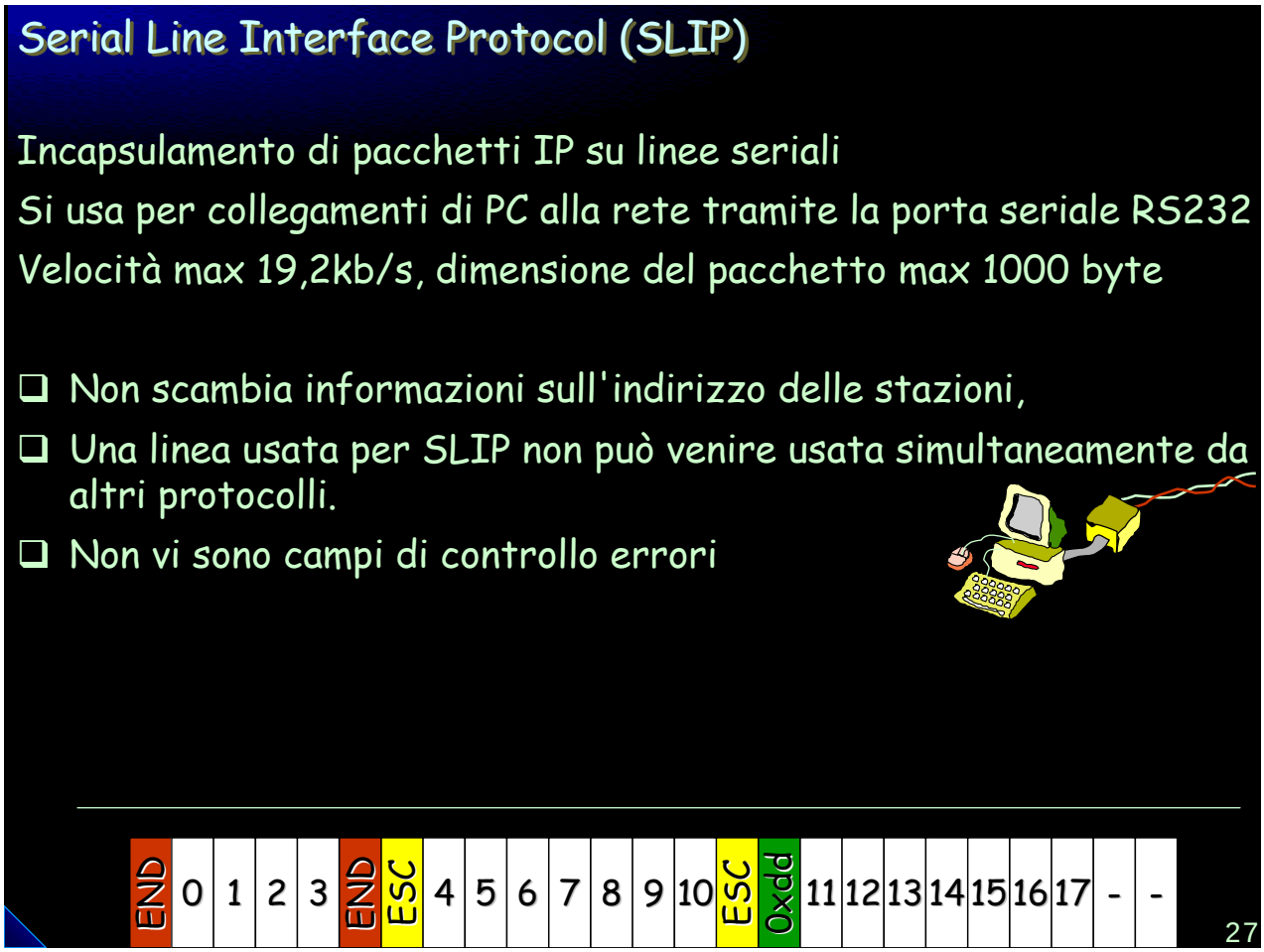
ICMP include una semplice funzione di echo. Quando un router o host riceve un messaggio di tipo Echo Request, risponde con un messaggio Echo Reply. Entrambi i messaggi hanno lo stesso formato.

Nel messaggio Echo Reply sono semplicemente invertiti i campi Indirizzo Sorgente e Indirizzo Destinazione, quindi vengono cambiati i campi Tipo e Checksum.

I messaggi di Echo vengono inviati dal famoso applicativo ping, che testa la raggiungibilita' di una stazione remota ed i tempi di transito, fornendo semplici statistiche.


Protocolli del livello data link
Data Link
ARP/
RARP
LLC
SLIP
PPP
26

Note:



Lo SLIP e' una semplice forma di incapsulamento per datagrammi IP su linee seriali, descritta dal documento RFC1055. E' in uso soprattutto per collegamenti di PC alla rete tramite la porta seriale RS232.

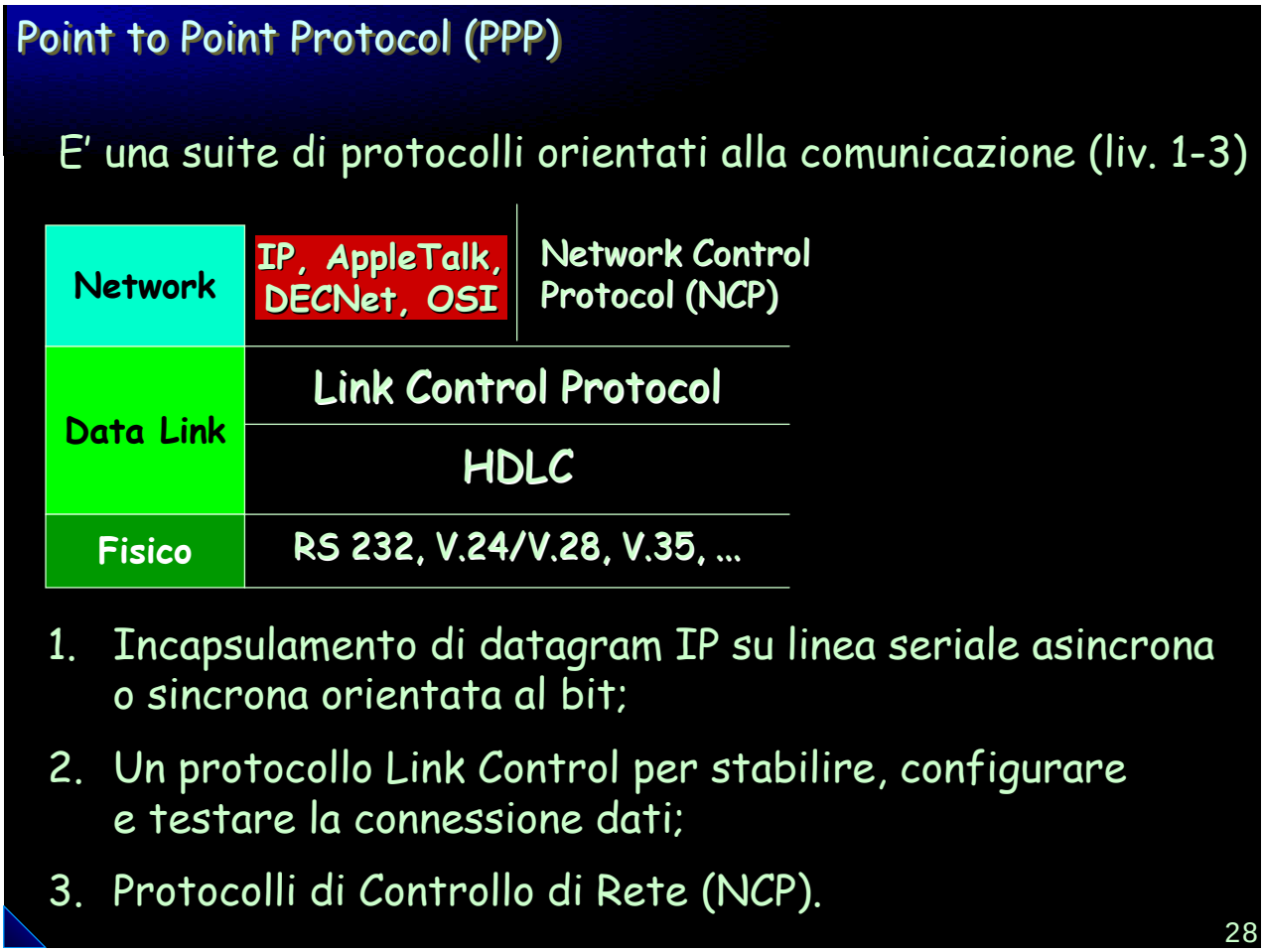
Le regole dello SLIP sono semplici:

• Il datagramma e' prefissato e terminato da un carattere END (0xc0).
• Se un byte interno del datagramma e' 0xc0, e' sostituito dalla coppia di byte 0xdb 0xdc. Il carattere 0xdb viene chiamato ESC.
• Se un byte interno del datagramma e' 0xdb, e' sostituito dalla coppia di byte 0xdb 0xdd.

SLIP ha i seguenti svantaggi:
• Non vi e' metodo per scambiare informazioni sull'indirizzo delle stazioni, tramite il protocollo stesso.
• Non vi e' campo esprimente il protocollo usato. Una linea usata per SLIP non puo' venire usata simultaneamente da altri protocolli.
• Non vi sono campi di controllo errori: il controllo deve avvenire da parte di protocolli a strati superiori.

Slip Compresso

Una nuova versione di SLIP, chiamata CSLIP (Slip Compresso), e' stata introdotta da Van Jacobson e documentata in RFC1144. Opera una compressione dei dati delle testate IP e TCP riducendole fino a volte a tre byte. Inoltre permette di mantenere informazioni sullo stato di fino a 16 comunicazioni TCP simultanee attive.

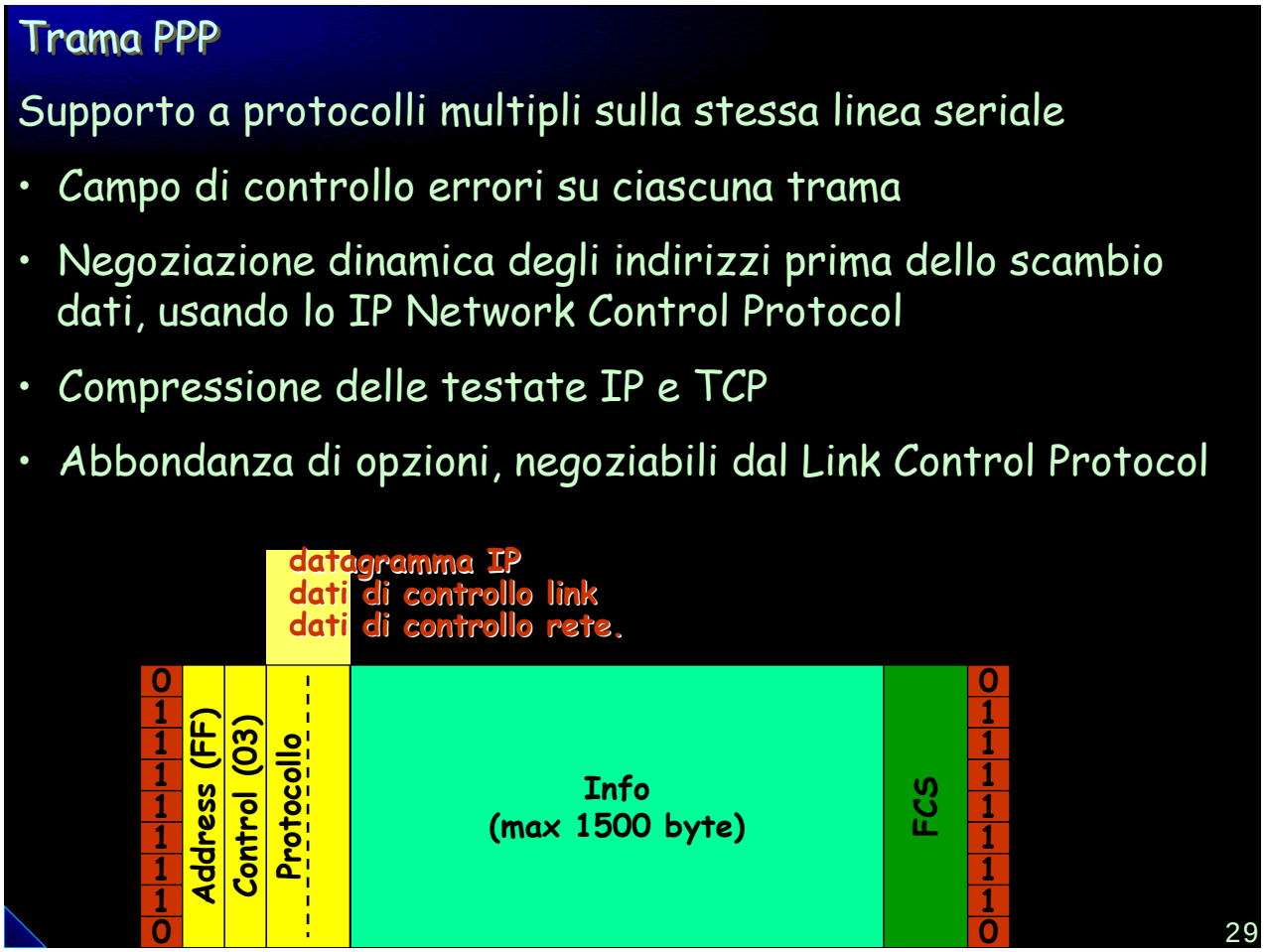


Il protocollo PPP, definito in RFC1331 ed RFC1332, consiste di tre componenti:

1. Incapsulamento di datagrammi IP su linea seriale, asincrona a 8 bit per carattere e senza parita' o sincrona orientata al bit.

2. Un protocollo Link Control per stabilire, configurare e testare la connessione dati

3. Una famiglia di Protocolli di Controllo di Rete (Network Control Protocols - NCP), tra cui protocolli per IP, rete OSI, DECnet e AppleTalk.



Il formato delle trame PPP ricorda quello delle trame ISO HDLC.

I primi tre campi, Flag, Indirizzo e Controllo, sono fissi.

Il campo Protocollo prevede valori diversi per protocolli diversi, tra cui : datagramma IP , dati di controllo link, dati di controllo rete.

Il campo CRC e' un controllo di ridondanza ciclica per il rilevamento degli errori. La trama termina con il carattere di flag con cui e' iniziata.

Comunicazione Asincrona

Il flag di guardia alla trama e' nascosto da un carattere di escape, che lo precede. Anche tutti i caratteri inferiori al decimale 32 sono preceduti dall'escape, per impedire che vengano interpretati come caratteri di controllo della linea seriale.

Nella **comunicazione sincrona** per garantire la trasparenza dei dati si fà uso di bit di stuffing.

Vantaggi di PPP

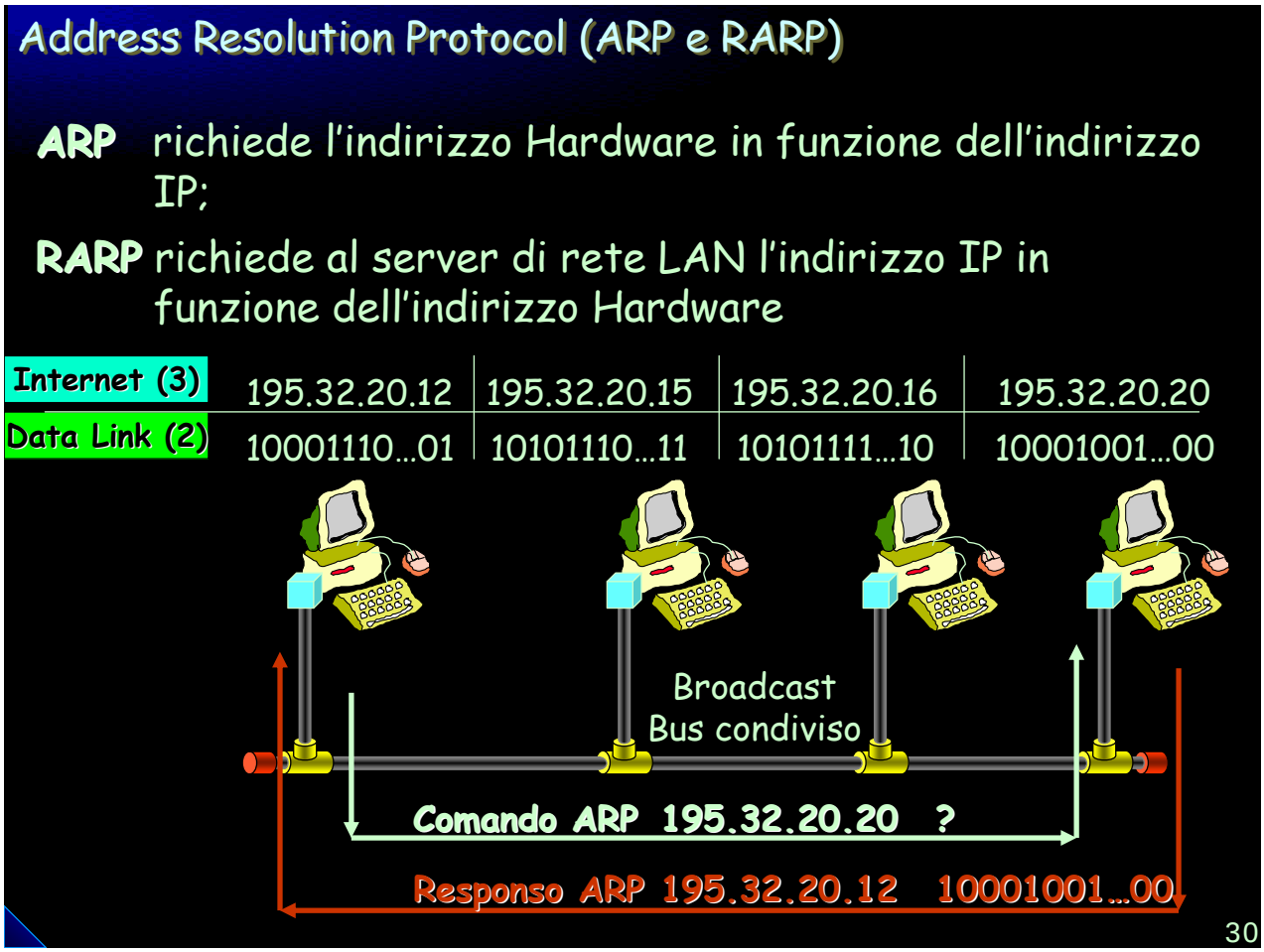
• Supporto a protocolli multipli sulla stessa linea seriale

• Campo di controllo errori su ciascuna trama

• Negoziazione dinamica degli indirizzi prima dello scambio dati, usando lo IP Network Control Protocol

• Compressione delle testate IP e TCP con algoritmo di Van Jacobson, simile a CSLIP. Inoltre molte implementazioni negoziano l'omissione dei campi Flag e Indirizzo e la riduzione del campo Protocollo ad un solo byte.

• Abbondanza di opzioni, negoziabili dal Link Control Protocol



Il protocollo ARP permette di informare gli Host appartenenti ad una stessa rete di tipo broadcast sulla corrispondenza, che deve essere biunivica, fra l'indirizzo internet a 32 bit (livello 3) e l'indirizzo fisico (Mac Address o altro di livello 2).

La sua specifica e' data nel documento RFC826.

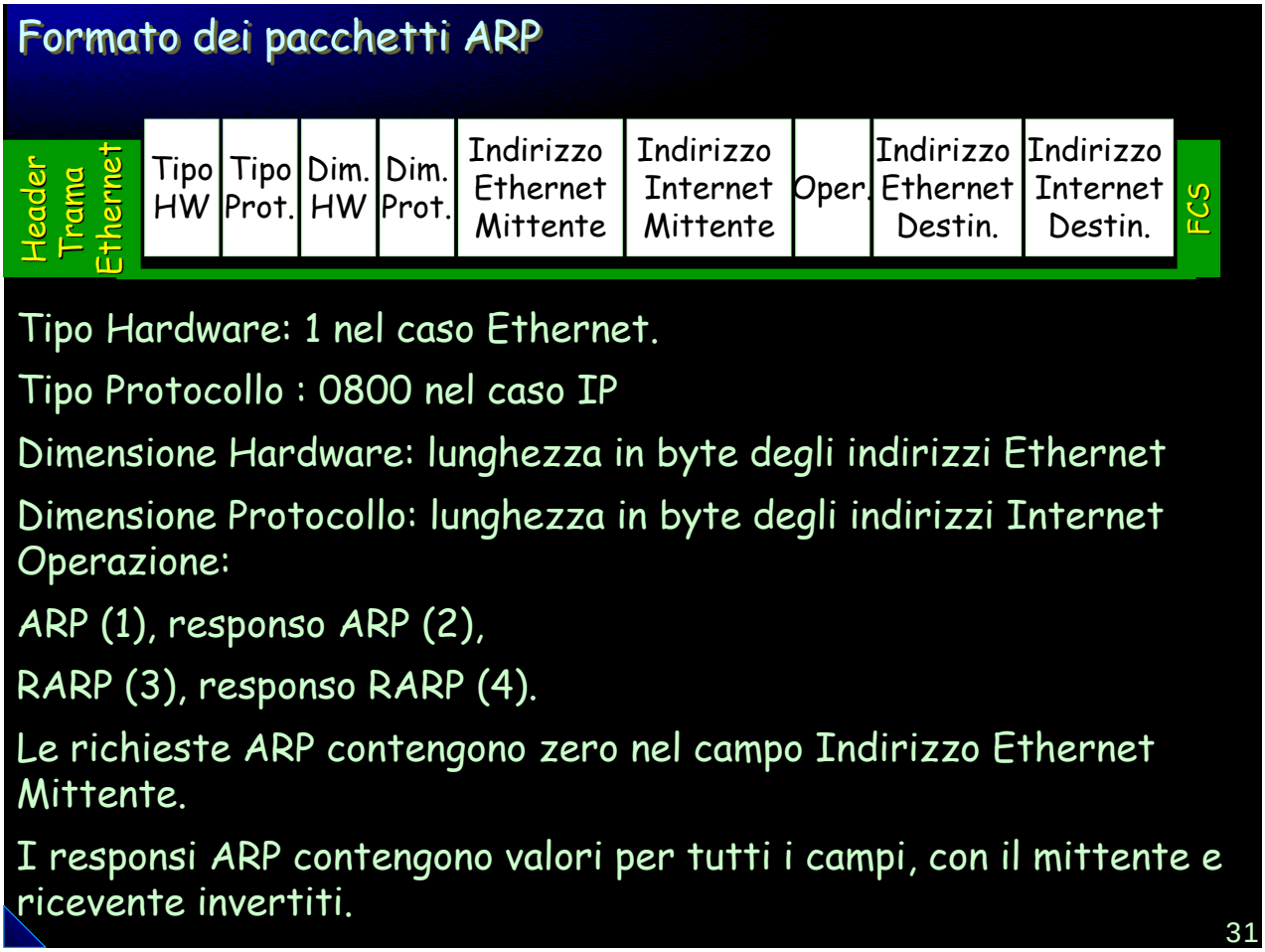
Il caso piu' tipico e' la necessita' di trasformare un indirizzo internet in indirizzo Ethernet (IEEE 802) a 48 bit.

ARP invia un pacchetto di richiesta broadcast contenente l'indirizzo internet di destinazione, e i propri indirizzi internet ed Ethernet. La stazione della rete che riconosce il proprio indirizzo internet nel pacchetto di richiesta invia un pacchetto di responso contenente il proprio indirizzo Ethernet. Reverse Address Resolution Protocol

Se un sistema è senza disco fisso locale non ha modo di registrare in modo permanente il proprio indirizzo internet, quindi all'atto del boot la stazione ottiene il proprio indirizzo fisico Ethernet dall'hardware e poi emette una richiesta RARP alla rete richiedendo il proprio indirizzo internet.

Un'altra stazione della rete, designata come server RARP, risponde con un messaggio di responso.

La specifica di RARP e' il documento RFC903.



Il pacchetto vero e proprio e' preceduto da una testata di trama Ethernet. In questa il tipo di trama e' 0x0806.

Il campo Tipo Hardware vale 1 nel caso Ethernet.

Il campo Tipo Protocollo specifica il protocollo mappato e vale 0x0800 per un indirizzo IP.

I campi Dimensione Hardware e Dimensione Protocollo specificano la lunghezza in byte dei campi seguenti contenenti gli indirizzi Ethernet e IP.

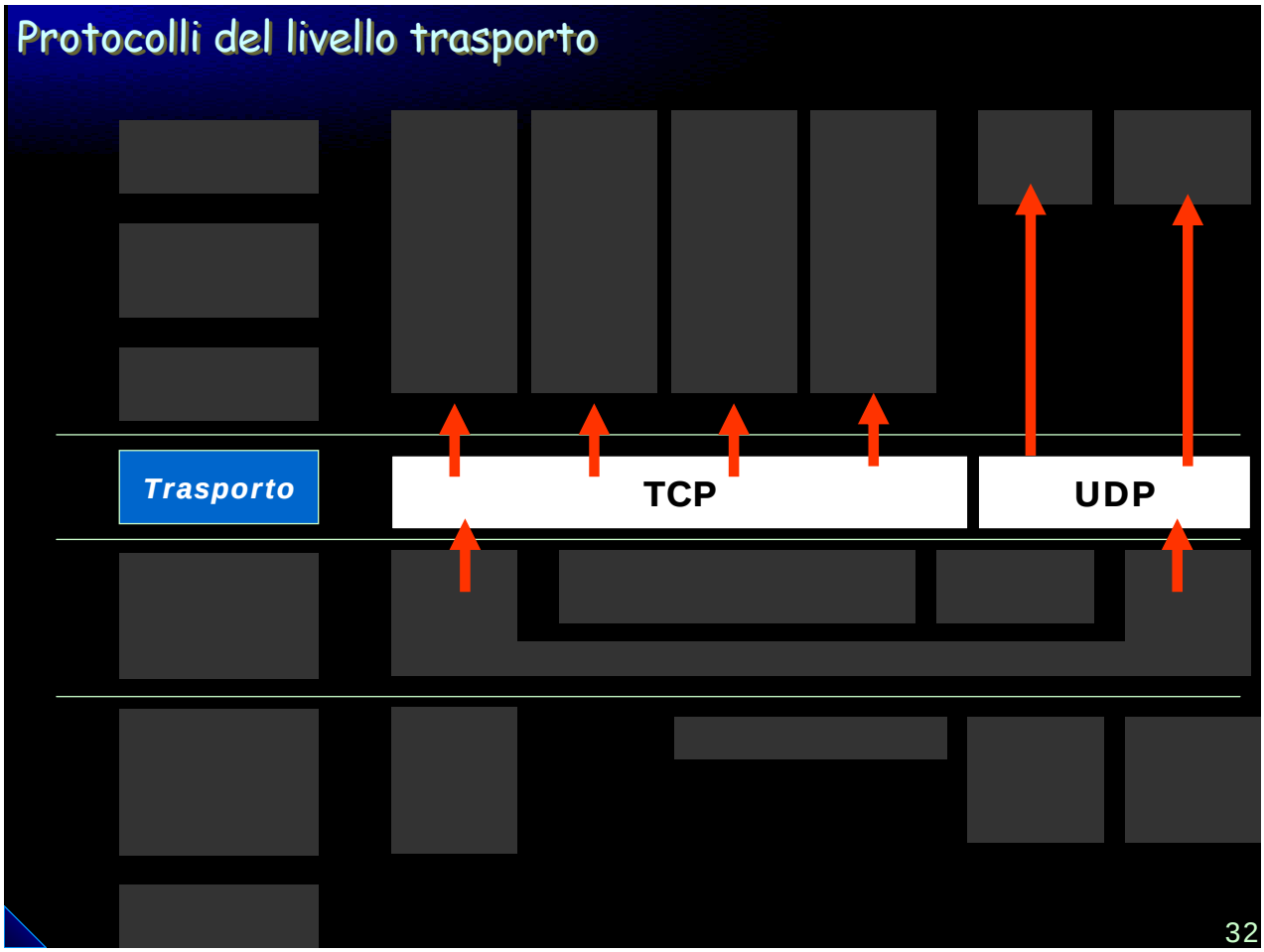
Il campo Operazione specifica se si tratti di una richiesta ARP (1), di un responso ARP (2), o anche di una richiesta RARP (3) o di un responso RARP (4).

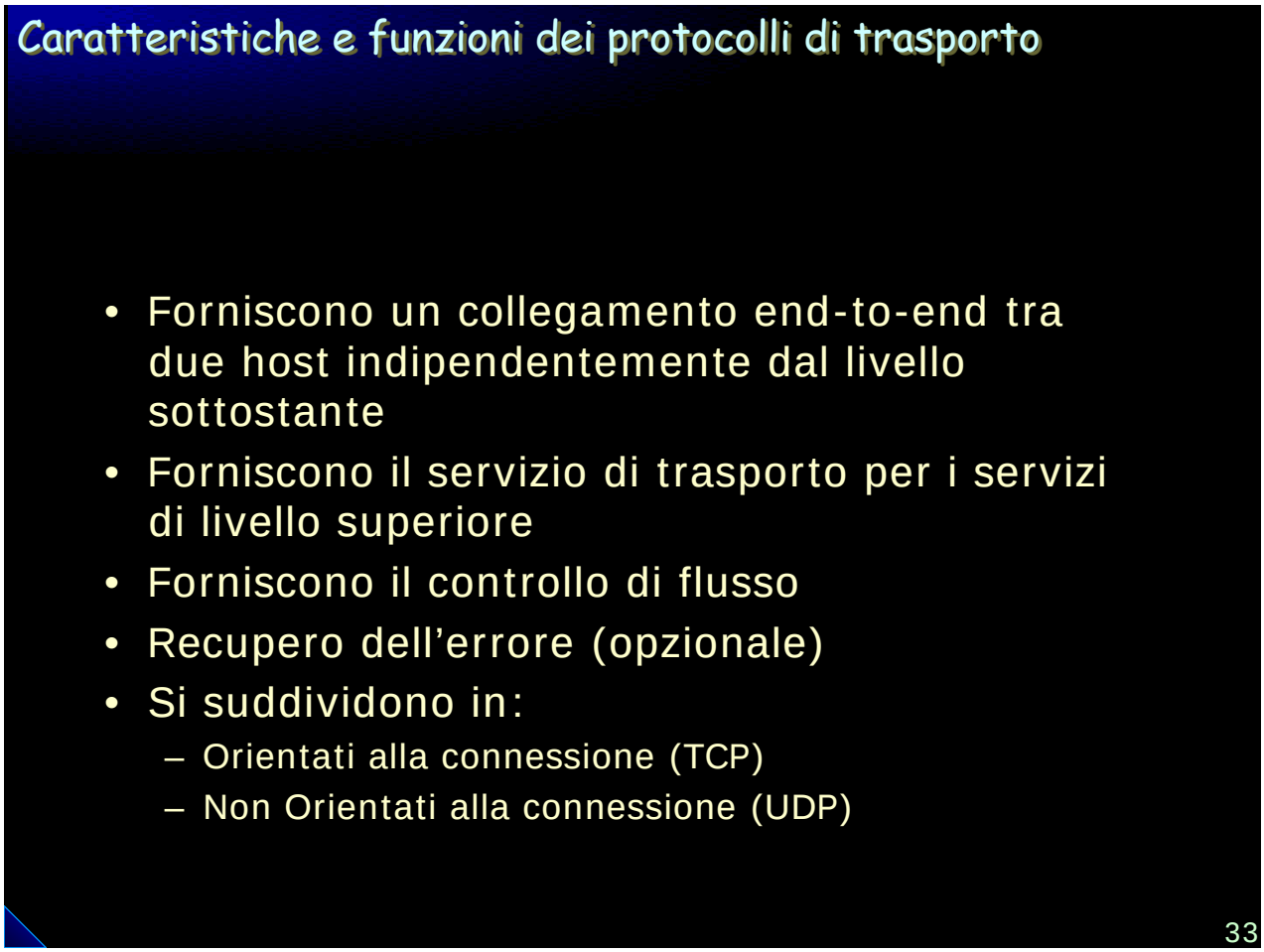
Le richieste ARP contengono zero nel campo Indirizzo Ethernet Mittente. I responsi ARP contengono valori per tutti i campi, con il mittente e ricevente invertiti.

ARP mantiene una cache degli indirizzi risolti, i cui campi diventano invalidi dopo un tempo di 20 minuti.

In caso di fallimento da parte della stazione chiamante di contattare la stazione cercata, ARP crea una entry incompleta nella cache. Il protocollo ARP di per se' non ritenta un contatto fallito. Sono i protocolli a livello superiore (p.es. TCP) che ritentano ad intervalli regolari di contattare una stazione che non risponde, fino ad un loro timeout massimo. ARP viene quindi da essi invocato piu' volte in caso di fallimento iniziale. ARP mantiene la entry incompleta nella cache per 3 minuti.

Il formato dei pacchetti RARP e' quasi identico a quelli ARP, solo che il campo Tipo di Trama vale 0x8035, ed il campo Operazione ha valore 3 per una richiesta RARP e 4 per un responso RARP.


Protocolli del livello trasporto
Trasporto
TCP
UDP
32



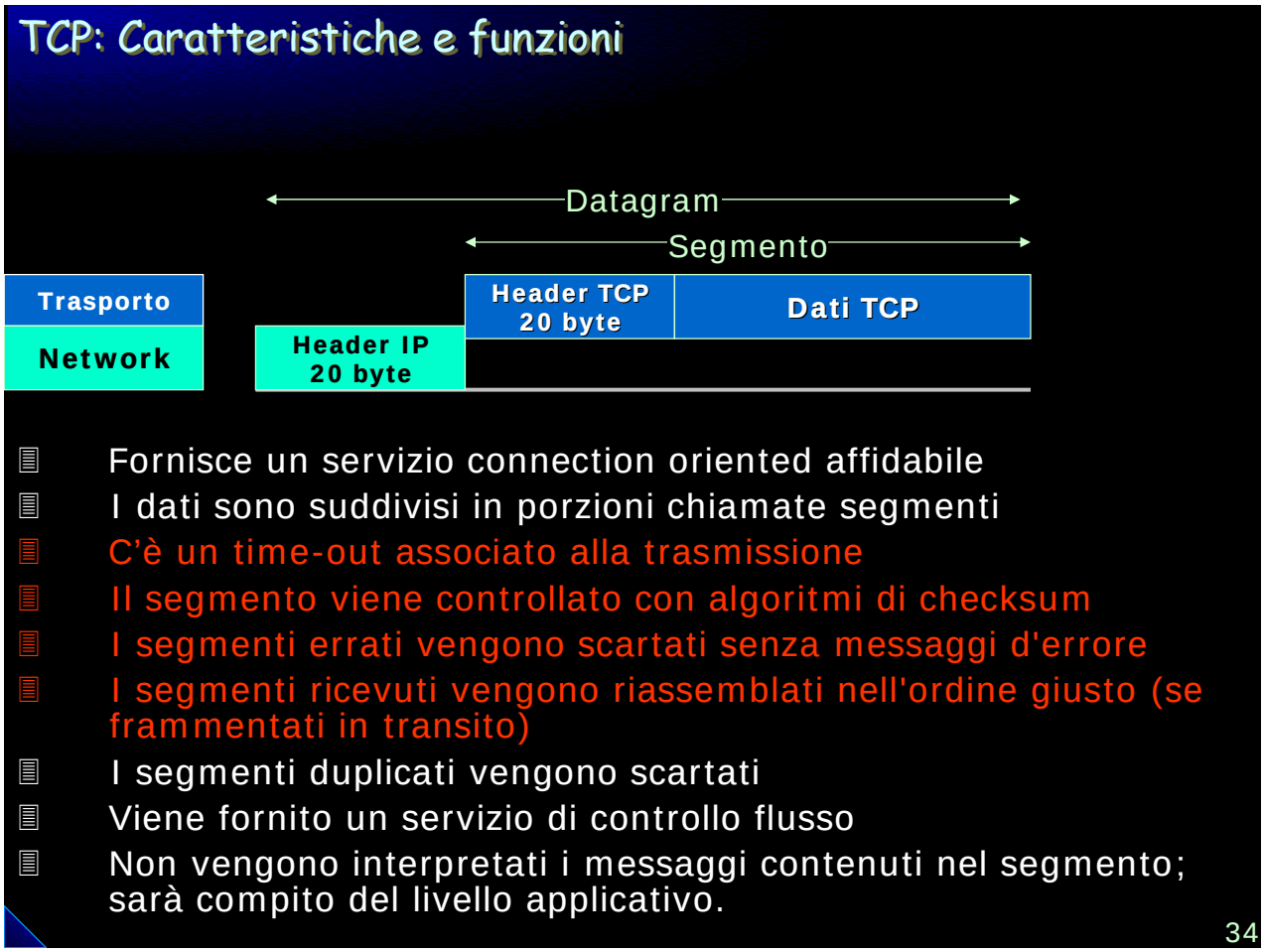
I protocolli di trasporto forniscono un collegamento punto-punto fra due host remoti collegati fra loro. Pertanto il loro utilizzo esula dal tipo di protocolli sottostanti, sia a livello IP che a maggior ragione a livello di accesso. A questi protocolli viene affidato il compito di fornire la struttura di trasporto per particolari servizi di livello superiore, pertanto a seconda dell'applicativo, si utilizzerà un particolare tipo di protocollo di trasporto.

A questi protocolli viene affidato il compito di gestire il flusso di informazioni e se previsto di correggere eventuali errori di trasmissione.

In particolare è possibile dividere i protocolli di trasporto in due grandi sotto-categorie: quelli orientati alla connessione e quelli non orientati alla connessione.

Nel primo caso, a tali protocolli spetta il controllo delle informazioni, accertarsi che il messaggio venga trasmesso e verificarne la ricezione e correzione di eventuali errori. Questo compito si rivela ancora più difficile nel caso in cui la rete sottostante sia non orientata alla connessione (best effort) così come lo è internet.

I protocolli non orientati alla connessione, invece, realizzano la trasmissione del messaggio senza curarsi del fatto che esso venga o meno ricevuto da parte del destinatario.



Il TCP (Transmission Control Protocol) è il protocollo che si occupa del controllo e della preparazione dei dati che gli arrivano dai livelli superiori. A livello di TCP i blocchi di dati si chiamano segmenti.

Tra i compiti del TCP vi è anche quello di multiplare le connessioni, ossia di convogliare in un'unica coda i dati provenienti da diversi applicativi. Il TCP distingue le varie comunicazioni in corso sul computer mediante il numero del port.

La caratteristica più importante del TCP è quella di essere connesso (connection oriented), ossia di stabilire un collegamento con il computer di destinazione, dialogando con esso allo scopo di capire se i dati vengono ricevuti correttamente.

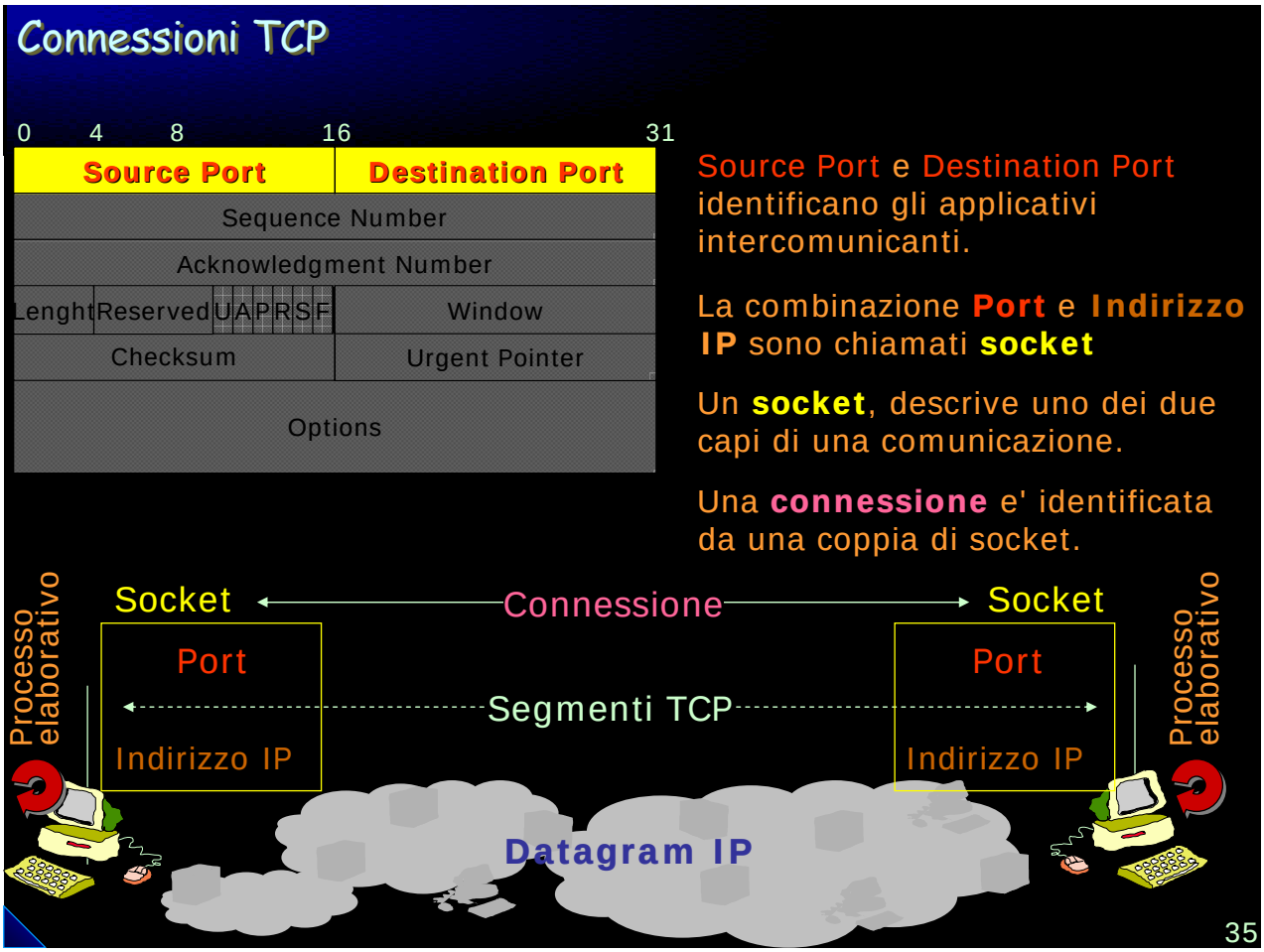
In particolare, la connessione viene sfruttata per effettuare tre tipi di controllo:

- Controllo d'errore,che verifica la correttezza di ciascun segmento a fronte di eventuali modifiche subite lungo il percorso a causa di errori di trasmissione;

- Controllo di sequenza, che verifica in ricezione il corretto ordine di invio dei segmenti. Questo tipo di funzione è necessario in quanto TCP si serve di IP per l'instradamento, il quale non ordina i pacchetti;

- Controllo di flusso, che regola ill controllo della velocità di trasmissione per evitare di congestionare la rete e quindi diperdere dei dati.

Il protocollo TCP fornisce un servizio connection oriented alla comunicazione tra due stazioni. Il documento di riferimento e' RFC793.



I campi Source Port e Destination Port identificano gli applicativi intercomunicanti. Questi due campi, uniti ai campi Source IP Address ed Destination IP Address, identificano univocamente una connessione.

La combinazione **Port** e corrispondente **Indirizzo IP** sono chiamati un **socket**, che descrive uno dei due capi di una comunicazione.

Una connessione è identificata da una coppia di socket.

Alcuni numeri di port standard sono:

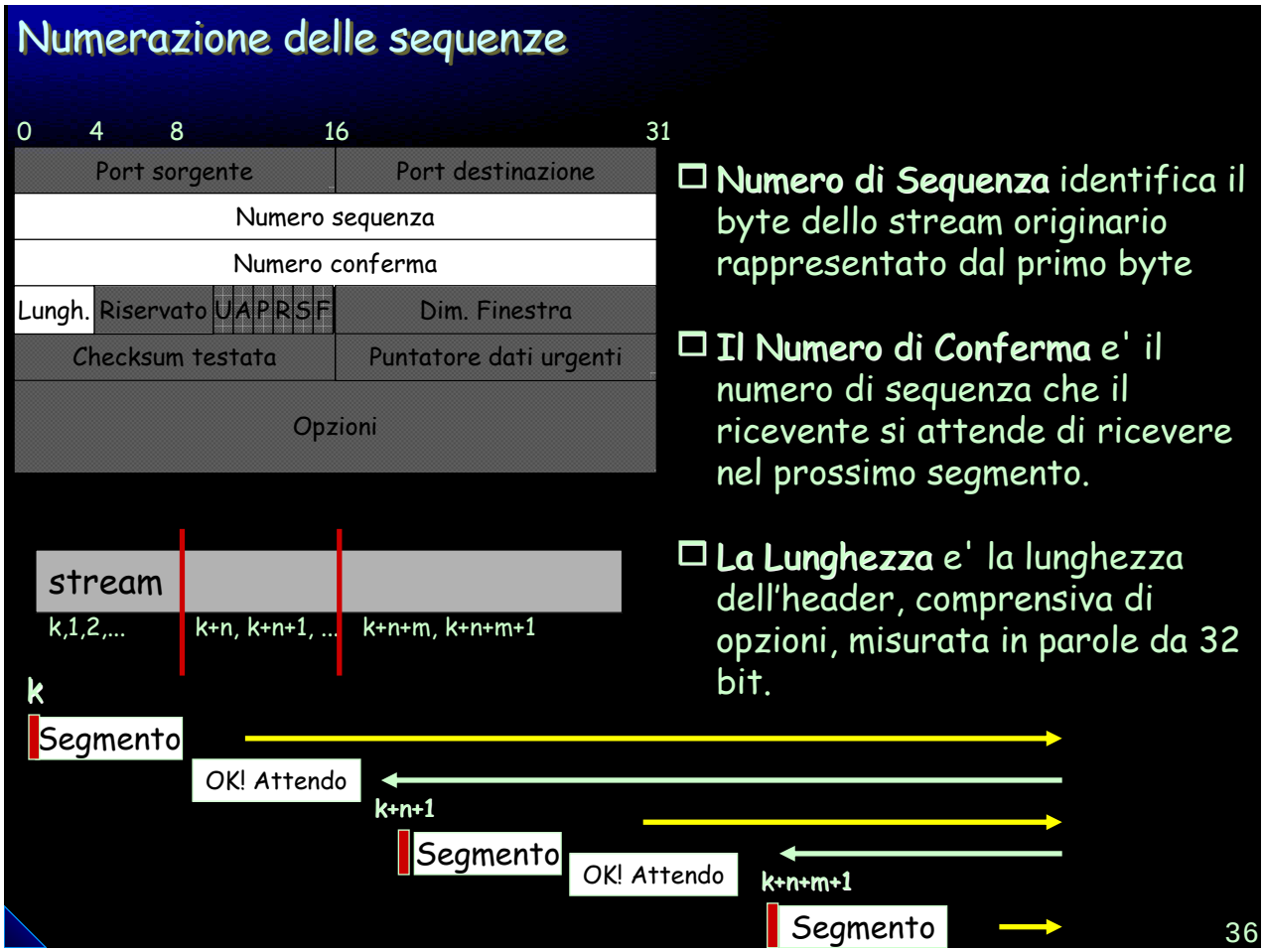
FTP: 21

Telnet: 23

SMTP: 25

HTTP: 80

POP3: 110

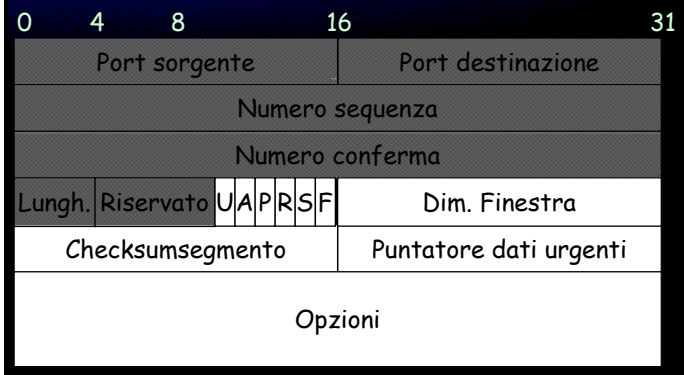


Il Numero di Sequenza identifica il byte dello stream originario rappresentato dal primo byte del segmento corrente. Il numero di sequenza e' di soli 32 bit, quindi i numeri bassi vengono riutilizzati per streams particolarmente lunghi.

Il Numero di Conferma e' il numero di sequenza che il ricevente si attende di ricevere nel prossimo segmento.

La Lunghezza e' la lunghezza della testata, comprensiva di opzioni, misurata in parole da 32 bit. La dimensione massima della testata e' di 60 byte, senza opzioni la lunghezza e' di 30 byte.

TCP: controllo dell'errore e del flusso



Flag:
- **URG** - Il Puntatore Dati Urgenti e' valido
- **ACK** - Il Numero Conferma e' valido
- **PSH** - Il ricevente deve passare queste informazioni all'applicativo nel piu' breve tempo possibile
- **RST** - Reset di connessione
- **SYN** - Sincronizzare i numeri di sequenza per iniziare una connessione
- **FIN** - Il trasmittente ha finito l'invio dei dati

Dimensione Finestra:
lunghezza della finestra di trasmissione in byte.

Checksum:
controllo dell'errore calcolato sull'intero segmento.

Puntatore Dati Urgenti:
offset da aggiungere al Numero di Sequenza per ottenere il numero di sequenza dell'ultimo byte di dati urgenti.

Opzioni:
sono usate in estensioni sperimentali al protocollo TCP.

37

I Flag sono sei:

URG - Il Puntatore Dati Urgenti e' valido; **ACK** - Il Numero Conferma e' valido; **PSH** - Il ricevente deve passare queste informazioni all'applicativo nel piu' breve tempo possibile; **RST** - Reset di connessione; **SYN** - Sincronizzare i numeri di sequenza per iniziare una connessione; **FIN** - Il trasmittente ha finito l'invio dei dati.

Il campo **Dimensione Finestra** esprime la lunghezza della finestra di trasmissione in byte. TCP e' un protocollo a finestra di trasmissione scorrevole senza ritrasmissione selettiva.

Il **Puntatore Dati Urgenti** e' l'offset da aggiungere al Numero di Sequenza per ottenere il numero di sequenza dell'ultimo byte di dati urgenti. TCP offre la possibilita' di indicare dati di emergenza.

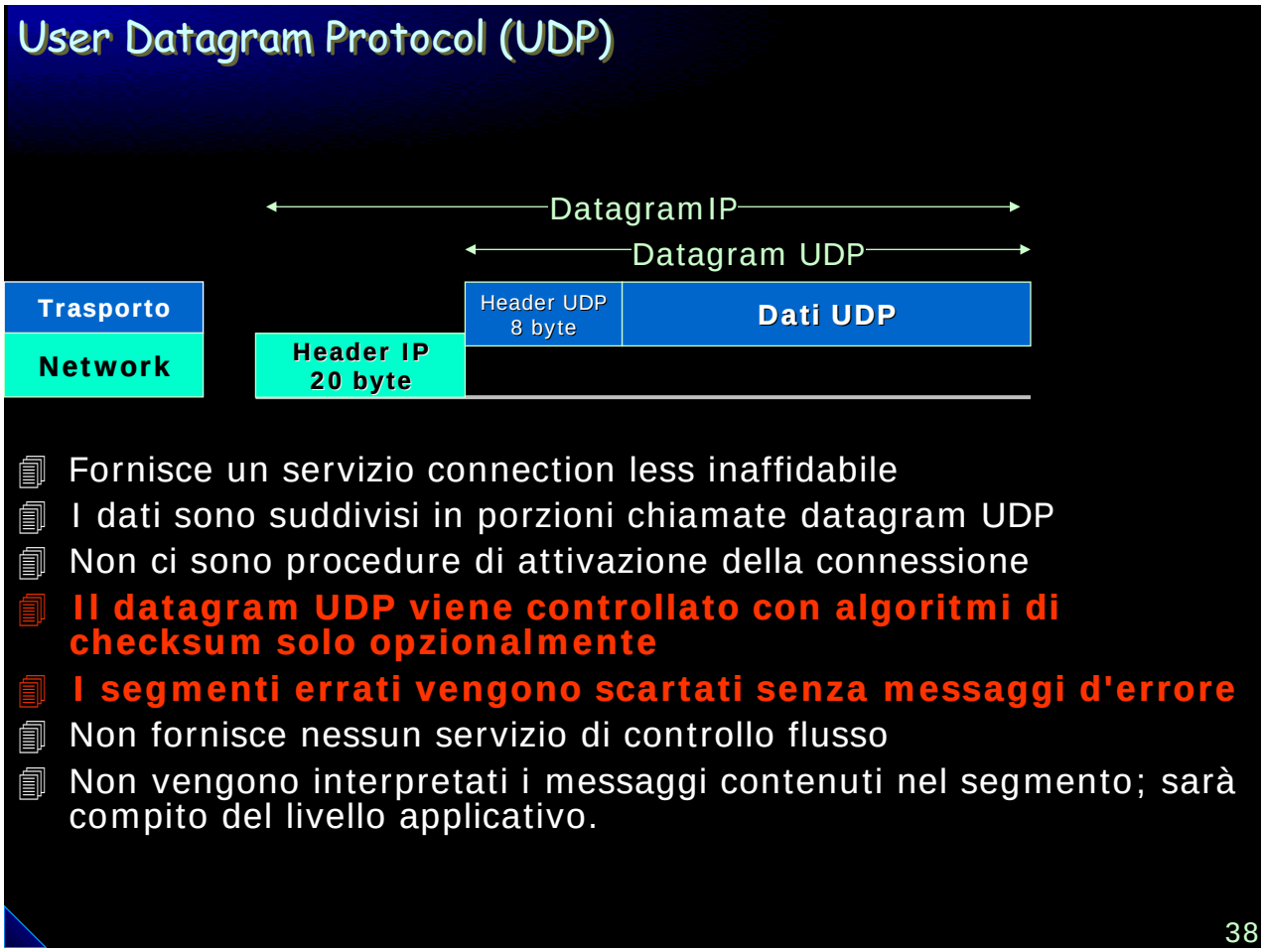
Il campo **Checksum**, che viene controllato solo end-to-end, viene formato in un modo speciale, dovuto al fatto che il protocollo IP ha un campo checksum della sola testata IP.

Innanzi tutto la lunghezza del campo dati e' estesa ad un confine di 16 bit usando se necessario un campo finale di Pad posto a zero. Viene formato uno pseudo-pacchetto composto da:

• una pseudo-testata IP contenente solo alcuni campi dell'header IP originale +

l'header TCP + i dati seguiti dall'eventuale Pad.

Il campo Checksum e' calcolato su questo pseudo-pacchetto come complemento a 1 della somma dei complementi a 1 di tutte le parole a 16 bit del pseudo-pacchetto.

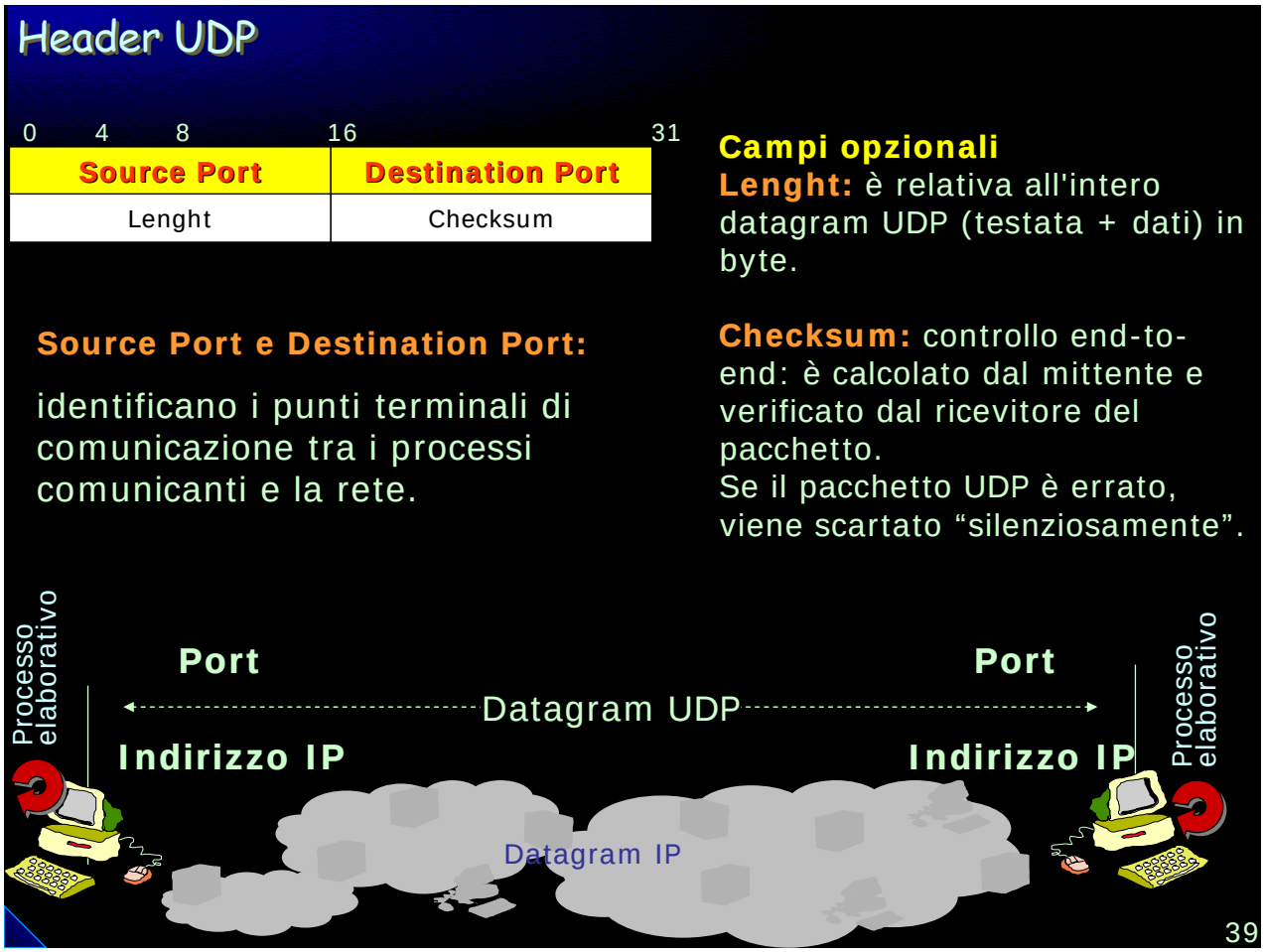


Il TCP è il principale protocollo per la gestione dei messaggi e il controllo della trasmissione, ma non è l'unico: talvolta accade difatti di avere messaggi estremamente corti, come le segnalazioni d'errore o le richieste di conversione DNS da nome a numero IP e viceversa. Per questo tipo di messaggi si può usare un altro protocollo di controllo della trasmissione, che continua peraltro a servirsi del protocollo IP per l'indirizzamento.

Tale protocollo è l'UDP (User Datagram Protocol), che generalmente spedisce messaggi che coincidono con la dimensione del pacchetto IP. UDP si limita ad assegnare al messaggio un numero di porta e una checksum.

UDP non e' affidabile: invia i datagram ma non garantisce che arrivino a destinazione. E' l'applicativo che deve preoccuparsi dell'affidabilità del servizio.

UDP e' descritto dal documento RFC768.



L’header UDP è relativamente semplice.

I campi Source Port e Destination Port identificano i punti terminali di comunicazione tra i processi comunicanti e la rete.

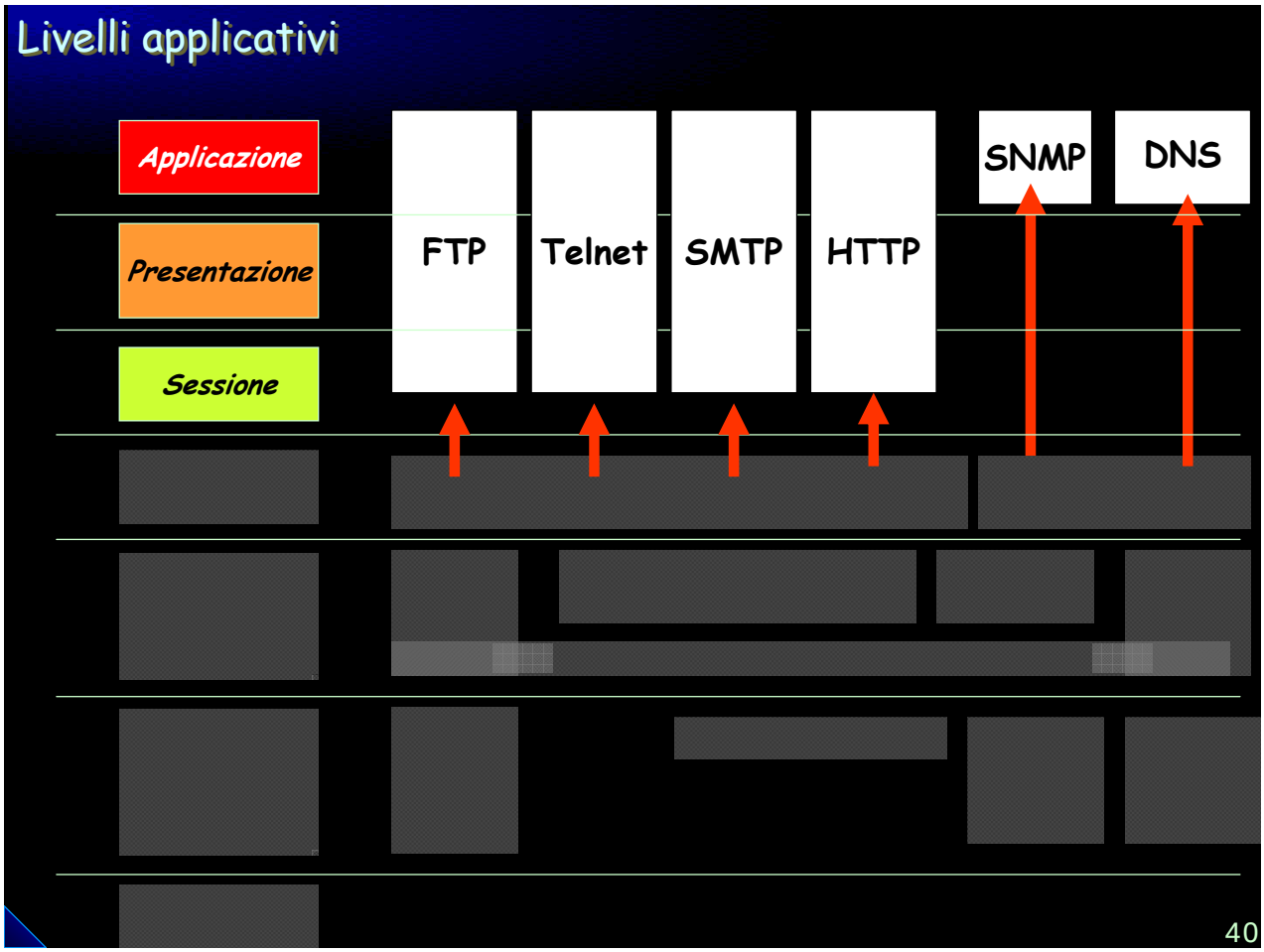
Il campo Lunghezza è dell'intero datagram UDP (testata + dati) in byte.

Il valore minimo è 8, ma in realtà questo campo non viene gestito dal protocollo, che sa che la Total Lenght è la lunghezza di un datagram IP meno i 20 byte dell'header IP.

Il checksum è un campo (opzionale) di controllo end-to-end: è calcolato dal mittente e verificato dal ricevitore del pacchetto. Se il pacchetto UDP è errato, viene scartato silenziosamente, senza generazione di messaggi d'errore.

L’algoritmo utilizzato è come quello per il TCP.

L'opzionalità dei checksum di UDP implica una velocità maggiore delle operazioni se disabilitata, ma può naturalmente fornire sorgenti irrecuperabili d'errori, specie su una rete non Ethernet o quando il datagram UDP transiti da router.



Protocolli di remote Login: il login remoto e' una delle applicazioni interattive di rete piu' comuni. Due applicativi tipici implementano questo servizio cio protocolli TCP/IP:

• rlogin - sviluppato a Berkeley ed inteso originariamente per la connessione tra soli sistemi UNIX, ma ora portato anche ad altri sistemi operativi

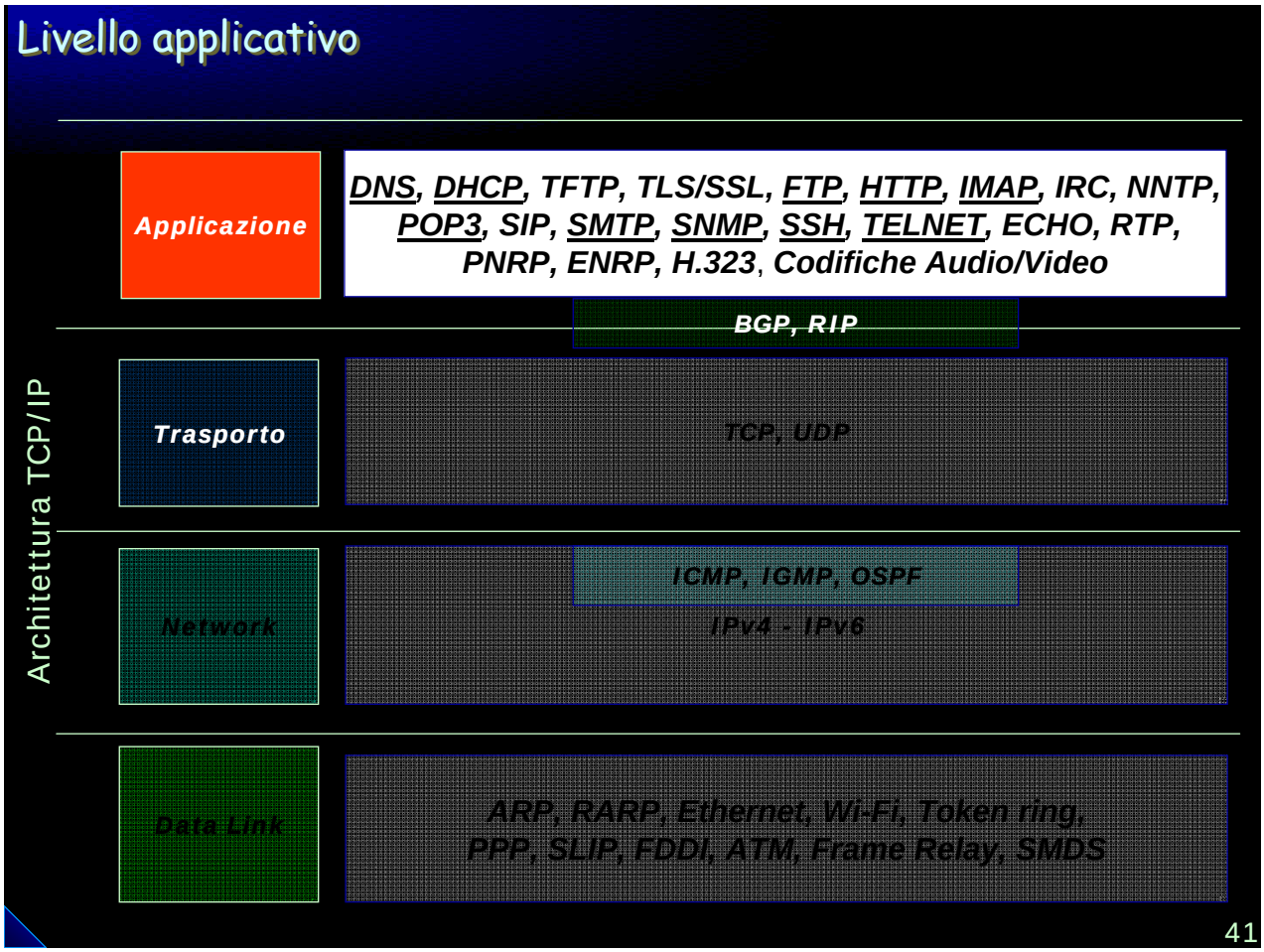
• Telnet - applicazione standard Internet tra qualsiasi sistema operativo

File Transfer Protocol: il protocollo FTP e' lo standard Internet per i file transfer. FTP copia un file completo tra due sistemi. Per usare FTP il client deve compiere un login alla stazione server, con un nome e password validi, oppure puo' essere configurato il server per accettare login anonimo, usando il nome convenzionale di login anonymous o ftp.

FTP e' stato progettato per il lavoro tra host diversi, con diversi sistemi operativi, diverse strutture di file system e anche diversi insiemi di caratteri. Il documento di riferimento di FTP e' lo RFC959.

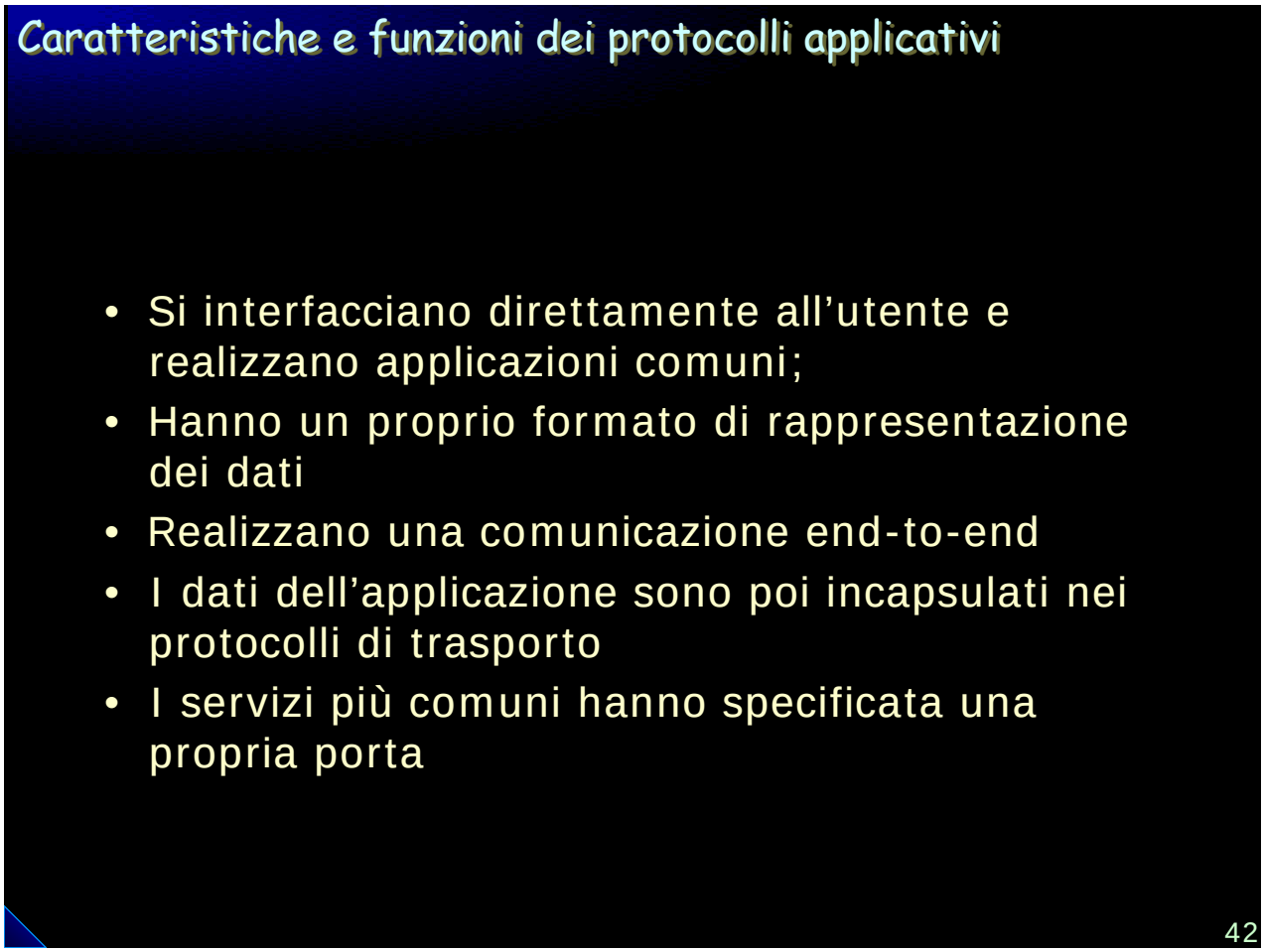
Simple Mail Transfer Protocol: la posta elettronica e' senza dubbio uno degli applicativi; si ritiene che quasi meta' dei pacchetti scambiati in rete appartengano al protocollo SMTP. Il messaggio medio di posta scambiata e' di 1500 byte, ma alcuni messaggi arrivano anche ad alcuni megabyte di dimensione, poichè la posta viene utilizzata come mezzo di file transfer. I documenti di riferimento del protocollo SMTP sono RFC821 e RFC822.

Hyper Text Transfer Protocol: consente la creazione, attraverso il formato HTML (Hyper Text Markup Language), di collegamenti ipertestuali tra documenti diversi; il protocollo HTTP è alla base del World Wide Web.



Analizziamo i protocolli sottolineati.

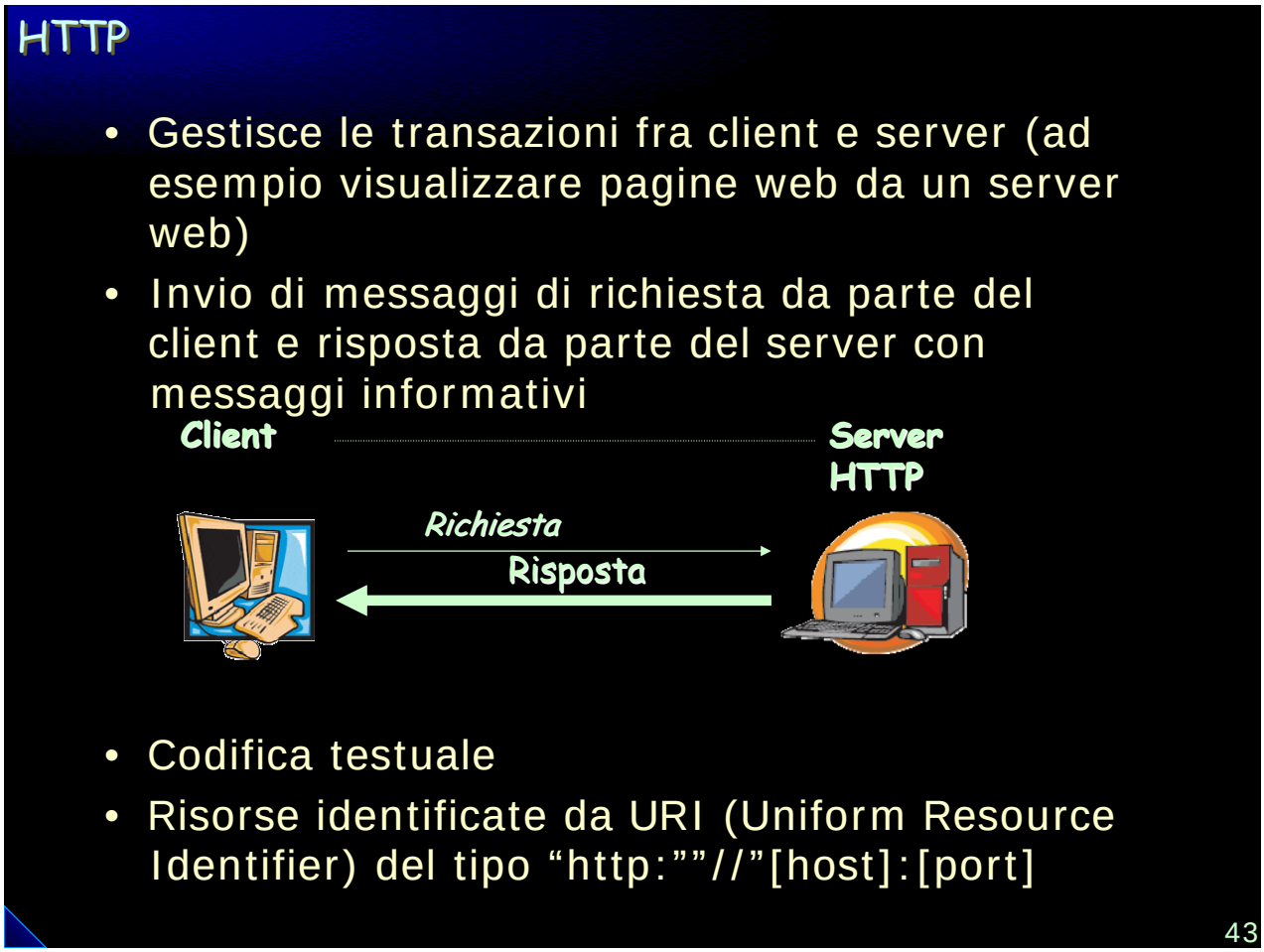
Cenno sulle codifiche audio/video PCM-H.264



Il livello di applicazione viene utilizzato per la comunicazione fra reti. I dati vengono elaborati dall'utente finale e passati al livello sottostante con uno specifico formato.

Ovviamente né router, nè switch vedono questo livello, che come detto per il livello di trasporto realizza una comunicazione end-to-end tra gli host

Nell'architettura TCP/IP alcuni servizi di livello applicativo sono legati al protocollo di trasporto da utilizzare. A tal proposito i protocolli di trasporto più utilizzati sono TCP/UDP. Le applicazioni più comuni hanno una specifica porta assegnata

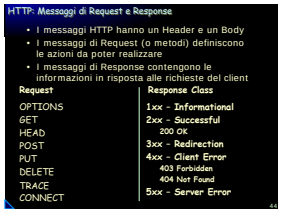


Seppur l’acronimo HTTP (Hyper-text transfer Protocol) induca a pensare che questo protocollo svolga il compito di trasferire degli ipertesti, in realtà ha l’obiettivo di gestire le transazioni tra due host. Dove alla richiesta da parte di un client, segue una risposta da parte del server e la conseguente apertura della connessione tcp tra i due punti.

La codifica di tale protocollo è testuale, e i pacchetti di questo protocollo possono essere facilmente interpretati.

Per poter identificare una risorsa all’interno di una rete che utilizza il protocollo HTTP risulta necessario utilizzare un meccanismo che la renda univoca. Per fare ciò si utilizzano i cosiddetti URL – Uniform Resource Location oppure URI Uniform Resource Identifier. Con tali stringhe testuali è possibile accedere ad una risorsa mediante il protocollo http. La struttura che ha una URI http è del tipo http: (definito schema) seguito da //, dopo di che viene indicato il nome dell’host al quale si vuole effettuare l’interrogazione (indirizzo IP o nome come vedremo grazie al DNS) e in maniera del tutto opzionale viene indicato anche il numero della porta di utilizzo.

Dare una nota sul SIP (che ha la stessa struttura)



In questo lucido si andranno brevemente ad analizzare i tipi di messaggi che vengono utilizzati in HTTP. In linea del tutto generale i messaggi http contengono un header e un body. Nel primo sono inseriti i campi che sono necessari per lo specifico messaggio che lo caratterizza dagli altri, mentre nel corpo sono contenuti i dati veri e propri da trasferire.

I messaggio http si dividono in due grandi sottocategorie, come precedentemente anticipato: i messaggi di request ai quali si risponde con i messaggi di response. Così come ogni altro protocollo vengono stabilite delle regole per realizzare la comunicazione tra due punti distinti.

Per quanto riguarda i metodi di request, nella RFC del protocollo HTTP 1.1 vengono descritti 8 diversi metodi divisi nelle categorie “sicuri” che consentono agli implementatori di conoscere gli effetti indesiderati del metodo e “idempotenti” che forniscono lo stesso risultato sia se vengono eseguiti più volte che una sola volta. Alla prima categoria appartengono i metodi GET e HEAD, mentre GET, HEAD, PUT e DELETE sono idempotenti e inoltre anche OPTIONS e TRACE sono intrinsecamente idempotenti in quanto non producono effetti nell’altro lato della comunicazione.

Passiamo ad una breve descrizione dei messaggi di request.

Il metodo OPTIONS consente la richiesta di informazioni relative alle opzioni di comunicazione sulla catena richiesta/risposta identificata dalla URI. In tal modo il client è in grado di determinare le opzioni o le richieste associate ad una risorsa oppure la capacità del server senza dover mettere in piedi un sistema di conoscenza della sorgente.

Il metodo GET assume differenti significati a seconda del ruolo svolto dalla URI che lancia la richiesta (Request-URI). Se l’informazione (inviata sotto forma di entità) è identificata dalla URI allora questo metodo ricava delle informazioni. Se la URI richiedente si riferisce ad un processo che produce dati, allora saranno i dati prodotti che verranno spediti (sotto forma di entità) in risposta e non il testo sorgente (anche se il risultato può essere sotto forma di testo). (vedi riferimento con html, dove questo metodo nei link viene utilizzato per ricavare informazioni da un altro processo di elaborazione come ad esempio una interrogazione dal database)

Il metodo HEAD è identico al GET tranne per il fatto che il server non deve ritornare un body nel messaggio di risposta. Pertanto le informazioni in risposta ad un messaggio di HEAD devono essere le stesse di un messaggio di GET. Questo metodo si rende utile per verificare le informazioni delle entità impiegate senza dover trasferire il corpo del messaggio.

Il metodo POST viene usato per richiedere se il server accetti l’entità racchiusa nella richiesta come una subordinata identificata dalla URI richiedente nella Request-Line. Questo metodo svolge le seguenti funzioni

•Annotare le risorse esistenti;

•Inviare un messaggio ad una mailing list o simile;

•Fornire un blocco di dati, come risultato del forma da inviare ad un elaboratore di questi dati (come viene fatto con il metodo post nel linguaggio HTML, dove si inviano su delle variabili tutte le informazioni digitate e contenute nei form di origine);

•Estendere un database con una operazione di append.

I dati “postati” sono subordinati comunque alla URI richiedente e i dati inviati sono elaborati secondo le specifiche dal server.

Il metodo PUT richiede che l’entità racchiusa venga immagazzinata nella stessa URI richiedente.

Il metodo DELETE chiede che il server di origine cancelli le risorse identificate dalla URI richiedente.

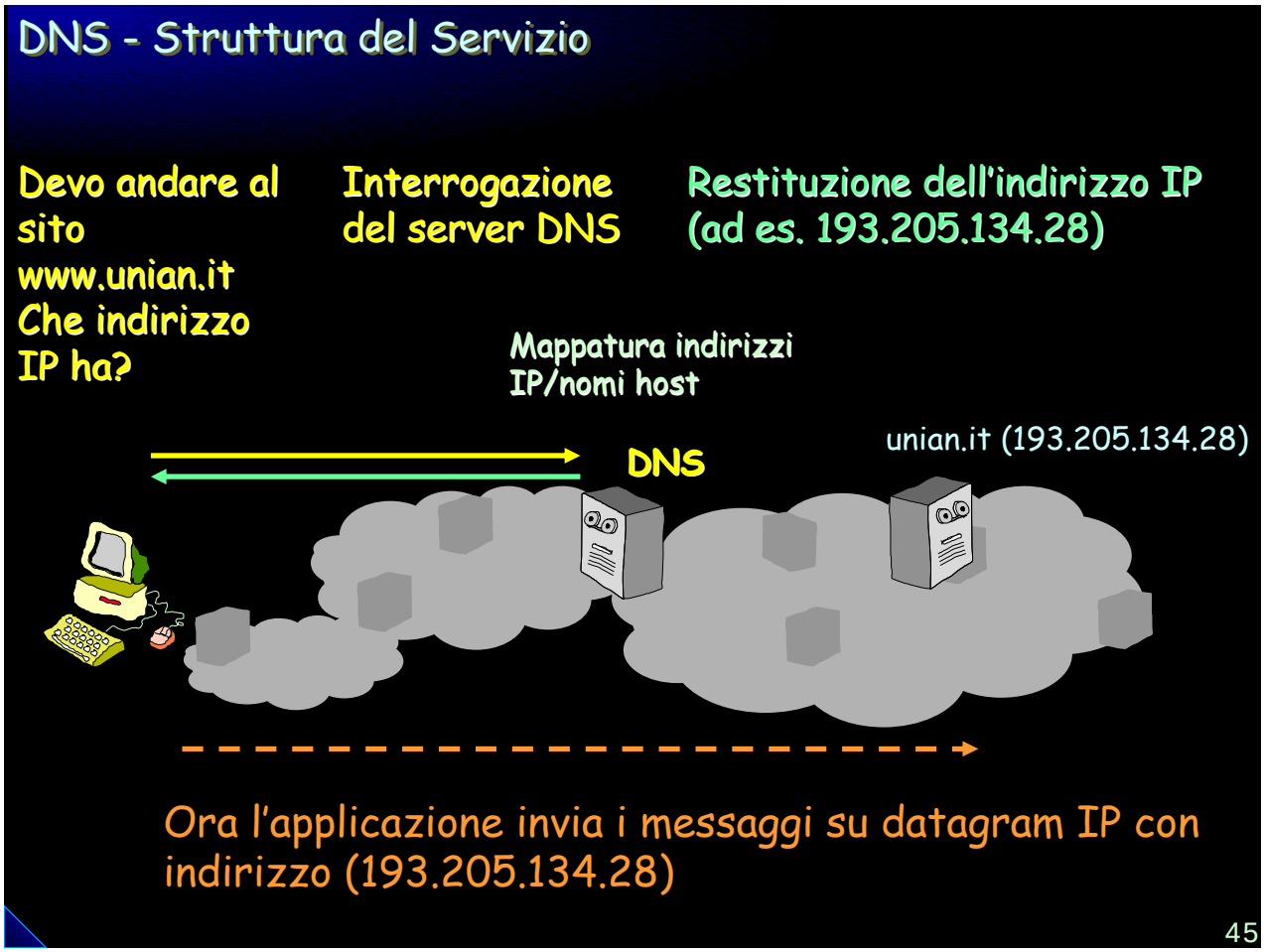
Il metodo TRACCE viene utilizzato per inviare un messaggio remoto di loop-back per il livello applicativo. TRACE non deve contenere delle entità (per cui non ci sono effetti provocati nei due lati trasmissivi).

Il metodo CONNECT viene riservato nello standard per l’uso con un proxy o con un tunnel (connessione diretta fra due host di rete)

Passiamo, infine alla descrizione dei messaggi di risposta. Questi messaggi sono numerosi e divisi in cinque classi così come illustrato nel lucido.

Ad ogni codice di risposta viene associato, inoltre, una frase al fine di ricordare meglio il significato. Questa frase è di solo ausilio per gli uomini in quanto le macchine elaborano direttamente i codici non badando alla frase ad essi collegata.

La classe 1xx contiene esclusivamente campi di header ed ha lo scopo di fornire informazioni sullo stato del

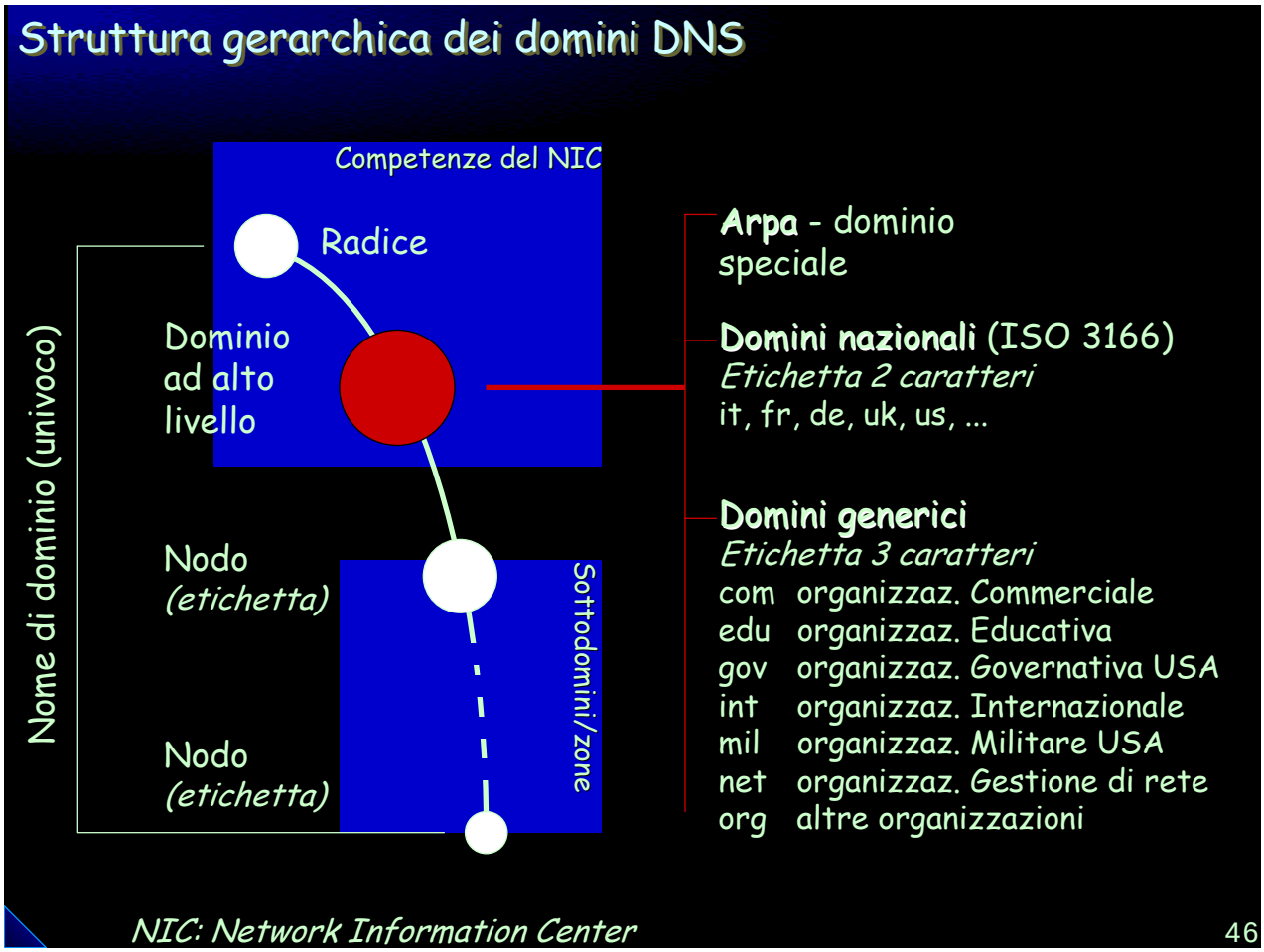


Il Domain Name System (DNS) e' un database distribuito usato dagli applicativi TCP/IP per la mappatura da **nomi host** ad indirizzi di **Internet**.

Non esiste un singolo sito in Internet che conosca tutte le informazioni del database, invece ogni sito mantiene la sua porzione di database limitata alla descrizione del sito, e gestisce un server di questa porzione di database, che puo' essere interrogato da altri sistemi in Internet.

E' compito degli applicativi convertire il nome host in indirizzo IP prima di invocare i protocolli TCP o UDP.

I concetti ed i dettagli implementativi del DNS sono contenuti nei documenti RFC1034 ed RFC1035. L'implementazione DNS piu' comune per UNIX si chiama BIND - Berkeley Internet Domain Server.



Lo spazio dei nomi del DNS e' gerarchico ed implementato in una struttura ad albero. Ogni nodo ha un'etichetta di lunghezza massima 63 caratteri. La radice e' un nodo speciale senza etichetta.

Il nome di dominio e' la sequenza di etichette a partire dal nodo verso la radice, usando il carattere 'punto' (.) come separatore di etichetta.

Ogni nodo dell'albero DNS deve avere un nome di dominio univoco.

Un nodo dell'albero direttamente sotto il nodo radice si chiama dominio ad alto livello (top-level domain). Questi sono divisi in tre aree:

1. arpa - dominio speciale usato per le mappature da indirizzi IP a nomi

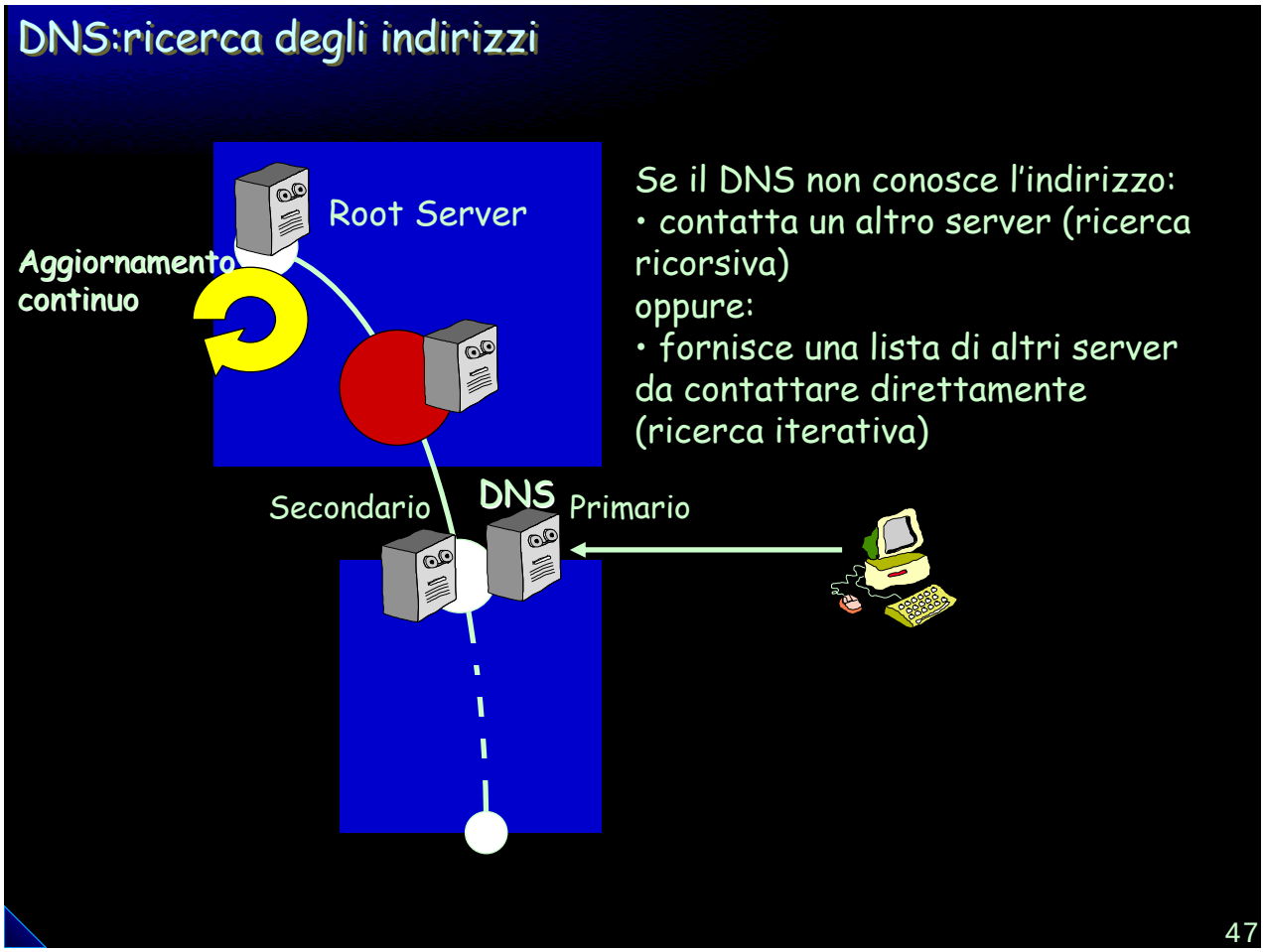
2.domini generici - sono 7 e sono identificati da codici a tre caratteri

3.domini nazionali - sono a due caratteri e definiti dallo standard ISO 3166.

Non vi e' nessuna implicazione che i domini generici siano riservati agli Stati Uniti, e difatti esiste anche il dominio nazionale US.

Molte nazioni (Gran Bretagna, Germania, ecc.) suddividono il dominio nazionale ad alto livello in domini a secondo ed anche a terzo livello predeterminati. Altre nazioni non lo fanno (Italia).

Il concetto principale amministrativo del DNS e' la delega di responsabilita'. L'organizzazione principale del DNS, il Network Information Center (NIC) amministra solo la porzione dell'albero che copre i domini ad alto livello e delega la responsabilita' amministrativa dei sottodomini a singole organizzazioni nazionali.



Si definisce zona come la porzione dell'albero DNS sottoposta ad amministrazione singola. Molti domini a secondo livello sono separati in zone. Ogni zona ha una amministrazione che si cura dell'allocazione dei nomi ed indirizzi della zona e della manutenzione del database e configurazione dei server.

Un Domain Name Server ha autorita' su una o piu' zone, è necessario configurare per ogni zona un DNS primario ed alcuni (minimo due) secondari. Il primario ed i secondari devono essere indipendenti e ridondanti, in modo da offrire un servizio continuativo.

Il server primario contiene il database DNS vero e proprio in file di configurazione, i server secondari contattano il server primario a intervalli regolari (p.es. ogni 3 ore) ed eseguono un zone transfer, cioe' il download del database corrente.

Se un server che viene interpellato da un programma applicativo, non conosce l'informazione richiestagli, puo' agire in uno di due modi:

• prendersi carico della richiesta del programma applicativo e contattare a sua volta un altro server (ricerca ricorsiva)

• fornire al programma applicativo richiedente una lista di altri server da contattare direttamente (ricerca iterativa)

La maggior parte dei server compiono una ricerca ricorsiva. Esiste un certo numero di server DNS ad alto livello, mantenuti dal NIC, i 'root' server. Ogni server di zona deve possedere la lista completa dei root server e contattare loro per risolvere richieste non locali. I root server possiedono la lista completa di tutti i server di zona al primo livello, ciascuno dei quali possiede la lista dei server a secondo livello nel proprio sottoalbero, e cosi' via fino ad arrivare ad un server autorevole per la zona in cui si trova il nome di dominio richiesto. Quest'ultimo server fornisce il messaggio di responso con l'informazione desiderata direttamente al resover richiedente originale.

A partire da questo lucido illustreremo le caratteristiche di alcuni dei protocolli di livello applicativo dedicati all'invio e ricezione della posta elettronica. Il primo che andremo ad analizzare è il Simple Mail Transfer Protocol che ha come obiettivo quello di inviare messaggi di posta elettronica attraverso un protocollo di tipo testuale secondo i dettami introdotti per l'http.

Innanzitutto dire la RFC di riferimento e l'obiettivo che ci si pone con questo protocollo. Lo schema di base con cui avviene una comunicazione SMTP è la stessa che è stata vista per l'HHTP. Quando un client SMTP vuole trasmettere un messaggio, viene stabilita una comunicazione bidirezionale con un server SMTP. Il compito di un client SMTP è quello di trasferire un messaggio di posta ad uno o pià server SMTP o riportare il mancato trasferimento (fallimento della comunicazione).

Pertanto, appena si presenta al client un messaggio da inviare, si dovrà provvedere a trovare il dominio al quale inviare il messaggio. In molti casi, si tratta dell'indirizzo finale di posta elettronica, mentre in altri si tratta di un indirizzo intremedio, nel caso in cui si faccia utilizzo di altri protocolli come POP o IMAP (che vedremo in seguito). Una volta fatto ciò si passa alla copia del messaggio ad un server SMTP in grado di determinare la posizione del dominio (in accordo con quanto visto per il DNS). D'altro canto, il server può essere il destinatario finale del messaggio, oppure essere un punto intermedio oppure svolgere la funzione di gateway nel caso si necessiti di una traduzione di protocollo da SMTP ad un altro che svolga lo stesso compito. Per cui la comunicazione da instaurare per l'invio del messaggio può essere diretta, oppure passare attraverso degli hop intermedi.

Una volta stabilito il canale con la procedura di handshake completata il client inizia con l'invio della mail, considerando di specificare la sorgente e la destinazione della mail e la trasmissione del contenuto stesso del messaggio. Nel caso di invio a più destinatari, lo standard prevede di far seguire a tutte le copie la stessa strada.

A questo punto la comunicazione ha inizio con il meccanismo client/server con richiesta e risposta.

Senza scendere troppo nei dettagli riguardanti i comandi e le risposte in una comunicazione SMTP, contenuti nella RFC di riferimento sembra più opportuno analizzare una semplice comunicazione client (C:) Server (S:) ottenuta con il comando telnet www.example.com 25. (o con lo sniffing di rete)

Partiamo innanzitutto dalla formattazione: il protocollo SMTP è di tipo testuale, per cui occorrono dei tag per consentire di separare i vari messaggi. In particolare con il tag <CRLF> vengono terminati i comandi.

Passiamo ora all'analisi dei comandi della comunicazione:

Il primo che incontriamo è quello di Hello (HELO) che viene utilizzato per identificare il client da parte del server SMTP.

A questo punto, è possibile iniziare la transizione vera e propria della mail attraverso il comando MAIL FROM indicando, tra l'altro appunto chi invia la mail. D'altra parte con RCPT, ovviamente, si indica il destinatario della mail che può essere uno o più.

A questo punto avviene il trasferimento dei dati da parte del client utilizzando il comando DATA. A questo comando il server risponde con un messaggio dal codice 354, ovvero si indica che le successive linee non devono essere considerate come comandi, ma esclusivamente come dati veri e propri della mail da agggiungere in coda al buffer contenente la mail.

Infine la comunicazione viene interrotta attraverso il messaggio di QUIT al quale si risponde con un OK del server e la chiusura della comunicazione. Da sottolineare che tale comando può essere lanciato in qualsiasi momento e si pone come obiettivo quello di chiudere ognuna delle comunicazioni pendenti.

Infine trattiamo le risposte del server analizzando i codici di risposta che comunque risultano essere di semplice interpretazione. Ciascuna delle tre cifre che compongono i codici di risposta hanno un preciso significato: la prima indica se la risposta è stata positiva, negativa o incompleta. Per cui, anche il più stupido client SMTP può sapere quali azioni intraprendere dalla sola conoscenza della prima cifra. La risposta è positiva se la prima cifra è sia 1, 2, o 3. Nel caso della classe 2, l'azione è andata a buon fine e si procede alla successiva, mentre nel caso della classe 3 la risposta è positiva e si aspettano comandi da parte del client per continuare.

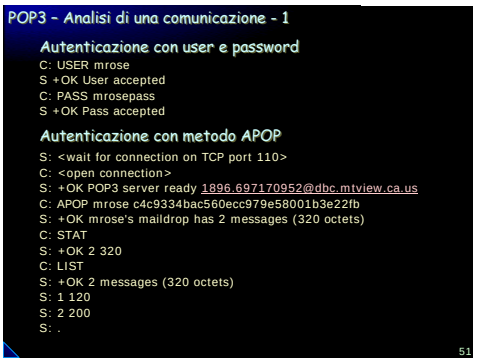
Per determinare, invece, il tipo di errore incontrato si deve analizzare la seconda cifra ed infine con la terza cifra si hanno le informazioni più precise possibili per la risposta.

Infine, le risposte 220 indicano il server pronto e 221 il bye e 250 ok.

Passiamo ora ad analizzare lo standard POP nella versione 3. Esso viene standardizzato nella rfc 1939 redatta nel 1996. Tale protocollo utilizza la porta 110 e si pone come obiettivo quello di far ricevere la posta a qualsiasi workstation, anche quello non in grado di mettere in piedi una comunicazione SMTP o che non può essere connessa (in alcuni casi la connessione POP cade dopo 10 min di inattività per evitare lunghi tempi di connessione). Per cui il protocollo POP3 ha come obiettivo quello di far accedere dinamicamente una macchina ad un server in modo utile. In altre parole con il protocollo POP3 si vuole consentire ad una stazione di prelevare la propria posta da un server che la trattiene per lui. In genere la posta viene scaricata e cancellata dal server. IMAP, protocollo che vedremo successivamente, rappresenta l'evoluzione di questo protocollo.

Per poter utilizzare questo protocollo si fa riferimento all'ascolto alla porta 110 e così come già visto per SMTP, si utilizza una codifica testuale, dove ciascun comando viene ad essere interrotto con il tag <CRLF>. Anche in questo caso si tratta di una architettura client-server, ovvero di fronte ad una richiesta da parte del client, si risponde on una risposta del server.

Inizialmente, il server avvia il servizio pop3 ascoltando la porta 110. Quando un client vuole prelevare la posta (lo si fa comunemente con outlook ed affini) viene stabilita una connessione con il server e quest'ultimo invia un greeting. A questo punto client e server si scambiano richieste/risposte fino alla fine della comunicazione. I comandi sono stringhe case sensitive, mentre le risposte sono codici numerici (come SMTP-HTTP ecc.)



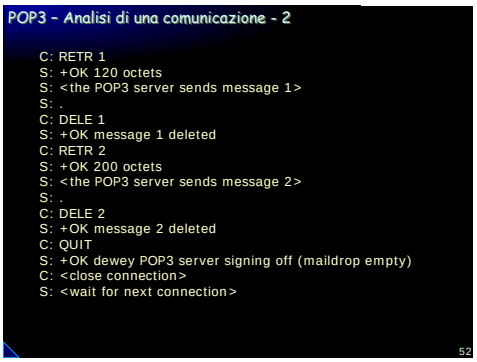
In questo lucido e nel prossimo analizzeremo un esempio di comunicazione utilizzando il protocollo POP3. IN questo caso verrà illustrata la comunicazione tra il server (S:) e client (C:) indicando le richieste e i corrispondenti messaggi di risposta.

Dapprima illustriamo il solo meccanismo di autenticazione che è proprio dei nostri sistemi di gestione della posta, ovvero quello con user name e password. Banalmente, il client invia al server il proprio user e la propria password e quest'ultimo accetterò o meno le informazioni fornite dal client prima di iniziare con la visualizzazione dei messaggi e lo scaricamento di questi ultimi. Il problema di questo metodo, così come per molti altri della stesso periodo vi era relativo all'autenticazione e sicurezza dei messaggi, in quanto nel caso sopra user e pwd viaggiano in chiaro (captabili da uno sniffer di rete) e pertanto si è introdotto per questo protocollo il metodo di autenticazione APOP.

Pertanto la comunicazione inizia con l'ascolto da parte del server nella porta 110 e quando viene aperta la connessione il server si dichiara pronto con il greeting server ready. A questo punto il client decide di prelevare la posta e sceglie come metodo, appunto, APOP. Con la stringa APOP deve essere specificato il nome della casella elettronica alla quale accedere e una stringa per l'autenticazione mediante funzioni di HASH (timestamp + parola chiave conosciuta da client/server. A questa si risponde solitamente con un messaggio di OK contenente lo stato della casella di posta.

Con il comando STAT, invece fornisce le informazioni sulla maildrop e con LIST si accede, invece alla scansione di tutti i messaggi e il server indica lo stato complessivo della casella e la lunghezza in byte di ciascun messaggio

C'è ben poco da dire per quanto riguarda la sintassi dei messaggi di risposta del server: esistono solo due tipi di risposta (testuale): +OK e –ERR.



A questo punto, senza scendere troppo nei dettagli e vista l'intuitività della comunicazione, il client con il comando RETR n (Retrieve n) chiede al server di scaricare l'n-esimo messaggio dal server e verificare che esso sia stato scaricato in maniera corretta. Similmente una volta fatto ciò è possibile scaricare il server POP con il comando DELE n (Delete n) che cancella l'n-esimo messaggio. Infine, la sessione si chiude con il messaggio di QUIT con conseguente scollegamente dell'utente e il server rimane in attesa per il prossimo collegamento.

Chiudiamo, infine, il discorso relativo ai protocollo di posta elettronica illustrando il protocollo IMAP nella sua versione 4 revision 1 del Marzo 2003. Si tratta di un protocollo client/server, che svolge le stesse funzioni di POP3. Ovvero fa in modo che un client connesso alla rete possa prelevare i propri messaggi di posta elettronica. In particolare, questo protocollo consente ad un client di accedere e manipolare i messaggi di posta elettronica direttamente su un server. Rispetto al protocollo POP3 i messaggi di posta non vengono scaricati dal server al pc, ma rimangono su di un server remoto. IMAP consente la manipolazione dei messaggi in un maniera del tutto simile a quella delle cartelle locali.

Senza scendere nei dettagli del protocollo (vedi RFC corrispondente) si passeranno a descrivere esclusivamente i vantaggi e gli svantaggi di tale protocollo in termini del POP3.

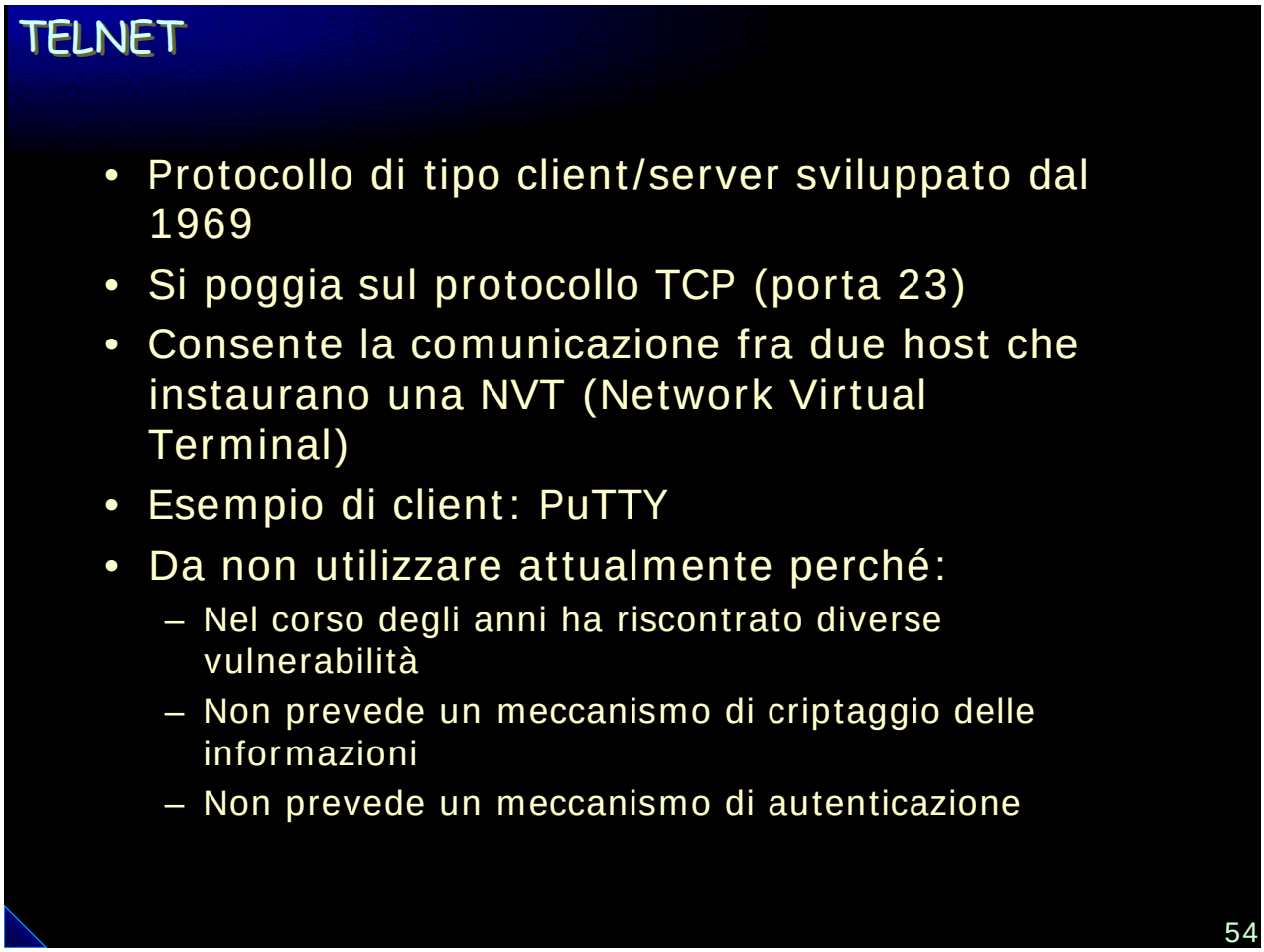
Uno dei primi vantaggi legati all'utilizzo del protocollo IMAP è relativo al fatto che si può provare a scaricare più velocemente la posta, nel senso che utilizzando il protocollo POP3 tutti i messaggi venivano scaricati e copiati sul computer locale, con il problema di lunghe connessioni nel caso di email molto lunghe, mentre nel caso di utilizzo del protocollo IMAP, allora la connessione può avere una durata molto più limitata.

Un altro vantaggio dell'utilizzo del protocollo IMAP risiede nella possibilità di consentire a più utenti di collegarsi contemporaneamente alla stessa casella elettronica, mentre nel POP3 era possibile una sola connessione per ciascun client. In tal caso, però, si deve poter tener traccia di tutte le modifiche effettuate ai messaggi per non avere versioni obsolete dei messaggi presenti nella casella di posta. Mediante appositi flag, inoltre, il server tiene traccia dello stato di ciascun messaggio. Per cui ciascun utente può controllare lo stato del proprio messaggio in ogni momento. I client IMAP possono inoltre creare, rinominare o cancellare le caselle di posta (solitamente presentate come cartelle) sul server e spostare messaggi tra le varie caselle di posta. Più caselle di posta forniscono l'accesso a cartelle pubbliche. Infine con IMAP viene fornito un meccanismo di ricerca direttamente sul server. In tal modo un client può impostare direttamente i criteri di ricerca e scaricare solo quei messaggi che a lui interessano senza dover per forza scaricare tutti i messaggi.

Per quanto riguarda gli svantaggi, il maggiore riguarda la complessità di implementazione e la difficoltà di gestire l'accesso multiplo di diversi utenti alla stessa casella di posta elettronica. Inoltre, la ricerca sul lato server può comportare un notevole dispiego di risorse in termini di capacità di elaborazione

Chiudiamo, infine, la panoramica relativa ai protocolli di posta elettronica, specificando che, nei nostri PC vengono utilizzati degli applicativi che gestiscono la posta. In commercio e liberi sono presenti diversi software, liberi e no. Tutti comunque utilizzano solitamente SMTP per inviare i messaggi di posta elettronica, mentre viene utilizzato il protocollo POP3 per la ricezione dei messaggi.

The Internet Message Access Protocol, Version 4rev1 (IMAP4rev1) allows a client to access and manipulate electronic mail messages on a server. IMAP4rev1 permits manipulation of mailboxes (remote message folders) in a way that is functionally equivalent to local folders. IMAP4rev1 also provides the capability for an offline client to resynchronize with the server.



Telnet è uno dei primi protocolli sviluppati dalla IETF (a partire dal 1969) con l'obiettivo di far comunicare fra loro due diversi punti presenti nella rete locale o nella rete Internet.

E' un protocollo di tipo client/server, nel quale un client effettua una richiesta che il server elabora e invia una risposta. La comunicazione che viene instaurata con questo protocollo è di tipo simmetrico in modo da consentire anche la comunicazione user-user e server-server (per eseguire processi cooperati) oltre alla tradizionale client-server

Al momento della connessione fra i due punti si stabilisce una NVT che sta per Network Virtual Terminal. Con questo termine si intende che per la durata della connessione è presente un terminale che consta di un meccanismo di scambio bidirezionale, ovvero vi è una tastiera e una stampante (monitor). Alla tastiera viene affidato il compito di inviare dati al server, mentre al monitor quello di rispondere, attraverso stampa, ai dati in ingresso. E' inoltre possibile coinvolgere, attraverso il comando di "echo", contemporaneamente sia tastiera che monitor.

In internet sono presenti diversi client e server telnet sia liberi che no e per qualsiasi sistema operativo. Nel lucido è stato menzionato Putty che costituisce il sistema di navigazione virtuale per Telnet, ma anche per altri protocolli come SSH che vedremo nel seguito.

Il protocollo in esame in questo lucido non è utilizzabile nei sistemi moderno in quanto presenta delle vulnerabilità nel meccanismo di comunicazione. Inoltre il suo uso in reti internet è inappropriato poiché non è dotato dei basilari meccanismi di sicurezza come il criptaggio e l'autenticazione. Questi problemi che non erano sorti nel corso del primo sviluppo del protocollo sono, invece, diventati basilari nel corso degli ultimi anni con lo sviluppo avuto dalle reti. Infatti nei primi periodi l'utilizzo di tale protocollo era limitato a reti locali protette dall'esterno. Mentre attualmente non avendo il criptaggio delle informazioni, con uno sniffer commerciale (come ad esempio wireshark) è possibile intercettare e interpretare il contenuto della comunicazione.

In quest'ultimo periodo questo protocollo sta per essere soppiantato da SSH (Secure SHell)

SSH – Secure SHell

- Con SSH vengono chiamate una serie di RFC che mettono in piedi una comunicazione sicura fra due punti connessi in remoto
- Attualmente è standardizzata la versione SSH-2
- Lista RFC di riferimento:
  - RFC 4250: The Secure Shell (SSH) Protocol Assigned Numbers
  - RFC 4251: The Secure Shell (SSH) Protocol Architecture
  - RFC 4252: The Secure Shell (SSH) Authentication Protocol
  - RFC 4253: The Secure Shell (SSH) Transport Layer Protocol
  - RFC 4254: The Secure Shell (SSH) Connection Protocol
  - RFC 4256: Generic Message Exchange Authentication for the Secure Shell Protocol (SSH)
  - RFC 4335: The Secure Shell (SSH) Session Channel Break Extension
  - RFC 4344: The Secure Shell (SSH) Transport Layer Encryption Modes
  - RFC 4345: Improved Arcfour Modes for the Secure Shell (SSH) Transport Layer Protocol
  - RFC 4419: Diffie-Hellman Group Exchange for the Secure Shell (SSH) Transport Layer Protocol
  - RFC 4716: The Secure Shell (SSH) Public Key File Format

55

Il protocollo in esame in questo lucido si pone gli stessi obiettivi del protocollo telnet visto in precedenza, ovvero quello di consentire la comunicazione in remoto fra due punti appartenenti alla rete, con l'aggiunta dei meccanismi di sicurezza, quali autenticazione e criptaggio delle informazioni attraverso un meccanismo di chiave pubblica e privata.

La versione attuale è datata Gennaio 2006 ed è definita SSH-2. Nel corso di questo lucido sono state elencate le varie RFC, che, collegate fra loro forniscono, forniscono la versione attuale della SSH.

Nel corso della lezione illustreremo esclusivamente la RFC 4251 che definisce l'architettura del protocollo specificando le necessità e le considerazioni da effettuare per realizzare una trasmissione sicura. Mentre per i dettagli su struttura dei pacchetti e dei messaggi si rimanda alle RFC di riferimento

(RFC 4250 utilizzata da IANA – ente che assegna i numeri, come indirizzi IP, per protocolli Internet – per dare i numeri)

La RFC 4251, introdotta nel precedente lucido, descrive l'architettura del protocollo, le notazioni e la terminologia utilizzata nei documenti SSH. Il protocollo SSH consta di tre parti principali:

•L'Authentication Protocol (RFC 4252)

•Il Transport Layer Protocol (RFC 4253)

•The Connecction Protocol (RFC 4254)

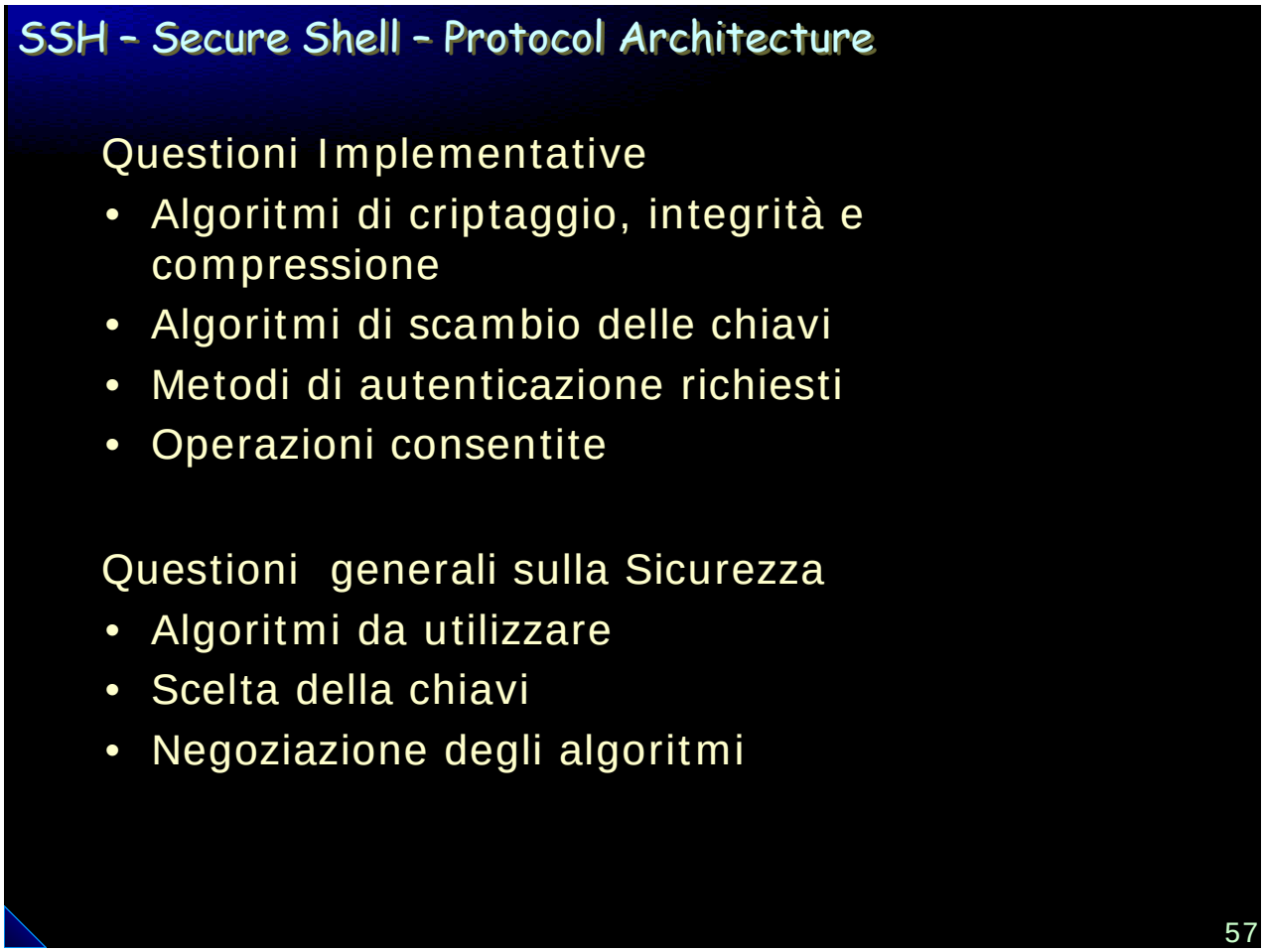
Per quanto riguarda la definizione del protocollo SSH la RFC riporta la seguente frase: SSH è un protocollo per il login da remoto sicuro e fornisce altri servizi di rete sicuri su una infrastruttura di rete non sicura. Inoltre esso consta principalmente di tre parti definite in altrettante RFC distinte

Il Transport Layer Protocol fornisce l'autenticazione al server, la confidenzialità (ovvero garantisce che le due parti in comunicazione siano quelle desiderate senza interferenza) e l'integrità (ovvero la non modifica del contenuto informativo)

Il protocollo di autenticazione dell'utente autentica l'utente dal lato client al server. Questo gira sul protocollo del livello di trasporto

Il protocollo di connessione specifica come multiplare più stream (canali) di dati sulla rete sicura. Specifica inoltre anche i canali per accedere con una shell interattiva, per il forwarding del proxy. Gira sopra il protocollo di autenticazione.

Il cliente invia una richiesta solo quando è stata stabilita una connessione sicura sullo strato di trasporto. Una seconda richiesta viene inviata dopo l'autenticazione d'utente. Ciò consente ai nuovi protocolli di essere definiti e di coesistere con i protocolli successivamente definiti. Il protocollo di connessione fornisce canali che possono essere usati per diversi scopi. Metodi standard sono forniti anche per impostare sessioni (shell) sicure e inoltrare le connessioni a porte TCP/IP arbitrarie (tunnelling).



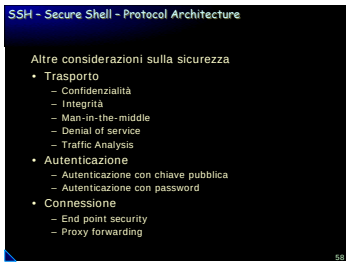
In questo lucido vengono illustrate alcune delle questioni implementative riferite al protocollo SSH. Innanzitutto si devono conoscere gli algoritmi utilizzati per il criptaggio, per garantire la integrità e quelli di compressione per ciascuna connessione. Inoltre si deve specificare quale sia quello preferenziale per ognuno di essi.

Inoltre si devono definire gli algoritimi di chiave Pubblica e di scambio delle chiavi utilizzate per l'autenticazione dell'host. L'esistenza di chiavi affidabili per differenti algoritimi a chiave pubblica influenzano questa scelta.

I metodi di autenticazione che defono essere richiesti per ciascun utente. Il server deve richiedere l'autenticazione multipla per alcuni o tutti gli utenti. Gli algoritmi dipendono dalla posizione da dove l'utente sta tentando l'accesso (ovvero non è possibile sempre collegarsi via SSH ovunque, ma solo da determinati indirizzi)

Infine le operazioni che l'utente è abilitato a svolgere utilizzando il protocollo di connessione. Alcune questione sono riferite alla sicurezza. Per esemio la politica di gestione non deve consentire al server di avviare sessioni o eseguire comandi sul client. Altre questioni tipo quali porte TCP/IP devono essere inoltrate e da chi sono chiaramente questioni locali. Molte di queste riguardano l'attraversamento o il superamento di firewall connesse con le questioni di gestione locali.

In aggiunta a quanto precedentemente esposto, si possono aggiungere delle questioni del tutto generali riferite alla sicurezza della comunicazione. Ovvero, la scelta degli algoritmi da utilizzare per criptaggio, integrità e scambio delle chiavi devono essere noti e stabiliti. Le chiavi da utilizzare devono garantire la sicurezza nei confronti del peggior attacco di criptanalisi. Ovvero, la dimensione delle chiavi da utilizzare deve essere tale da fare in modo che qualsiasi attacco di criptanalisi, volto alla scoperta delle chiavi, non debba andare a buon fine. Il metodo di criptanalisi più semplice è quello del brute force che prova tutte le possibili chiavi. Infine deve essere possibile la negoziazione degli algoritmi in modo da utilizzarne uno condiviso fra client e server ed inoltre si deve essere in grado di poter cambiare lo stesso algoritmo nel corso della sessione



In questo lucido verranno affrontate le principali questioni sulla sicurezza descritte nella RFC4251, la quale a sua volta rimanda alle altre RFC per la descrizione specifica dei protocolli che affrontano tali problematiche. Per i nostri scopi è sufficiente l'analisi dei problemi e come poterli risolvere, mentre è troppo dispersivo passare a raccontare i dettagli dei tre protocolli principalmente utilizzati per far funzionare SSH.

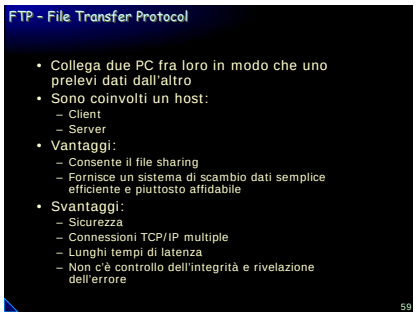
In riferimento ai problemi del livello di trasporto, in grado di mettere in piedi una rete sicura, la prima questione è quella relativa alla Confidenzialità e Integrità. Come già detto in precedenza si vuole che quello che viene trasmesso da un lato sia quello che viene effettivamente ricevuto dall'altro e che le parti in comunicazione siano quelle giuste senza intrusioni. Pertanto si tratta di scegliere un opportuno cifrario tra quelli messi a disposizione. Allo stato attuale, sono noti il 3DES, e soprattutto l'AES. Inoltre viene previsto anche il cifrario nullo. Tale cifrario che non effettua operazioni va utilizzato esclusivamente per le finalità di debug, in quanto la sua applicazione in ambienti reali non rispecchia i principi propri del SSH. Ovviamente un implementatore del sistema SSH deve utilizzare il miglior cifrario possibile tra quelli proposti in letteratura.

Anche per l'integrità è possibile disabilitare, per fini di debug, il protocollo che la garantisce.

Il problema del man-in-the-middle consiste nella presenza di un disturbatore tra i due estremi della comunicazione. Un punto di vulnerabilità di questo protocollo si riferisce al fatto che non sempre viene controllata la corrispondenza fra la chiave del server host ed il nome del server prima di instaurare il collegamento. In tal modo è semplice avere delle intrusioni stile man-in-the-middle. Pertanto i motivi di criticità riferiti all'associazione di chiave e nome dell'host prima dell'avvio della sessione rendono SSH non sicuro nei confronti di man-in-the-middle.

Sempre riferendosi a questo tipo di attacco sono possibili diversi scenari di attacco: l'attaccante posto fra client e server prima dell'inizio della sessione. L'obiettivo dell'attacco è quello di ottenere le chiavi per la sessione. Mentre nel caso in cui l'attaccante agisca dopo l'avvio della sessione, allora il suo compito sarà quello di ricevere e manipolare il flusso informativo.

A conclusione della parte relativa allo strato di trasporto trattiamo il problema dell'analisi del traffico. Infatti un costante monitoraggio del traffico può fornire all'attaccante alcune informazioni relative alla sessione, l'utente o le informazioni specifiche del protocollo che altrimenti non avrebbero potuto ottenere. Per esempio, si è visto sperimentalmente che l'analisi di una sessione SSH può far ricavare informazioni relative alla lunghezza della password. Inoltre, riferendosi ancora all'analisi del traffico, un problema noto è quello del non invio di dati. Infatti, qualora si instauri una sessione di comunicazione SSH senza trasmettere nulla, quello che si ottiene è di cifrare esclusivamente il contesto della comunicazione, ovvero la chiave e di conseguenza, grazie all'utilizzo di algoritimi pubblici è possibile ricavare la chiave. Pertanto gli implementatori di un sistema SSH devono prevedere dei meccanismi di sicurezza all'analisi del traffico, il più semplice dei quali è relativo all'inserimento di bit di padding che si vadano a sostituire ai momenti di non comunicazione.



In sostanza il protocollo FTP si occupa di far comunicare fra loro due PC, in modo tale che uno prelevi dei dati dall'altro.

Si comprende così, che anche per questo protocollo si farà riferimento ad un modello client-server. Ovvero una postazione potrà accedere ai contenuti di un'altra postazione remota (server) dalla quale potrà prelevare i file da esso condivisi. Così come per il protocollo HTTP anche i server FTP provvedono a rispondere ai client attraverso dei messaggi che indicano lo stato del trasferimento dei dati, messaggi che non vedremo.

Discuteremo invece dei vantaggi e svantaggi che comporta l'utilizzo di tale protocollo. Dall'analisi di questi aspetti vedremo alcune peculiarità del protocollo.

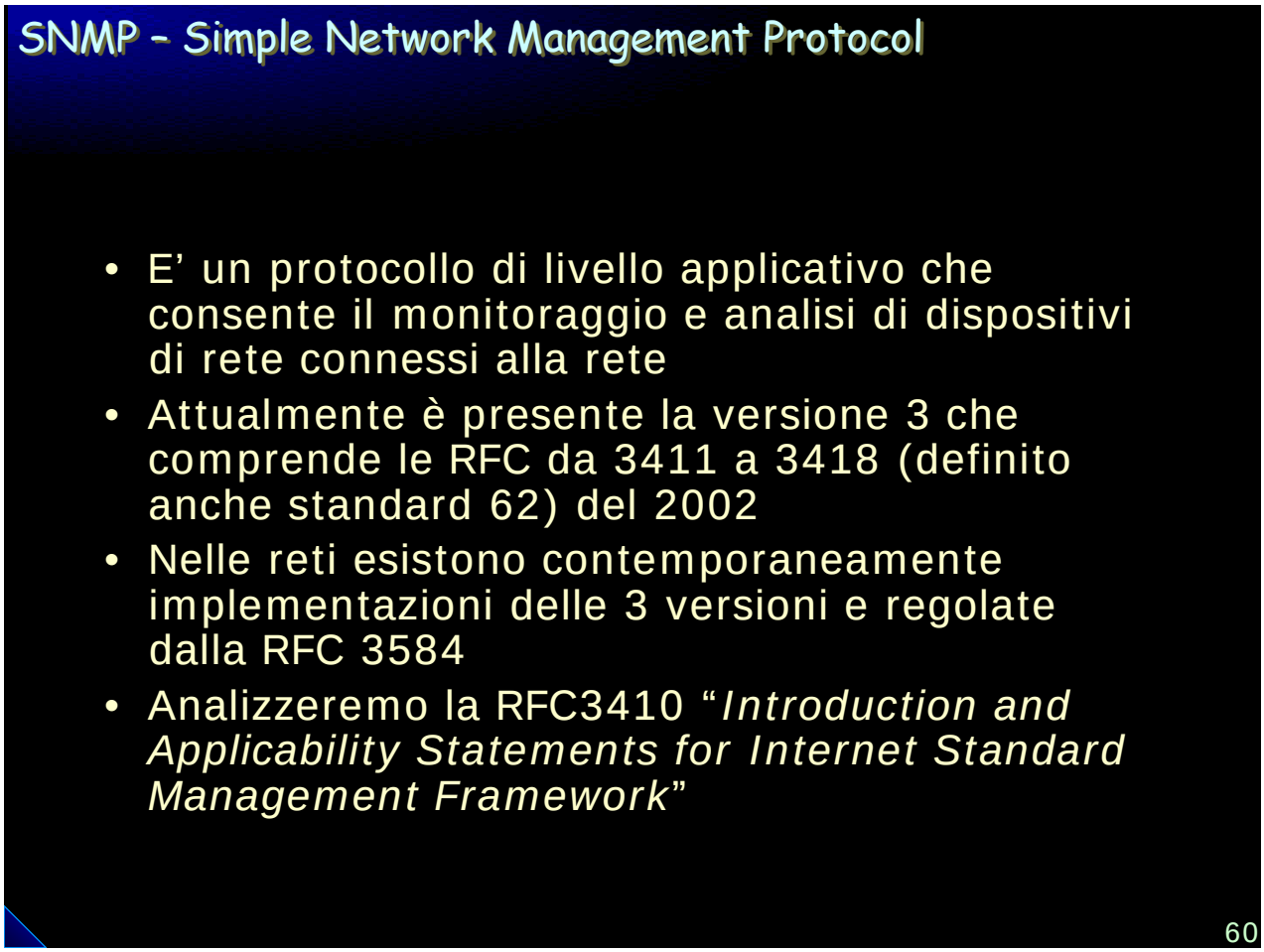
Innanzitutto il grande vantaggio e obiettivo di questo protocollo è quello di promuovere un uso massiccio del file sharing e dei computer remoti. In aggiunta fornisce un sistema di scambio dei dati molto semplice per un utente non esperto ed anche efficiente e piuttosto affidabile.

A questi vantaggi si contrappongono però diversi svantaggi molto importanti. In primo luogo le questioni relative alla sicurezza della comunicazione fra i due host di rete. Innanzitutto sia il contenuto dei file che le password viaggiano nella rete in chiaro e in tal modo soggetti a intercettazioni (Eavesdropping). Questo problema è molto importante, in quanto non viene previsto nessun metodo per criptare i dati da trasferire e pertanto con l'utilizzo di un solo sniffer (ad esempio ethereal-wireshark come vedremo) è possibile vedere e intercettare i dati trasmessi. Tale problema è esistente con l'utilizzo di protocolli come HTTP, SMTP e Telnet. Un miglioramento nella sicurezza si ha con il SFTP (SSH FTP) che si basa sull'utilizzo del protocollo SSH che vedremo in seguito.

Un'altra questione riguarda la presenza di più connessioni TCP/IP contemporanee. Vengono infatti stabilite connessioni per il controllo della connessione, per l'upload, download e aggiornamento della directory list. Per poter gestire al meglio tali connessioni si necessiterebbe di un meccanismo di firewalling. In realtà risulta molto difficile filtrare il traffico FTP attivo dal lato client fintanto che i client possano utilizzare qualsiasi porta purchè venga messa in piedi la comunicazione.

In aggiunta vi sono tempi di latenza molto lunghi dovuti essenzialmente alla mole di informazioni che devono essere trasferite prima di avviare l'effettivo scambio di file.

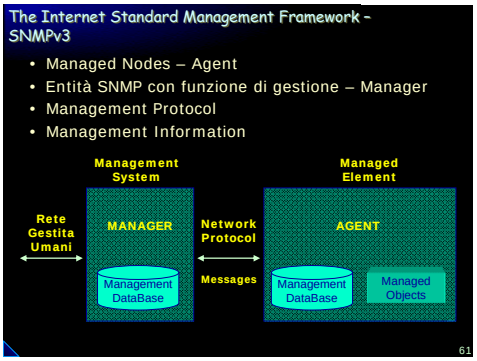
Infine una ultima criticità di questo protocollo riguarda il controllo sulle informazioni scambiate. Non vi sono infatti meccanismi di controllo delle informazioni e verificare se il flusso di dati sia stato o meno interrotto. Infine non c'è una rivelazione dell'errore in quanto l'unico meccanismo a tal proposito è il CRC fornito con il sottostante protocollo FTP.



Chiudiamo la rassegna di tutti i protocolli di livello provando a descrivere il protocollo applicativo definito SNMP (Simple Network Management Protocol). Con questo protocollo si vuole facilitare lo scambio di informazioni tra dispositivi di rete, inoltre consente agli amministratori di rete di gestire le prestazioni della rete, trovare e risolvere problemi e pianificare la crescita delle reti.

Attualmente, la versione ufficiale è la 3, definita nella serie delle RFC a partire dalla 3411 fino alla 3418 finite di standardizzare nel 2002. In realtà, però esistono dispositivi che utilizza i vecchi protocolli con versioni 1 e 2. Pertanto, si è scritta una RFC apposita (la RFC 3548 dal titolo "Coexistence between Version 1, Version 2, and Version 3 of the Internet-standard Network Management Framework") che consente la coesistenza di tutte le versioni SNMP nelle reti.

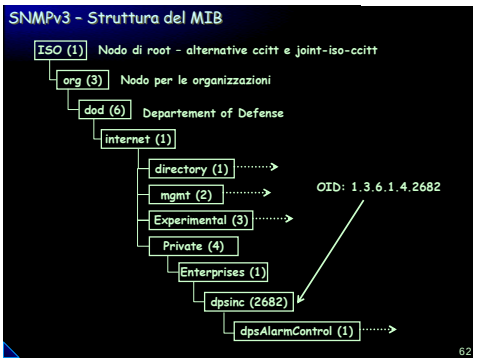
Nel corso dei prossimi lucidi ci riferiremo alla RFC3410 che è uno standard di tipo informativo, pertanto da utilizzare solo come tutorial e non contenente specifiche di applicazioni e progetti. In questa RFC viene descritta la struttura di SNMPv3 (definita Internet Standard Management Framework) e il rapporto di questa con le precedenti versioni.



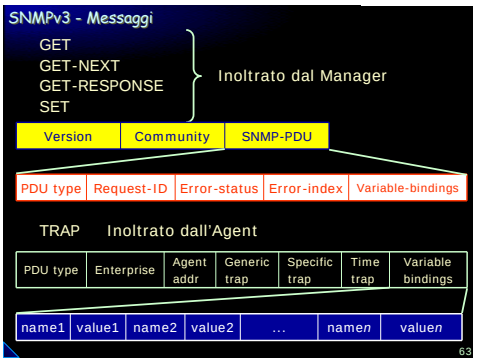
La versione 3 di SNMP condivide la stessa struttura di base delle precedenti versioni. In particolare sono presenti quattro componenti di base che agiscono secondo lo schema sottostante.

La struttura è di tipo manager e agent, Il manager fornisce un interfaccia fra una rete gestita dall'amministratore di rete in carne ed ossa e il sistema di gestione. L'agent invece fornisce l'interfaccia fra il manager e i dispositivi fisici da gestire.

Il manager e l'agent utilizzano il Management Information Base (MIB) dove vengono definite le informazioni di gestione. SNMP utilizza inoltre un insieme di comandi per scambiare le informazioni relativamente piccolo. Inoltre il MIB è organizzato con una struttura ad albero con variabili individuali rappresentati come i rami. Infine un lungo tag numerico (OID – Object Identifier) è usato per identificare univocamente ogni variabile all'interno del MIB e dei messaggi SNMP.



Nel lucido è possibile analizzare come vengono gestiti gli oggetti gestiti con il protocollo SNMP. Tali informazioni sono organizzate con una struttura gerarchica ad albero. Ciascun oggetto (caratteristica) da monitorare ha un unico identificativo (OID=Object ID) che consiste in una serie di numeri separati da un punto a cui viene associata una etichetta leggibile (ad es. 1.3.6.1=Internet). Pertanto lo scopo del MIB è quello di fornire un dizionario utilizzato per assemblare e interpretare i messaggi SNMP.



SNMP utilizza principalmente cinque differenti messaggi per comunicare tra il manager e l'agent. I messaggi di GET e GET-NEXT consentono al manager di richiedere informazioni all'agent in merito ad una specifica variabile (e alla seguente). L'agente, una volta ricevuto uno dei due messaggi precedentemente descritti invia un messaggio di GET-RESPONSE al manager con le informazioni richieste oppure un indicazione di un errore indicante il perché la richiesta non è stata processata. Un messaggio di SET consente al manager di richiedere un cambio di uno specifico valore in caso di allarme remoto. Anche in questo aso l'agente risponde con un messaggio di GET-RESPONSE che indica se il cambiamento è stato fatto o meno e riportando il codice di errore. Infine il messaggio di TRAP consente all'agente di inviare spontaneamente delle informazioni al manager. Tra i cinque tipi di messaggi l'unico che viene inviato direttamente dall'agent è quello di TRAP. Esso viene utilizzato essenzialmente per indicare le condizioni di allarme di un dispositivo. In tal modo si invia subito l'allarme senza dover aspettare che il manager SNMP lo richieda.

Descriviamo, ora la struttura dei messaggi. I primi 4 rispondono ad una determinata struttura, mentre il TRAP ha una propria struttura, diversa dagli altri quattro messaggi.

In linea del tutto generale un pacchetto di monitoraggio SNMP contiene un campo di versione e uno di community che contiene una sorta di password per il controllo degli accessi del client al server. Infine vi è il pacchetto SNMP vero e proprio che a sua volta contiene altre informazioni.

Come è possibile vedere, vi due possibili tipi di PDU per i messaggi SNMP. Una contiene come PDU type uno tra i messaggi di GET, GET-NEXT GET-RESPONSE e SET, mentre l'ultimo è proprio del messaggio di TRAP.

In entrambi i pacchetti è inoltre presente il campo Variable binding che contengono come identificatore un tipo e un valore (nel caso di SET o di RESPONSE). Ogni Agent controlla ogni identificatore del proprio MIB per indicare il nome della variabile ed interpretarne il valore.

Riferendosi ai messaggi inoltrati dal Manager, nel pacchetto viene inserito l'identificativo della richiesta, se si verifica un errore e infine il campo Variable-Bindings come già illustrato in precedenza.

Il messaggio di TRAP indica un campo definito Enterprise dove viene indicato quale tipo di hardware abbia generato il TRAP, inserendone l'indirizzo IP nel campo Agent Addr. Il campo Generic Trap descrive in maniera del tutto generale il problema, mentre il campo specific contiene ulteriori informazioni sul Trap dipendente dal campo Enterprise. Viene inoltre attaccato anche un timestamp indicante l'istante della trap. Per Variable Binding vale quanto già detto in precedenza.

Riassumendo la procedura di comunicazione e monitoraggio: Quando un manager SNMP vuole conoscere il valore di una caratteristica, crea un pacchetto GET che comprende l'OID di ogni caratteristica di interesse. L'elemento riceve la richiesta e controllo l'OID nel proprio MIB. Se viene trovato l'OID, allora viene assemblato un pacchetto di risposta con il valore corrente della caratteristica ricercata, o se non trovata una risposta di errore che specifica il perché non è stato possibile reperire l'informazione. Quando un agent, invece, invia un pacchetto di TRAP, viene incluso il valore della OID e un valore per dettagliare l'evento. I manager più evoluto possono utilizzare i binding per correlare gli eventi accaduti con la gestione, inoltre essi generano delle etichette leggibili per semplificare il lavoro di decisione del gestore del sistema.