SIA · SISTEMI INFORMATIVI AZIENDALI · DISPENSA DI STUDIO

Il viaggio del pacchetto

La pila ISO/OSI, l'incapsulamento, la connessione TCP, il protocollo HTTP e la sicurezza con HTTPS contro l'attacco man-in-the-middle, spiegati con un unico esempio: un browser che chiede una pagina a un server web.



Prof. Enrico Maria Caruso
Informatica · classi quarte e quinte

Lezione interattiva collegata:
emcdigitalsolutions.it/didattica/viaggio-del-pacchetto

Indice

1 Reti e architettura client-server

2 Il modello ISO/OSI

3 Incapsulamento e decapsulamento

4 Il viaggio attraverso il router

5 TCP: la connessione affidabile

6 HTTP: richiesta e risposta

7 Sicurezza e attacco man-in-the-middle

8 HTTPS e TLS 1.3

9 Glossario delle sigle

10 Domande di ripasso

11 Mappa concettuale

Come usare questa dispensa

Ogni capitolo segue lo stesso esempio della lezione interattiva: lo studente apre il browser e scrive `www.example.com/index.html`. Le sigle sono spiegate la prima volta che compaiono e raccolte nel glossario (capitolo 9). Alla fine trovi le domande di ripasso con le soluzioni e una mappa concettuale da usare per ripetere.

L'esempio che useremo

	HOST A · CLIENT	HOST B · SERVER
Ruolo	Browser dello studente	Server web www.example.com
Indirizzo IP	`198.51.100.23`	`203.0.113.10`
Indirizzo MAC	`3c:22:fb:1a:07:9e`	`52:54:00:6f:a3:21`
Porta	`52344 (temporanea)`	`80 (HTTP) · 443 (HTTPS)`

Gli indirizzi sono riservati alla documentazione dallo standard RFC 5737 (RFC, Request for Comments: i documenti ufficiali che descrivono i protocolli di Internet): non appartengono a nessun computer reale.

1 Reti e architettura client-server

Una rete è un insieme di dispositivi collegati che si scambiano dati seguendo regole comuni, chiamate **protocolli**.

Ogni dispositivo collegato a una rete si chiama **host**. Su Internet la maggior parte dei servizi funziona con l'**architettura client-server**:

- il **client** è il programma che **chiede** un servizio (il browser, l'app di posta, un videogioco online);
- il **server** è il programma che **resta in ascolto** e **risponde** (il server web, il server di posta).

Il client prende sempre l'iniziativa; il server è sempre acceso e aspetta richieste su una **porta** conosciuta (80 per HTTP, 443 per HTTPS). Lo stesso server può servire migliaia di client contemporaneamente.

> **Protocollo**
>
> Un protocollo è un insieme di regole che stabilisce il formato dei messaggi, il loro ordine e cosa fare quando arrivano. Esempio di vita quotidiana: una telefonata. Chi risponde dice «Pronto», chi chiama si presenta, alla fine ci si saluta. Se uno dei due non rispetta le regole, la comunicazione non funziona.

Gli indirizzi

Per comunicare servono tre tipi di indirizzo, che lavorano a livelli diversi:

INDIRIZZO	COSA IDENTIFICA	ESEMPIO	LIVELLO
MAC (Media Access Control)	La scheda di rete, dentro la rete locale	`3c:22:fb:1a:07:9e`	2
IP (Internet Protocol)	L'host, in tutta Internet	`198.51.100.23`	3
Porta	Il programma (processo) dentro l'host	`80, 443, 52344`	4

Un'analogia: l'IP è l'indirizzo del condominio, la porta è il numero dell'appartamento, il MAC è il nome sul campanello della singola stanza nella rete di casa.

Dal nome all'indirizzo: il DNS

Le persone ricordano nomi come `www.example.com`, i computer usano indirizzi IP. Il **DNS** (Domain Name System, sistema dei nomi di dominio) traduce il nome nell'indirizzo `203.0.113.10` prima che il browser possa connettersi. Il DNS è anche lui un protocollo applicativo e di solito usa UDP sulla porta 53.

---

2 Il modello ISO/OSI

Il modello **OSI** (Open Systems Interconnection, interconnessione di sistemi aperti) è stato definito dall'**ISO** (International Organization for Standardization, organizzazione internazionale per la normazione). Divide la comunicazione in sette livelli.

Dividere un problema complesso in livelli ha tre vantaggi: ogni livello ha un compito solo, si può cambiare un livello senza toccare gli altri (per esempio passare dal cavo al Wi-Fi senza cambiare il browser) e prodotti di aziende diverse possono funzionare insieme.

Due regole fondamentali

- Ogni livello usa i **servizi del livello sotto** e offre servizi al livello sopra.
- Ogni livello «parla» con lo **stesso livello dell'altro host** (comunicazione tra pari), usando un suo protocollo.

N°	LIVELLO	COMPITO	PDU	ESEMPI	DISPOSITIVI
7	**Applicazione**	Offre i servizi di rete ai programmi	Dati	HTTP, HTTPS, DNS, SMTP, FTP	Host
6	**Presentazione**	Rappresentazione dei dati: codifica, compressione, cifratura	Dati	UTF-8, JPEG, gzip, TLS	Host
5	**Sessione**	Apre, mantiene e chiude il dialogo	Dati	Sessione TLS, RPC	Host
4	**Trasporto**	Collega i processi con le porte; affidabilità	Segmento	TCP, UDP	Host, firewall
3	**Rete**	Indirizzi IP e instradamento fra reti	Pacchetto	IPv4, IPv6, ICMP	Router
2	**Collegamento dati**	Consegna nella rete locale con il MAC; scopre errori	Frame	Ethernet, Wi-Fi	Switch, access point
1	**Fisico**	Trasmette i bit come segnali	Bit	Cavo UTP, fibra, radio	Hub, ripetitore

La **PDU** (Protocol Data Unit, unità di dati del protocollo) è il «pacchetto» visto da ciascun livello: il suo nome cambia a seconda del livello.

I livelli uno per uno

7 · Applicazione. È il livello dei programmi che usiamo: il browser parla HTTP (HyperText Transfer Protocol), il client di posta parla SMTP (Simple Mail Transfer Protocol), il DNS traduce i nomi.

6 · Presentazione. Decide come sono scritti i dati, perché i due host li interpretino nello stesso modo: codifica dei caratteri (UTF-8, Unicode Transformation Format a 8 bit), compressione (gzip), formati delle immagini (JPEG) e cifratura.

5 · Sessione. Organizza il dialogo: chi parla, quando, come riprendere una conversazione interrotta.

4 · Trasporto. Porta i dati dal processo del client al processo del server. **TCP** (Transmission Control Protocol) garantisce che arrivino tutti e in ordine; **UDP** (User Datagram Protocol) è più veloce ma non garantisce nulla.

3 · Rete. Usa gli indirizzi **IP** e sceglie la strada attraverso reti diverse (instradamento o routing). È il livello dei router.

2 · Collegamento dati. Consegna i dati al dispositivo successivo nella stessa rete locale (LAN, Local Area Network) usando gli indirizzi MAC e controlla gli errori di trasmissione.

1 · Fisico. Trasforma i bit in segnali: impulsi elettrici nel cavo, luce nella fibra ottica, onde radio nel Wi-Fi.

OSI e TCP/IP

Internet non usa esattamente il modello OSI ma il modello **TCP/IP**, a quattro livelli. OSI resta il riferimento per studiare e per descrivere i problemi di rete.

MODELLO OSI	MODELLO TCP/IP	PROTOCOLLI TIPICI
7 Applicazione · 6 Presentazione · 5 Sessione	**Applicazione**	HTTP, HTTPS, DNS, TLS
4 Trasporto	**Trasporto**	TCP, UDP
3 Rete	**Internet**	IP, ICMP
2 Collegamento dati · 1 Fisico	**Accesso alla rete**	Ethernet, Wi-Fi

Da ricordare

In TCP/IP i livelli 5, 6 e 7 li svolge l'applicazione stessa: per questo, nel nostro esempio, non compare un'intestazione separata per Sessione e Presentazione.

---

3 Incapsulamento e decapsulamento

Quando il browser invia la richiesta, i dati scendono la pila del client. Ogni livello aggiunge davanti la sua **intestazione** (header): è l'**incapsulamento**.

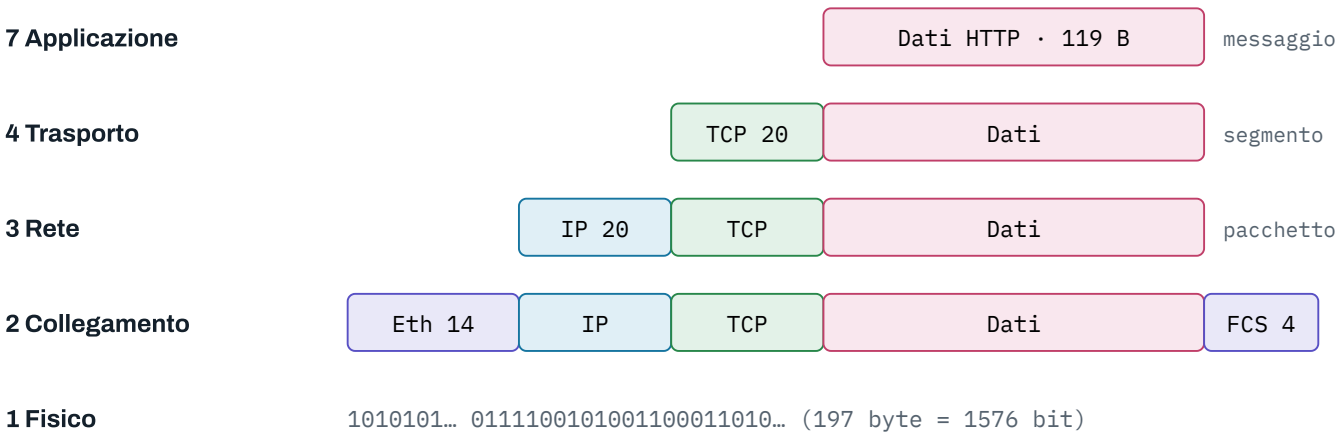


Figura 1 · Incapsulamento della richiesta GET. I numeri sono le dimensioni in byte delle intestazioni. Il frame Ethernet ha anche una coda, l'FCS.

Che cosa aggiunge ogni livello

- **Applicazione:** il messaggio HTTP, cioè il testo della richiesta (119 byte nel nostro esempio).

- **Trasporto:** l'intestazione **TCP** con porta sorgente (52344), porta di destinazione (80), numero di sequenza e numero di conferma. La PDU diventa un **segmento**.
- **Rete:** l'intestazione **IP** con indirizzo del mittente e del **destinatario finale** e il **TTL** (Time To Live, tempo di vita: parte da 64 e cala di 1 a ogni router; a 0 il pacchetto viene scartato). La PDU diventa un **pacchetto**.
- **Collegamento dati:** l'intestazione **Ethernet** con MAC di destinazione e di sorgente e, in fondo, la coda **FCS** (Frame Check Sequence, sequenza di controllo calcolata con il CRC, Cyclic Redundancy Check) per scoprire errori. La PDU diventa un **frame**.
- **Fisico:** trasforma il frame in **bit**, preceduti da un preambolo che serve a sincronizzare il ricevitore.

> **Decapsulamento**
>
> Sul server il percorso è inverso: ogni livello legge la sua intestazione, controlla che sia per lui, la toglie e passa il resto al livello sopra. Alla fine il server web riceve esattamente il testo che il browser aveva scritto.

Le dimensioni del nostro esempio

PARTE	BYTE	TOTALE PROGRESSIVO
Dati HTTP (richiesta GET)	119	119
+ intestazione TCP	20	139 (segmento)
+ intestazione IP	20	159 (pacchetto)
+ intestazione Ethernet + FCS	14 + 4	197 (frame)

Le intestazioni occupano 78 byte su 197: circa il 40%. Per messaggi piccoli il «costo» delle intestazioni è alto; per file grandi diventa trascurabile.

---

4 Il viaggio attraverso il router

Il server non sta nella stessa rete locale del client. Il frame non può quindi andare direttamente al server: va consegnato al **router**, il gateway predefinito della rete.



Figura 2 · Cosa cambia e cosa resta uguale quando il pacchetto attraversa un router.

Che cosa fa il router, passo per passo

1. Riceve i bit e ricompone il frame; controlla l'FCS e verifica che il MAC di destinazione sia il suo.
2. Toglie l'intestazione Ethernet e legge l'**IP di destinazione**.
3. Consulta la **tabella di instradamento** per scegliere l'interfaccia di uscita (qui: la rete `203.0.113.0/24` si raggiunge dall'interfaccia verso Internet).
4. Abbassa il TTL di 1 (da 64 a 63).
5. Costruisce un **frame nuovo** con il suo MAC come sorgente e il MAC del dispositivo successivo come destinazione, e lo trasmette.

Il router lavora fino al livello 3: non guarda le porte e non legge il contenuto HTTP. Su Internet un pacchetto attraversa spesso da 10 a 20 router, e ognuno ripete lo stesso lavoro.

> **Come fa il client a conoscere il MAC del router?**
>
> Con il protocollo **ARP** (Address Resolution Protocol, protocollo di risoluzione degli indirizzi). Il client chiede a tutta la rete locale: «chi ha l'IP del gateway?». Il router risponde con il suo MAC, che resta memorizzato per un po' nella cache ARP.

> **Nota sul NAT**
>
> Nelle reti di casa e di scuola il router fa anche il **NAT** (Network Address Translation, traduzione degli indirizzi di rete): sostituisce l'IP privato del client con il proprio IP pubblico. Nel nostro esempio lo abbiamo tralasciato per concentrarci sull'instradamento.

---

5 TCP: la connessione affidabile

HTTP si appoggia su TCP. Prima di inviare la richiesta, client e server devono aprire una **connessione**.

Che cosa garantisce TCP

- **Consegna affidabile:** ogni byte ricevuto viene confermato; quelli persi vengono ritrasmessi.
- **Ordine:** i numeri di sequenza permettono di rimettere i byte nell'ordine giusto.
- **Multiplazione con le porte:** più programmi sullo stesso host possono comunicare insieme.
- **Controllo di flusso:** il campo «finestra» evita di inviare più dati di quanti il ricevitore riesca a gestire.

Il three-way handshake (stretta di mano in tre passi)

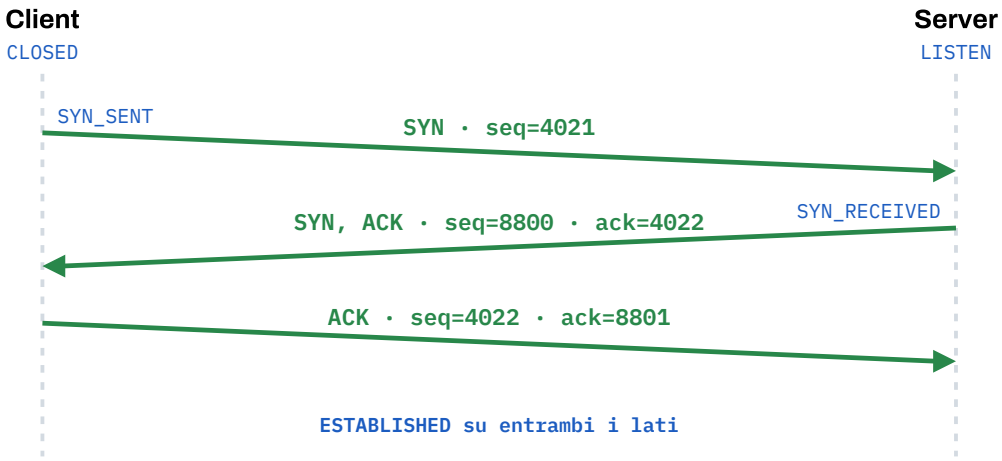


Figura 3 · Apertura della connessione TCP. Sotto i nomi degli host, il loro stato.

1. **SYN** (synchronize): il client sceglie un numero di sequenza iniziale casuale (4021) e chiede di connettersi.
2. **SYN-ACK**: il server conferma con ACK (acknowledgement, conferma) = 4022 e propone il proprio numero iniziale (8800).
3. **ACK**: il client conferma con ack = 8801. La connessione è aperta (`ESTABLISHED`).

> **La regola dell'ACK**
>
> Il numero di conferma indica sempre il **prossimo byte che ci si aspetta** di ricevere. Il flag SYN, anche senza dati, «consuma» un numero: per questo al SYN con seq = 4021 si risponde con ack = 4022. Dopo la richiesta HTTP di 119 byte, il server confermerà con ack = 4022 + 119 = 4141.

La chiusura

Quando il dialogo è finito, una delle due parti invia **FIN** (finish). Ogni direzione si chiude separatamente: FIN, ACK in un verso e FIN, ACK nell'altro (spesso il server unisce il suo ACK e il suo FIN in un solo segmento, quindi bastano tre segmenti). Chi chiude per primo resta per qualche istante in stato `TIME_WAIT`, nel caso l'ultimo ACK andasse perso.

	TCP	UDP
Connessione	Sì, con handshake	No
Affidabilità e ordine	Garantiti	Non garantiti
Velocità	Più lento	Più veloce
Usi tipici	Web, posta, trasferimento file	DNS, videochiamate, giochi online

6 HTTP: richiesta e risposta

HTTP (HyperText Transfer Protocol, protocollo di trasferimento di ipertesti) è il protocollo con cui browser e server web si scambiano pagine. È un protocollo a **richiesta e risposta** e i suoi messaggi sono testo leggibile.

La richiesta

```
GET /index.html HTTP/1.1          ← riga di richiesta: metodo, percorso, versione
Host: www.example.com             ← intestazioni (nome: valore)
User-Agent: Mozilla/5.0
Accept: text/html
Connection: keep-alive
                                  ← riga vuota: fine delle intestazioni
```

I **metodi** più usati sono `GET` (chiedere una risorsa), `POST` (inviare dati, per esempio un modulo di login), `PUT` e `DELETE`. La **URL** (Uniform Resource Locator, localizzatore uniforme di risorse) `http://www.example.com/index.html` indica protocollo, nome del server e percorso.

La risposta

```
HTTP/1.1 200 OK                   ← riga di stato: versione, codice, descrizione
Content-Type: text/html; charset=utf-8
Content-Length: 175
Connection: keep-alive

<!doctype html>                   ← corpo: la pagina HTML
<html lang="it"> …
```

CODICE	CLASSE	ESEMPIO
1xx	Informazione	100 Continue
2xx	Successo	200 OK: la risorsa è stata trovata e inviata
3xx	Reindirizzamento	301 Moved Permanently: la risorsa si è spostata
4xx	Errore del client	404 Not Found: la risorsa non esiste
5xx	Errore del server	500 Internal Server Error

Pagina statica e pagina dinamica

Una pagina **statica** è un file già pronto sul disco del server, inviato così com'è e uguale per tutti (come la nostra `index.html`). Una pagina **dinamica** viene generata da un programma (PHP, Node.js, Python) a ogni richiesta, per esempio leggendo un database.

Connessione persistente (keep-alive)

Una pagina web contiene anche immagini, fogli di stile e script: ognuno è una richiesta HTTP. Con `Connection: keep-alive` tutte le richieste usano la stessa connessione TCP, senza rifare ogni volta l'handshake. È il compito che il modello OSI assegna al livello Sessione.

Il problema di HTTP

HTTP viaggia **in chiaro**: chiunque riesca a intercettare il traffico può leggere ogni parola, password comprese, e persino modificare la risposta. Il capitolo successivo mostra come.

---

7 Sicurezza e attacco man-in-the-middle

Una comunicazione sicura deve garantire tre proprietà.

PROPRIETÀ	SIGNIFICATO	COME SI OTTIENE
Riservatezza	Solo mittente e destinatario possono leggere i dati	Cifratura
Integrità	I dati non possono essere modificati senza che ce ne si accorga	Codici di autenticazione (tag)
Autenticità	Si è sicuri dell'identità dell'altra parte	Certificati digitali

L'attacco man-in-the-middle

Nell'attacco **MITM** (Man In The Middle, uomo nel mezzo) l'attaccante si mette fra client e server, che credono di parlarsi direttamente. Tutti i messaggi passano da lui. Due modi tipici:

- **Wi-Fi finto** (evil twin): l'attaccante crea una rete aperta dal nome invitante, per esempio «WiFi_Gratis», e chi si collega gli consegna tutto il traffico.
- **ARP spoofing**: nella rete locale l'attaccante invia risposte ARP false e fa credere al client che il MAC del gateway sia il suo.

Che cosa succede con HTTP

Nella lezione l'attaccante si chiama Mallory, il nome che i testi di crittografia danno a chi attacca attivamente.

1. L'handshake TCP passa da Mallory, che inoltra tutto: nessuno se ne accorge.
2. Lo studente fa il login. La richiesta è testo leggibile e Mallory copia la password:

```
POST /login HTTP/1.1
Host: www.example.com
Content-Type: application/x-www-form-urlencoded

utente=mario.rossi&password=Gatto2026!
```

3. Mallory modifica la risposta del server e aggiunge un link a un finto «aggiornamento obbligatorio» che in realtà è un programma dannoso.

> **Esito con HTTP**
>
> Riservatezza ✗ (password letta) · Integrità ✗ (pagina modificata) · Autenticità ✗ (il browser non può sapere con chi sta parlando).

---

8 HTTPS e TLS 1.3

HTTPS (HTTP Secure) è HTTP trasportato dentro un canale cifrato **TLS** (Transport Layer Security, sicurezza del livello di trasporto), sulla porta 443. TLS ha sostituito il vecchio SSL (Secure Sockets Layer).

Nel modello OSI TLS si colloca fra il livello Sessione (apre e gestisce la sessione sicura) e il livello Presentazione (cifra i dati). L'ordine reale è: **prima** l'handshake TCP, **poi** l'handshake TLS, **infine** le richieste HTTP, già cifrate.

L'handshake TLS 1.3

1. **ClientHello:** il browser propone la versione (TLS 1.3), gli algoritmi di cifratura che conosce e la sua **chiave pubblica** per lo scambio di chiavi. Indica anche il nome del sito (SNI, Server Name Indication), che viaggia in chiaro.
2. **ServerHello:** il server sceglie l'algoritmo (per esempio `TLS_AES_128_GCM_SHA256`) e manda la sua chiave pubblica. Da qui in poi tutto è cifrato: il server invia il suo **certificato**, una firma (CertificateVerify) e il messaggio Finished.
3. Il browser **verifica il certificato** e risponde con il suo Finished. Il canale è pronto e nel browser compare il lucchetto.

Il segreto condiviso: lo scambio Diffie-Hellman

Browser e server devono avere la stessa chiave segreta, ma l'attaccante vede tutto quello che si scambiano. Il metodo **Diffie-Hellman** (DH, dal nome dei due inventori) risolve il problema. L'analogia dei colori:

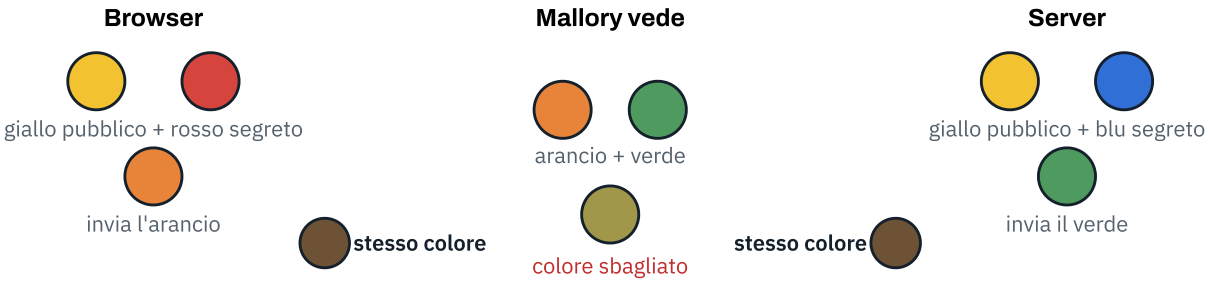


Figura 4 · Ognuno aggiunge il proprio colore segreto al colore ricevuto: entrambi ottengono giallo + rosso + blu, senza mai spedire i colori segreti. Mallory, mescolando ciò che vede, ottiene un altro colore.

Nella realtà i «colori» sono numeri enormi e il «mescolare» è un calcolo facile da fare ma praticamente impossibile da invertire. La chiave ottenuta serve poi a un algoritmo di cifratura veloce come l'**AES**

(Advanced Encryption Standard).

Il certificato e le Certificate Authority

Il certificato è la carta d'identità digitale del server: contiene il nome del sito, la sua chiave pubblica, il periodo di validità ed è **firmato** da una **CA** (Certificate Authority, autorità di certificazione). Il browser controlla che:

- la firma della CA sia autentica e la CA sia tra quelle di cui il browser si fida;
- il nome nel certificato coincida con il sito richiesto;
- il certificato sia nel suo periodo di validità;
- il server possieda davvero la chiave privata (lo dimostra con la firma CertificateVerify).

Perché Mallory non vince più

TENTATIVO DI MALLORY	CHE COSA SUCCEDE	PROPRIETÀ
Leggere il login	Vede solo byte casuali (`17 03 03 00 ab a5 c4 …`)	Riservatezza ✓
Cambiare un byte	Il **tag di autenticazione** di 16 byte (modalità GCM, Galois/Counter Mode) non coincide: il server scarta il record e chiude la connessione	Integrità ✓
Fingersi il server	Il suo certificato non è firmato da una CA fidata: il browser mostra l'avviso «La connessione non è privata»	Autenticità ✓

Che cosa vede ancora Mallory

Gli indirizzi IP, il nome del sito (SNI), quanti dati passano e quando. Non vede pagine, password e cookie.

Regole pratiche

- Davanti all'avviso «La connessione non è privata» non si clicca mai su «Procedi comunque».
- Sulle reti Wi-Fi pubbliche si inseriscono password solo su siti HTTPS.
- Molti siti usano **HSTS** (HTTP Strict Transport Security), che obbliga il browser a usare sempre HTTPS: così l'attaccante non può far partire la connessione in HTTP per poi dirottarla (attacco di SSL stripping).

9 Glossario delle sigle

ACK Acknowledgement · conferma di ricezione in TCP

AES Advanced Encryption Standard · algoritmo di cifratura simmetrica usato da TLS

ARP Address Resolution Protocol · trova il MAC conoscendo l'IP

CA Certificate Authority · autorità che firma i certificati digitali

CRC Cyclic Redundancy Check · calcolo per scoprire errori di trasmissione

DH Diffie-Hellman · metodo per concordare una chiave segreta su un canale pubblico

DNS Domain Name System · traduce i nomi di dominio in indirizzi IP

FCS Frame Check Sequence · coda del frame con il codice di controllo

FIN Finish · flag TCP per chiudere la connessione

FTP File Transfer Protocol · protocollo per trasferire file

GCM Galois/Counter Mode · modalità che unisce cifratura e tag di integrità

HSTS HTTP Strict Transport Security · obbliga il browser a usare sempre HTTPS

HTTP HyperText Transfer Protocol · protocollo del web, in chiaro

HTTPS HTTP Secure · HTTP dentro un canale cifrato TLS

ICMP Internet Control Message Protocol · messaggi di controllo e di errore (es. ping)

IP Internet Protocol · indirizzi e instradamento dei pacchetti

ISO International Organization for Standardization · ente internazionale per la normazione

JPEG Joint Photographic Experts Group · formato di compressione delle immagini

LAN Local Area Network · rete locale

MAC Media Access Control · indirizzo fisico della scheda di rete (in TLS «MAC» indica invece il codice di autenticazione del messaggio)

MITM Man In The Middle · attacco dell'uomo nel mezzo

NAT Network Address Translation · traduzione fra indirizzi privati e pubblici

OSI Open Systems Interconnection · modello di riferimento a sette livelli

PDU Protocol Data Unit · unità di dati di un livello

PSH Push · flag TCP: consegna subito i dati all'applicazione

RFC Request for Comments · documenti ufficiali degli standard di Internet

RPC Remote Procedure Call · chiamata di una funzione su un altro computer

SMTP Simple Mail Transfer Protocol · protocollo per inviare la posta

SNI Server Name Indication · nome del sito indicato nel ClientHello

SSL Secure Sockets Layer · predecessore di TLS, oggi abbandonato

SYN Synchronize · flag TCP per aprire la connessione

TCP Transmission Control Protocol · trasporto affidabile e ordinato

TLS Transport Layer Security · protocollo che cifra e autentica la comunicazione

TTL Time To Live · contatore che limita la vita del pacchetto

UDP User Datagram Protocol · trasporto veloce senza garanzie

URL Uniform Resource Locator · indirizzo completo di una risorsa web

UTF-8 Unicode Transformation Format, 8 bit · codifica dei caratteri in byte

UTP Unshielded Twisted Pair · cavo a doppini intrecciati non schermati

10 Domande di ripasso

1. Qual è la differenza fra client e server? Fai un esempio diverso dal browser.
2. Elenca i sette livelli del modello ISO/OSI dal basso verso l'alto e indica la PDU di ciascuno.
3. Perché si usa un modello a livelli? Indica almeno due vantaggi.
4. Che cosa si intende per incapsulamento? Che cosa aggiunge il livello Collegamento dati?
5. Il frame che esce dal client verso il server: qual è il suo MAC di destinazione e perché?
6. Quando un router inoltra un pacchetto, che cosa cambia e che cosa resta uguale?
7. A che cosa serve il TTL? Che cosa succede quando arriva a 0?
8. Descrivi il three-way handshake indicando flag, numeri di sequenza e di conferma dell'esempio.
9. Il server riceve una richiesta di 119 byte con seq = 4022: quale ack invia? Perché?
10. Differenze fra TCP e UDP; perché il DNS usa di solito UDP?
11. Scrivi una richiesta HTTP GET e indica le sue parti. Che cosa significano i codici 200, 301, 404, 500?
12. Che differenza c'è fra pagina statica e pagina dinamica?
13. Quali sono le tre proprietà di una comunicazione sicura? Con quale strumento le garantisce HTTPS?
14. Descrivi l'attacco man-in-the-middle e due modi per realizzarlo.
15. Spiega con l'analogia dei colori lo scambio Diffie-Hellman. Perché l'attaccante non ottiene la chiave?
16. Quali controlli fa il browser sul certificato del server?
17. Con HTTPS, che cosa può ancora vedere un attaccante?

Tracce di soluzione

1. Il client chiede, il server resta in ascolto e risponde. Esempio: l'app di posta (client) e il server di posta.
2. Fisico (bit), Collegamento dati (frame), Rete (pacchetto), Trasporto (segmento), Sessione, Presentazione e Applicazione (dati).
3. Compiti separati; si cambia un livello senza toccare gli altri; prodotti diversi funzionano insieme.
4. Ogni livello aggiunge un'intestazione ai dati del livello sopra. Il livello 2 aggiunge intestazione Ethernet (MAC destinazione e sorgente, tipo) e coda FCS.
5. Il MAC del router, perché il server è in un'altra rete e il frame vale solo per il tratto locale.
6. Cambiano i MAC, il TTL e quindi l'FCS; restano uguali IP, porte, numeri di sequenza e dati.
7. Limita la vita del pacchetto: cala di 1 a ogni router; a 0 il pacchetto viene scartato, così non gira all'infinito.
8. SYN seq = 4021; SYN-ACK seq = 8800 ack = 4022; ACK seq = 4022 ack = 8801.
9. ack = 4022 + 119 = 4141: indica il prossimo byte atteso.
10. TCP: connessione, affidabilità, ordine. UDP: niente connessione, più veloce. Il DNS fa domande brevi: se una si perde, basta ripeterla.
11. Riga di richiesta (metodo, percorso, versione), intestazioni, riga vuota. 200 trovato, 301 spostato, 404 non trovato, 500 errore del server.
12. La statica è un file pronto e uguale per tutti; la dinamica è generata da un programma a ogni richiesta.

13. Riservatezza (cifratura), integrità (tag di autenticazione), autenticità (certificato firmato da una CA).
14. L'attaccante si mette fra le due parti e vede o modifica i messaggi. Wi-Fi finto, ARP spoofing.
15. Ognuno mescola un segreto con un valore pubblico e spedisce il risultato; mescolando il proprio segreto con quello ricevuto entrambi ottengono lo stesso colore. Mallory vede solo i miscugli e non può separarli.
16. Firma di una CA fidata, nome del sito, validità, possesso della chiave privata (CertificateVerify).
17. IP, nome del sito (SNI), quantità e orari del traffico.

11 Mappa concettuale

