

Report Esercizio 4: Analisi del Traffico HTTP e HTTPS con Wireshark

A cura di Iris Canole, Federico Giannini, Daniele Castello, Luca Pani, Rosario Papa, Yari Olmi, Alessandro Salerno

Questo report documenta le procedure e i risultati dell'esercizio di laboratorio incentrato sulla cattura e l'analisi del traffico di rete **HTTP** e **HTTPS** utilizzando `tcpdump` e Wireshark. L'obiettivo primario dell'esercizio è osservare le differenze pratiche nella sicurezza dei dati trasmessi tra i due protocolli, evidenziando le vulnerabilità intrinseche del traffico in chiaro e l'efficacia della crittografia.

Parte 1: Cattura e Analisi del Traffico HTTP

La prima parte dell'esercizio si è concentrata sulla cattura di traffico HTTP **non crittografato**.

L'obiettivo era osservare in modo diretto come i dati, incluse informazioni sensibili come le credenziali di accesso, vengono trasmessi in chiaro attraverso la rete. Comprendere questa vulnerabilità è strategicamente importante per apprezzare i meccanismi di sicurezza implementati nei protocolli moderni.

La procedura ha previsto l'uso del comando `tcpdump` sull'interfaccia di rete `eth0` per catturare tutto il traffico generato durante un tentativo di login al sito di test `http://testphp.vulnweb.com/login.php`. I pacchetti catturati sono stati salvati nel file `httpdump.pcap` per la successiva analisi con Wireshark.

http					
No.	Time	Source	Destination	Protocol	Length Info
125	0.995664	192.168.50.10	142.250.186.131	OCSP	488 Request
126	0.996120	192.168.50.10	142.250.186.131	OCSP	488 Request
127	0.996439	192.168.50.10	142.250.186.131	OCSP	488 Request
155	1.113409	142.250.186.131	192.168.50.10	OCSP	1157 Response
161	1.136118	142.250.186.131	192.168.50.10	OCSP	1157 Response
162	1.136119	142.250.186.131	192.168.50.10	OCSP	1157 Response
1051	4.057638	192.168.50.10	34.107.221.82	HTTP	364 GET /success.txt?ipv4 HTTP/1.1
1056	4.120135	34.107.221.82	192.168.50.10	HTTP	270 HTTP/1.1 200 OK (text/plain)
1159	22.555881	192.168.50.10	44.228.249.3	HTTP	402 GET /Login.php HTTP/1.1
1167	22.926723	44.228.249.3	192.168.50.10	HTTP	1342 HTTP/1.1 200 OK (text/html)
1169	23.095540	192.168.50.10	44.228.249.3	HTTP	371 GET /style.css HTTP/1.1
1173	23.304265	44.228.249.3	192.168.50.10	HTTP	1395 HTTP/1.1 200 OK (text/css)
1178	23.351360	192.168.50.10	44.228.249.3	HTTP	431 GET /images/logo.gif HTTP/1.1
1180	23.523061	192.168.50.10	44.228.249.3	HTTP	424 GET /favicon.ico HTTP/1.1
1183	23.729749	44.228.249.3	192.168.50.10	HTTP	1189 HTTP/1.1 200 OK (image/x-icon)
1189	23.766477	44.228.249.3	192.168.50.10	HTTP	1114 HTTP/1.1 200 OK (GIF89a)
1530	53.772084	192.168.50.10	44.228.249.3	HTTP	580 POST /userinfo.php HTTP/1.1 (application/x-www-form-urlencoded)
1532	54.003718	44.228.249.3	192.168.50.10	HTTP	330 HTTP/1.1 302 Found (text/html)
1534	54.012981	192.168.50.10	44.228.249.3	HTTP	449 GET /login.php HTTP/1.1
1538	54.290781	44.228.249.3	192.168.50.10	HTTP	1342 HTTP/1.1 200 OK (text/html)


```
> Frame 1530: Packet, 580 bytes on wire (4640 bits), 580 bytes captured (4640 bits)
> Ethernet II, Src: PCSSystemtec_d1:f8:5d (08:00:27:d1:f8:5d), Dst: 52:54:00:12:35:00 (52:54:00:12:35:00)
> Internet Protocol Version 4, Src: 192.168.50.10, Dst: 44.228.249.3
> Transmission Control Protocol, Src Port: 35062, Dst Port: 80, Seq: 378, Ack: 6901, Len: 526
> Hypertext Transfer Protocol
> HTML Form URL Encoded: application/x-www-form-urlencoded
  > Form item: "uname" = "Admin"
  > Form item: "pass" = "Admin"
```

Quali due informazioni vengono visualizzate?

L'analisi del file di cattura in Wireshark, e in particolare del pacchetto 1530, rivela la trasmissione delle credenziali di accesso in formato **plaintext (testo in chiaro)**. Espandendo i dettagli del protocollo HTTP, nella sezione "HTML Form URL Encoded" sono chiaramente visibili le due informazioni inviate dal client al server, associate ai rispettivi nomi dei campi del form:

- uname : Admin
- pass : Admin

Questa osservazione dimostra in modo inequivocabile come **chiunque** sia in grado di **intercettare** il traffico di rete (*eavesdropping*) possa leggere le credenziali senza alcuna difficoltà. Il chiaro rischio per la sicurezza evidenziato da questa cattura sottolinea la necessità di meccanismi di protezione, che verranno analizzati nella parte successiva dedicata al traffico crittografato.

Parte 2: Cattura e Analisi del Traffico HTTPS

Questa seconda parte dell'esercizio si è focalizzata sull'analisi del traffico HTTPS per dimostrare come l'implementazione della crittografia protegga efficacemente i dati in transito. L'obiettivo era confrontare i risultati ottenuti con quelli della precedente analisi HTTP, evidenziando il ruolo del livello di **sicurezza aggiuntivo**.

La procedura è stata analoga alla precedente, catturando il traffico generato durante l'accesso a un sito che utilizza HTTPS. Il traffico è stato salvato nel file `httpsdump.pcap`. All'interno di Wireshark, è stato applicato il filtro `tcp.port==443` per isolare e analizzare esclusivamente i pacchetti relativi alla sessione HTTPS.

tcp.port==443					
No.	Time	Source	Destination	Protocol	Length Info
37	11.248835	192.168.50.10	34.160.144.191	TCP	74 47872 -> 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TStamp=927043242 TSectr=0 WS=1024
38	11.295122	34.160.144.191	192.168.50.10	TCP	60 443 -> 47872 [SYN, ACK] Seq=0 Ack=1 Win=32768 Len=0 MSS=1460
39	11.295269	192.168.50.10	34.160.144.191	TCP	54 47872 -> 443 [ACK] Seq=1 Ack=1 Win=64240 Len=0
40	11.295803	192.168.50.10	34.160.144.191	TLSv1.2	274 Client Hello (SNI=content-signature-2.cdn.mozilla.net)
41	11.322702	34.160.144.191	192.168.50.10	TCP	60 443 -> 47872 [ACK] Seq=1 Ack=221 Win=32548 Len=0
42	11.366495	34.160.144.191	192.168.50.10	TLSv1.2	2974 Server Hello, Certificate
43	11.366659	192.168.50.10	34.160.144.191	TCP	54 47872 -> 443 [ACK] Seq=221 Ack=2921 Win=65535 Len=0
44	11.368869	34.160.144.191	192.168.50.10	TLSv1.2	154 Server Key Exchange, Server Hello Done
45	11.368892	192.168.50.10	34.160.144.191	TCP	54 47872 -> 443 [ACK] Seq=221 Ack=3021 Win=65535 Len=0
46	11.390132	192.168.50.10	34.160.144.191	TLSv1.2	147 Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
47	11.444622	34.160.144.191	192.168.50.10	TLSv1.2	434 New Session Ticket, Change Cipher Spec, Encrypted Handshake Message, Application Data
48	11.470810	192.168.50.10	34.160.144.191	TLSv1.2	153 Application Data
49	11.473633	192.168.50.10	34.160.144.191	TLSv1.2	92 Application Data
50	11.474197	34.160.144.191	192.168.50.10	TCP	60 443 -> 47872 [ACK] Seq=3401 Ack=451 Win=32318 Len=0
51	11.536724	34.160.144.191	192.168.50.10	TLSv1.2	92 Application Data
52	11.581999	192.168.50.10	34.160.144.191	TCP	54 47872 -> 443 [ACK] Seq=451 Ack=3439 Win=65535 Len=0

Frame 48: Packet, 153 bytes on wire (1224 bits), 153 bytes captured (1224 bits)
Ethernet II, Src: PCSystemtec_d1:f8:5d (08:00:27:d1:f8:5d), Dst: 52:54:00:12:35:00 (52:54:00:12:35:00)
Internet Protocol Version 4, Src: 192.168.50.10, Dst: 34.160.144.191
Transmission Control Protocol, Src Port: 47872, Dst Port: 443, Seq: 314, Ack: 3401, Len: 99

Transport Layer Security
[Stream Index: 0]
TLSv1.2 Record Layer: Application Data Protocol: HyperText Transfer Protocol 2

Cosa ha sostituito la sezione HTTP che era nel file di cattura precedente?

L'analisi del pacchetto rivela che i dati dell'applicazione non sono più direttamente visibili sotto il

livello TCP. La sezione HTTP in chiaro, presente nella cattura precedente, è stata sostituita da un nuovo livello protocollare: **Transport Layer Security (TLS)**. Più specificamente, come mostrato nei dettagli del pacchetto, i dati sono incapsulati all'interno del **TLSv1.2 Record Layer**, che gestisce la crittografia dei dati dell'applicazione prima della loro trasmissione.

tcp.port==443						
No.	Time	Source	Destination	Protocol	Length	Info
37	11.248835	192.168.50.10	34.160.144.191	TCP	74	47872 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=927043242 TSecr=0 WS=1024
38	11.295122	34.160.144.191	192.168.50.10	TCP	60	443 → 47872 [SYN, ACK] Seq=0 Ack=1 Win=32768 Len=0 MSS=1460
39	11.295269	192.168.50.10	34.160.144.191	TCP	54	47872 → 443 [ACK] Seq=1 Ack=1 Win=64240 Len=0
40	11.295883	192.168.50.10	34.160.144.191	TLSv1.2	274	Client Hello (SNI=content-signature-2.cdn.mozilla.net)
41	11.322702	34.160.144.191	192.168.50.10	TCP	60	443 → 47872 [ACK] Seq=1 Ack=221 Win=32548 Len=0
42	11.366495	34.160.144.191	192.168.50.10	TLSv1.2	2974	Server Hello, Certificate
43	11.366659	192.168.50.10	34.160.144.191	TCP	54	47872 → 443 [ACK] Seq=221 Ack=2921 Win=65535 Len=0
44	11.368869	34.160.144.191	192.168.50.10	TLSv1.2	154	Server Key Exchange, Server Hello Done
45	11.368892	192.168.50.10	34.160.144.191	TCP	54	47872 → 443 [ACK] Seq=221 Ack=3021 Win=65535 Len=0
46	11.390132	192.168.50.10	34.160.144.191	TLSv1.2	147	Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
47	11.444622	34.160.144.191	192.168.50.10	TLSv1.2	434	New Session Ticket, Change Cipher Spec, Encrypted Handshake Message, Application Data
48	11.470810	192.168.50.10	34.160.144.191	TLSv1.2	153	Application Data
49	11.473633	192.168.50.10	34.160.144.191	TLSv1.2	92	Application Data
50	11.474197	34.160.144.191	192.168.50.10	TCP	60	443 → 47872 [ACK] Seq=3401 Ack=451 Win=32318 Len=0
51	11.536724	34.160.144.191	192.168.50.10	TLSv1.2	92	Application Data
52	11.581999	192.168.50.10	34.160.144.191	TCP	54	47872 → 443 [ACK] Seq=451 Ack=3439 Win=65535 Len=0

Frame 48: Packet, 153 bytes on wire (1224 bits), 153 bytes captured (1224 bits)
Ethernet II, Src: PCSSystemec_d1:f8:5d (08:00:27:d1:f8:5d), Dst: 52:54:00:12:35:00 (52:54:00:12:35:00)
Internet Protocol Version 4, Src: 192.168.50.10, Dst: 34.160.144.191
Transmission Control Protocol, Src Port: 47872, Dst Port: 443, Seq: 314, Ack: 3401, Len: 99
Transport Layer Security
[Stream index: 0]
TLSv1.2 Record Layer: Application Data Protocol: HyperText Transfer Protocol 2
Content Type: Application Data (23)
Version: TLS 1.2 (0x0303)
Length: 94
Encrypted Application Data: 00000000000000001503990c0520ff5d0b7f9d2109ec325c5d9e4b5c8a0e45229e3493e18ce8fbcc10e4938cf2ad973b648796306449b656f23af708da8ff5fb134374a4985...
[Application Data Protocol: HyperText Transfer Protocol 2]

I dati dell'applicazione sono in formato plaintext o leggibile?

Esaminando il contenuto del livello TLS, si osserva un campo denominato "**Encrypted Application Data**". Il contenuto di questo campo è una stringa di byte esadecimale non interpretabile da un essere umano. Si può quindi concludere in modo definitivo che i dati dell'applicazione **non sono in plaintext e non sono leggibili**.

Tuttavia, un'analisi più approfondita rivela un dettaglio significativo: lo strumento di analisi di Wireshark è in grado di identificare il tipo di protocollo incapsulato nel flusso crittografato, segnalandolo come **[Application Data Protocol: HyperText Transfer Protocol 2]**. Questo dimostra che, sebbene il contenuto dei dati sia protetto e illeggibile (*ciphertext*), i moderni analizzatori di protocollo possono comunque derivare metadati sul tipo di traffico che transita all'interno del tunnel TLS.

Questo risultato dimostra come HTTPS riesca a nascondere i dati, garantendo la confidenzialità della comunicazione e mitigando i rischi di intercettazione visti nell'analisi del traffico HTTP.

Domande di Riflessione

Questa sezione finale passa dall'osservazione pratica dei pacchetti alla riflessione sulle implicazioni concettuali e di sicurezza più ampie. Sulla base delle prove raccolte durante l'esercizio, vengono analizzati i vantaggi di HTTPS e i limiti della sua garanzia di affidabilità.

Vantaggi dell'Uso di HTTPS

Quali sono i vantaggi dell'uso di HTTPS invece di HTTP?

Il vantaggio principale e fondamentale dell'uso di HTTPS rispetto a HTTP è la **crittografia**. Come dimostrato empiricamente in questo laboratorio, HTTPS utilizza un protocollo di sicurezza (TLS) per applicare un algoritmo matematico che nasconde il vero significato dei dati scambiati.

Questo si traduce in un beneficio cruciale:

- **Confidenzialità:** Impedisce a terzi di intercettare e leggere informazioni sensibili in transito (*eavesdropping*). La cattura HTTPS ha mostrato dati illeggibili ("Encrypted Application Data"), mentre la cattura HTTP ha esposto le credenziali di login in chiaro.

Affidabilità dei Siti HTTPS

Tutti i siti web che usano HTTPS sono considerati affidabili?

No, non tutti i siti che usano HTTPS sono necessariamente affidabili. Sebbene HTTPS garantisca che la *comunicazione* tra il browser e il server del sito sia crittografata e sicura da intercettazioni, **non fornisce alcuna garanzia sull'affidabilità o sulle intenzioni del sito stesso**.

Come indicato nel materiale dell'esercizio, "Solo perché un sito usa HTTPS non significa che sia un sito affidabile". Gli attori malevoli utilizzano comunemente HTTPS proprio per nascondere le loro attività illecite, il che significa che la crittografia protegge la comunicazione dannosa tanto quanto quella legittima.

Conclusione

Questo esercizio di laboratorio ha dimostrato con successo il ruolo essenziale di HTTPS nel proteggere la confidenzialità dei dati in transito. L'analisi comparativa dei pacchetti ha evidenziato la vulnerabilità critica di HTTP, che espone dati sensibili in chiaro, e la robusta protezione offerta dalla crittografia di HTTPS. In sintesi, l'attività ha confermato il ruolo indispensabile di HTTPS nel garantire la confidenzialità dei dati, un principio fondamentale della sicurezza delle comunicazioni di rete.