

# BUILD WEEK III

“Esercizio 4 - Usare Wireshark per Esaminare il Traffico HTTP e HTTPS”

## PARTE I - Catturare e Visualizzare il Traffico HTTP

Dopo aver avviato Kali Linux ed aver confermato la sua effettiva connessione alla rete, procediamo ad avviare *tcpdump* per la cattura il contenuto del traffico HTTP:

```
(kali㉿kali)-[~]
$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:d1:f8:5d brd ff:ff:ff:ff:ff:ff
    inet 192.168.178.60/24 brd 192.168.178.255 scope global dynamic noprefixroute eth0
        valid_lft 863958sec preferred_lft 863958sec
    inet6 fd17:625c:f037:2:6df7:7f0:c11d:a28f/64 scope global deprecated dynamic noprefixroute
        valid_lft 7167sec preferred_lft 0sec
    inet6 fd26:5b8a:3700:0:7c78:77f8:f11e:7a5a/64 scope global dynamic noprefixroute
        valid_lft 7167sec preferred_lft 3567sec
    inet6 fe80::1d10:1873:1516:57/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
```

-interfaccia di rete-

```
(kali㉿kali)-[~]
$ sudo tcpdump -i eth0 -s 0 -w catturahttp.pcap
[sudo] password for kali:
tcpdump: listening on eth0, link-type EN10MB (Ethernet), snapshot length 262144 bytes
```

-messa in ascolto di tcpdump-

Dopo aver messo in ascolto *tcpdump*, mediante la stringa che possiamo osservare nell'immagine soprastante, nel dettaglio:

- *sudo* Richiede i permessi di superutente
- *tcpdump* specifica il comando
- *-i eth0* specifica l'interfaccia da ascoltare
- *-s 0* imposta lo snaplen (se 0 cattura l'intero pacchetto)
- *-w catturahttp.pcap* scrive l'output del file

e procediamo con la navigazione verso il sito richiesto:

*<http://testphp.vulnweb.com/login.php>*

ed inserisco nei campi Username e Password → Admin:

### N.B.

Da notare come venga specificato (visualizzabile nell'immagine) come la connessione non sia sicura.

search art

[Browse categories](#)

[Browse artists](#)

[Your cart](#)

[Signup](#)

[Your profile](#)

[Our guestbook](#)

[AJAX Demo](#)

**Links**

[Security art](#)

[PHP scanner](#)

[PHP vuln help](#)

[Fractal Explorer](#)

If you are already registered please enter your login information below:

Username :

Password :

You can also  
Signup disabled

This connection is not secure.

Logins entered here could be compromised. [Learn More](#)

the password **test**.

**Warning:** This is not a real shop. This is an example PHP application, which is intentionally vulnerable to web attacks. It is intended to help you test Acunetix. It also helps you understand how developer errors and bad configuration may let someone break into your website. You can use it to test other tools and your manual hacking skills as well. Tip: Look for potential SQL Injections, Cross-site Scripting (XSS), and Cross-site Request Forgery (CSRF), and more.

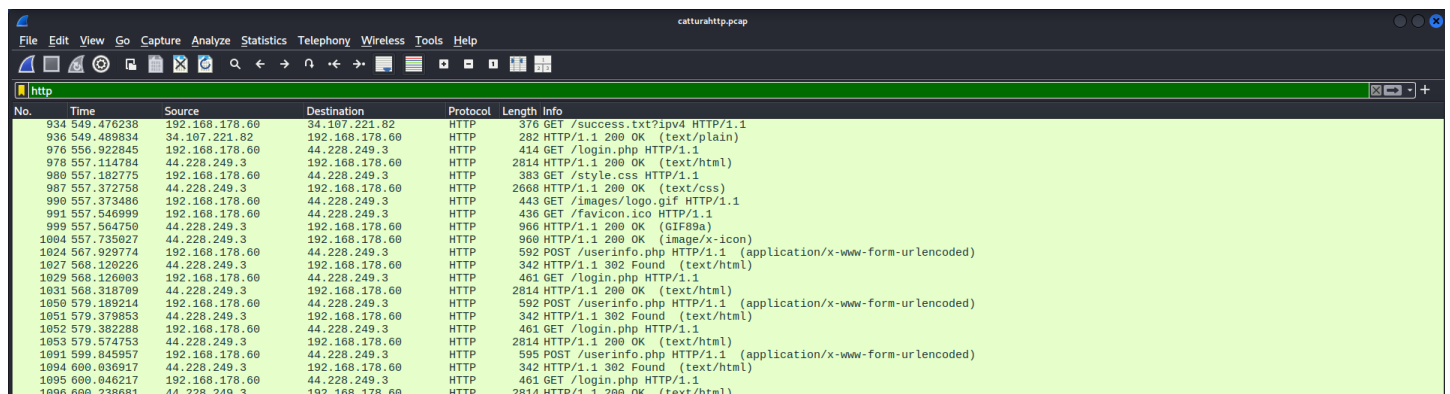
-connessione al sito richiesto-

Ora, andiamo a chiudere il browser e, torniamo al terminale per fermare la cattura dei pacchetti:

```
(kali@kali)-[~]
$ sudo tcpdump -i eth0 -s 0 -w catturahttp.pcap
[sudo] password for kali:
tcpdump: listening on eth0, link-type EN10MB (Ethernet), snapshot length 262144 bytes
^C1376 packets captured
1376 packets received by filter
0 packets dropped by kernel
```

-stop della cattura dei pacchetti-

Per esaminare i pacchetti catturati, procediamo ad avviare *Wireshark* ed andiamo a selezionare il file appena creato, applicando il filtro *http* nella barra, per ricercare i dati di nostro interesse:



No.	Time	Source	Destination	Protocol	Length	Info
934	549.476238	192.168.178.60	34.107.221.82	HTTP	376	GET /success.txt?ip=4 HTTP/1.1
936	549.489834	34.107.221.82	192.168.178.60	HTTP	282	HTTP/1.1 200 OK (text/plain)
976	556.922845	192.168.178.60	44.228.249.3	HTTP	414	GET /login.php HTTP/1.1
978	557.114784	44.228.249.3	192.168.178.60	HTTP	2814	HTTP/1.1 200 OK (text/html)
980	557.182775	192.168.178.60	44.228.249.3	HTTP	383	GET /style.css HTTP/1.1
987	557.372758	44.228.249.3	192.168.178.60	HTTP	2668	HTTP/1.1 200 OK (text/css)
989	557.373486	192.168.178.60	44.228.249.3	HTTP	443	GET /images/logo.gif HTTP/1.1
991	557.548999	192.168.178.60	44.228.249.3	HTTP	436	GET /favicon.ico HTTP/1.1
999	557.564750	44.228.249.3	192.168.178.60	HTTP	966	HTTP/1.1 200 OK (GIF89a)
1004	557.735827	44.228.249.3	192.168.178.60	HTTP	960	HTTP/1.1 200 OK (image/x-icon)
1024	567.929774	192.168.178.60	44.228.249.3	HTTP	592	POST /userinfo.php HTTP/1.1 (application/x-www-form-urlencoded)
1027	568.120226	44.228.249.3	192.168.178.60	HTTP	342	HTTP/1.1 302 Found (text/html)
1029	568.126803	192.168.178.60	44.228.249.3	HTTP	461	GET /login.php HTTP/1.1
1031	568.318709	44.228.249.3	192.168.178.60	HTTP	2814	HTTP/1.1 200 OK (text/html)
1050	579.189214	192.168.178.60	44.228.249.3	HTTP	592	POST /userinfo.php HTTP/1.1 (application/x-www-form-urlencoded)
1051	579.379853	44.228.249.3	192.168.178.60	HTTP	342	HTTP/1.1 302 Found (text/html)
1052	579.382208	192.168.178.60	44.228.249.3	HTTP	461	GET /login.php HTTP/1.1
1053	579.574753	44.228.249.3	192.168.178.60	HTTP	2814	HTTP/1.1 200 OK (text/html)
1091	599.845957	192.168.178.60	44.228.249.3	HTTP	595	POST /userinfo.php HTTP/1.1 (application/x-www-form-urlencoded)
1094	600.036917	44.228.249.3	192.168.178.60	HTTP	342	HTTP/1.1 302 Found (text/html)
1095	600.046217	192.168.178.60	44.228.249.3	HTTP	461	GET /login.php HTTP/1.1
1096	600.238681	44.228.249.3	192.168.178.60	HTTP	2814	HTTP/1.1 200 OK (text/html)

-filtraggio dei pacchetti su Wireshark (http)-

Per visualizzare i dati di nostro interesse, andiamo, selezionando un messaggio di POST ad espandere la sezione:

*HTML Form URL Encoded: application/x-www-form-urlencoded*

Osservando, potremo notare alcune informazioni di interesse:

```

> Frame 1024: 592 bytes on wire (4736 bits), 592 bytes captured (4736 bits)
> Ethernet II, Src: PCSSystemtec_d1:f8:5d (08:00:27:d1:f8:5d), Dst: AVMAudiovisu_90:de:a8 (48:5d:35:90:de:a8)
> Internet Protocol Version 4, Src: 192.168.178.60, Dst: 44.228.249.3
> Transmission Control Protocol, Src Port: 43108, Dst Port: 80, Seq: 726, Ack: 9649, Len: 526
> Hypertext Transfer Protocol
  > POST /userinfo.php HTTP/1.1\r\n
    Host: testphp.vulnweb.com\r\n
    User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0\r\n
    Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8\r\n
    Accept-Language: en-US,en;q=0.5\r\n
    Accept-Encoding: gzip, deflate\r\n
    Content-Type: application/x-www-form-urlencoded\r\n
  > Content-Length: 22\r\n
    Origin: http://testphp.vulnweb.com\r\n
    Connection: keep-alive\r\n
    Referer: http://testphp.vulnweb.com/login.php\r\n
    Upgrade-Insecure-Requests: 1\r\n
    Priority: u=0, i\r\n
    \r\n
    [Response in frame: 1027]
    [Full request URI: http://testphp.vulnweb.com/userinfo.php]
    File Data: 22 bytes
  > HTML Form URL Encoded: application/x-www-form-urlencoded
    > Form item: "uname" = "Admin"
    > Form item: "pass" = "Admin"

```

*-analisi approfondita del pacchetto POST-*

### **DOMANDA:**

*Quali due informazioni vengono visualizzate?*

Mediante questa procedura, potremo visualizzare username (*uname*) e password (*pass*) trasmessi come parte integrante del pacchetto.

## PARTE II - Catturare e Visualizzare il Traffico HTTPS

Come fatto precedentemente, andiamo ad aprire il terminale per catturare questa volta il traffico HTTPS:

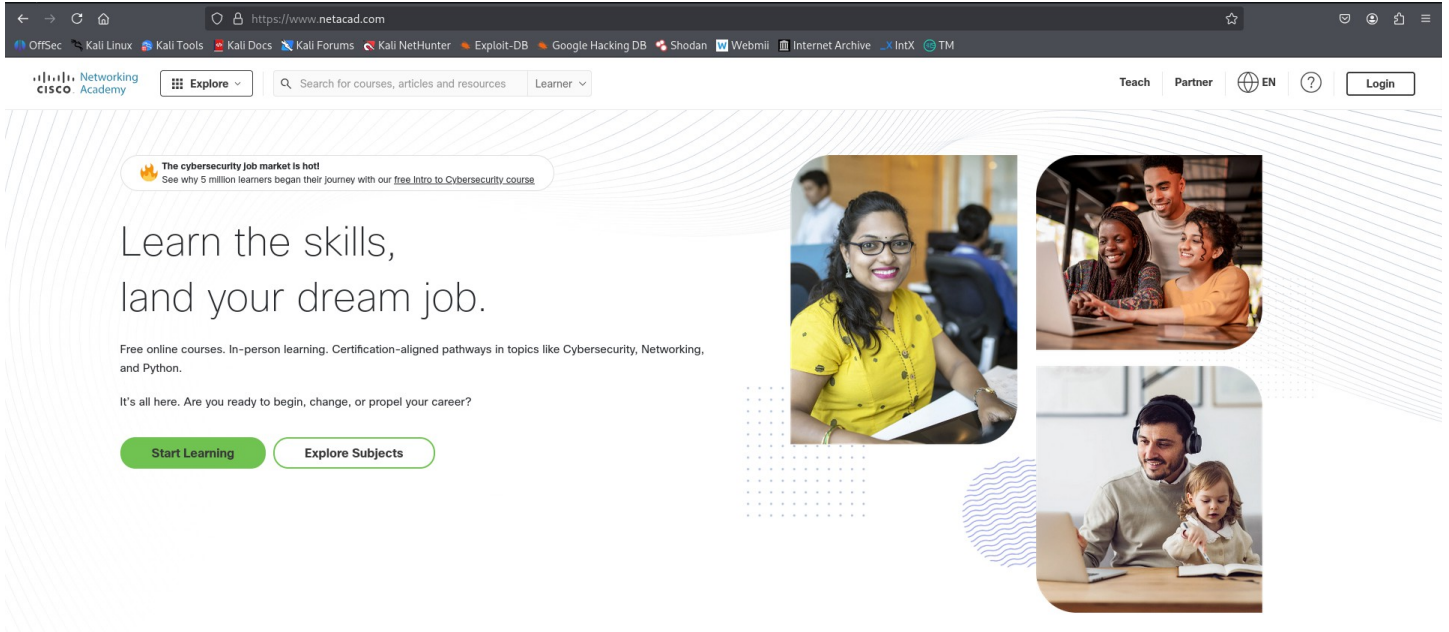
```

(kali@kali)-[~]
$ sudo tcpdump -i eth0 -s 0 -w catturahttps.pcap
[sudo] password for kali:
tcpdump: listening on eth0, link-type EN10MB (Ethernet), snapshot length 262144 bytes

```

*-cattura del traffico e creazione pacchetto catturahttps.pcap-*

Ora, apriamo il browser e navighiamo verso il sito di interesse (www.netacad.com):



-navigazione verso il sito-

## DOMANDA

Cosa noti riguardo all'URL del sito web?

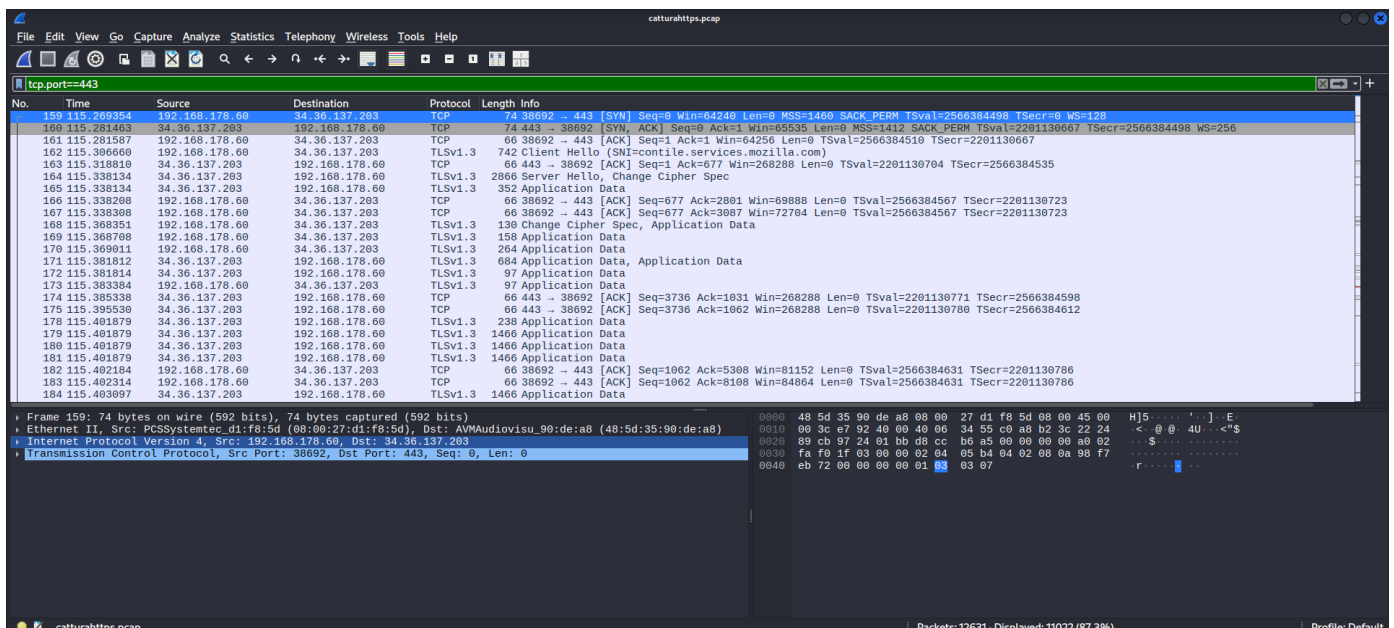
Possiamo subito notare come l'URL inizi con *https://*, indicando che si che il sito utilizzi il protocollo *Hypertext Transfer Protocol Secure* (HTTPS), che crittografa la comunicazione.

Dopo aver fatto il *login* nell'apposita pagina, andremo a chiudere il browser ed, in seguito il terminale per procedere con l'analisi del file contenente i pacchetti.

```
(kali㉿kali)-[~]
└─$ sudo tcpdump -i eth0 -s 0 -w catturahttps.pcap
[sudo] password for kali:
tcpdump: listening on eth0, link-type EN10MB (Ethernet), snapshot length 262144 bytes
^C12631 packets captured
12675 packets received by filter
0 packets dropped by kernel
```

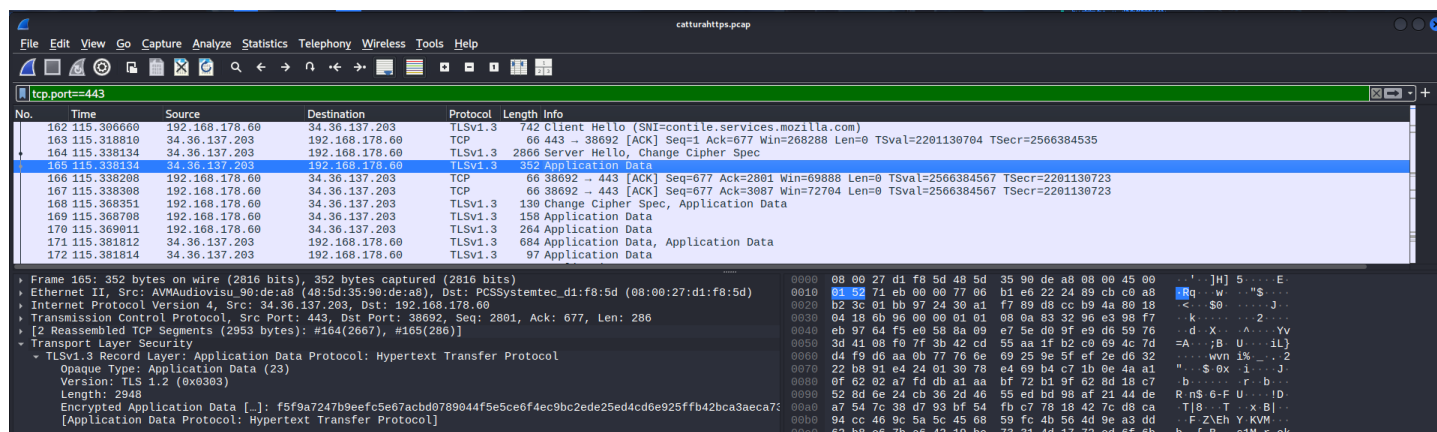
-chiusura della cattura dei pacchetti-

Dopo l'apertura di Wireshark, andiamo ad aprire il file appena creato, filtrando la ricerca sulla porta 443 (porta utilizzata per il traffico HTTPS):



-apertura di Wireshark e filtro porta 443 (HTTPS)-

Selezionando un pacchetto Application Data (Dati Applicazione) potremo andare ad esaminare nel dettaglio i cambiamenti rispetto alla precedente protocollo:



*-selezione pacchetto Application Data ed apertura dei dettagli-*

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

Le sezioni di protocollo di livello superiore (HTTP e HTML Form URL encoded) sono state sostituite da:

TLS (*Transport Layer Security*) nel questo caso, versione 1.2

Sotto TLS, possiamo notare che si trova la sezione *Encrypted Application Data* (dati applicazione crittografati)

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

I dati sono in formato crittografato (mostrati come stringa esadecimale), non sono in plaintext (testo in chiaro) e quindi non sono leggibili.

## DOMANDE DI RIFLESSIONE

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

I vantaggi principali dell'uso di HTTPS sono:

- Crittografia I dati scambiati tra browser e server sono crittografati. Se venissero intercettati, apparirebbero come dati incomprensibili e illeggibili. Questo impedisce che informazioni sensibili siano compromesse.
- Integrità HTTPS include meccanismi per rilevare se i dati sono stati alterati durante la trasmissione. Se un utente tentasse di modificare i dati in transito, il browser o il server lo rileveranno
- Autenticazione HTTPS utilizza certificati digitali per verificare l'identità del server. Ciò assicura che gli utenti si stiano collegando al sito web legittimo e non un sito falso gestito da malintenzionati

*Tutti i siti web che usano HTTPS sono considerati affidabili?*

No, non tutti. L'uso di HTTPS garantisce che la connessione sia sicura e che i dati non possano essere intercettati/alterati. Tuttavia, questo non garantisce la legittimità o l'affidabilità del sito stesso.

## NOTE FINALI

E' sempre bene verificare la reputazione e l'URL del sito per determinarne l'affidabilità complessiva.