

DevSecOps

The importance of integrating security measures throughout development.

<https://linkedin.com/in/fvecchi07>



01

INPUT SANITIZATION: XSS

Exploiting unsanitized inputs
with reflected XSS

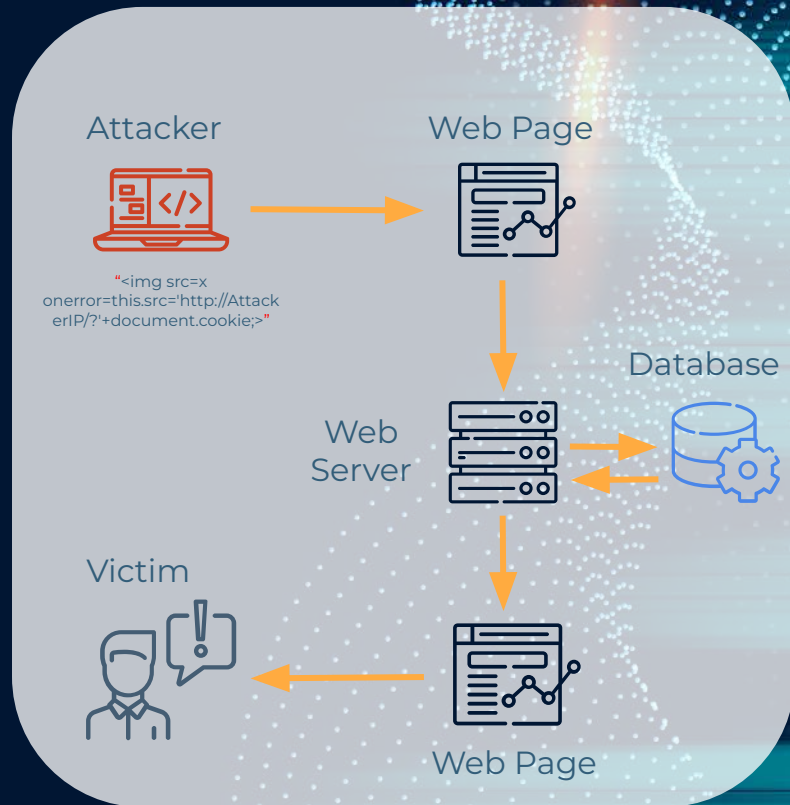
Exploiting Unsanitized Fields with XSS

Frontend - Making the request

```
userComment = "<h1>Hi<h1>" #unsanitized input field  
  
def requestHandler(url, input):  
    try:  
        res = requests.post(url, input)  
    catch:  
        print("error")  
  
requestHandler("localhost:3000/api/comments", userComment)
```

Walkthrough

- Attacker makes malicious request
- Frontend handles that request as code (injecting the html or javascript)
- Frontend calls the backend API, storing the malicious code in the DB
- Victim views the webpage, infecting his/her machine





02

INPUT SANITIZATION: SQL

How unsanitized inputs can
lead to data leaks

Exploiting Unsanitized Fields with SQL Injections

Login

Email

Password

- Password field will always return 'true'
- The rest of the code will not run because of the '--'
- The first speech mark closes the input field, leaving it empty
- The injectable field can be used to query the database and leak files

```
SELECT *
FROM users
WHERE email = 'user@email.com'
AND pass = '' or 1=1--' LIMIT 1
```



03

Broken Authentication

Bypassing cookie restrictions
with broken authentication


Exploiting Broken Authentication

Types of Broken Authentication

- Insecure Cookies
- Unsafe password policy
- No request restrictions
- Unhashed passwords

Solutions to Integrate

- Hash passwords in database
- Create unique & hashed cookies for the user
- Log requests and set limiter
- Set password policy



The screenshot shows the Chrome DevTools Application tab. The 'Application' tab is selected, and the 'Cookies' section is visible. A table lists cookies with columns for 'Name' and 'Value'. One cookie is highlighted with a red border: 'privilege' with the value 'user'. A red line points from this entry to the text below.

Name	Value
privilege	user

Cookie value can be changed to 'admin'

Insecure Design

- User can change token value to something easily predictable
- Cookie value is displayed in plain text and not encrypted or unique to each user



04

Directory Exploits

Exploiting an LFI Vulnerability on a Linux server to dump shadow contents

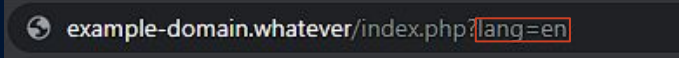
Exploiting Unrestricted Directories with LFI

LFI Vulnerable PHP Code

```
<?php
$file = $_GET['file'];
if(isset($file))
{
    include("pages/$file");
}
else
{
    include("index.php");
}
?>
```

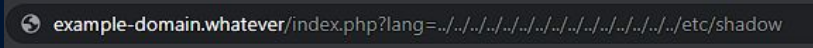
Why it's Vulnerable

PHP LFI Vulnerabilities are less common, however they are easy to exploit and can cause severe damage to victims. On line 5 of the LFI Vulnerable code, it allows a user to make a request to any directory using path traversal.



example-domain.whatever/index.php?lang=en

Parameter 'lang' is vulnerable to LFI



example-domain.whatever/index.php?lang=../../../../../../../../../../../../etc/shadow

'index.php?lang=' exploited to output contents of '/etc/shadow' via directory traversal.



05

Prevention Methods

Security Measures we can take to prevent these vulnerabilities

Listen to your Package Manager

removed 2 packages, changed 30 packages, and audited 1416 packages in 1m

225 packages are looking for funding
run `npm fund` for details

6 **high** severity vulnerabilities

To address all issues (including breaking changes), run:
npm audit fix --force

```
css-select <=3.1.0
Depends on vulnerable versions of nth-check
node_modules/svgo/node_modules/css-select
svgo 1.0.0 - 1.3.2
Depends on vulnerable versions of css-select
node_modules/svgo
@svgr/plugin-svgo <=5.5.0
Depends on vulnerable versions of svgo
node_modules/@svgr/plugin-svgo
@svgr/webpack 4.0.0 - 5.5.0
Depends on vulnerable versions of @svgr/plugin-svgo
node_modules/@svgr/webpack
```

```
~/projects/private/vulnerable-repository/dotnet % dotnet list package --vulnerab
le

The following sources were used:
https://api.nuget.org/v3/index.json

The given project 'VulnerableRepo.CLI' has no vulnerable packages given the curr
ent sources.
The given project 'VulnerableRepo.Application' has no vulnerable packages given
the current sources.
Project 'VulnerableRepo.Infrastructure' has the following vulnerable packages
[net6.0]:
Top-level Package      Requested  Resolved  Severity  Advisory URL
> UmbracoForms         8.4.0     8.4.0     High      https://github.com/a
dvisories/OHSA-8m73-w2r2-6xxj

The given project 'VulnerableRepo.Domain' has no vulnerable packages given the c
urrent sources.
```



Get a Pentest Scheduled

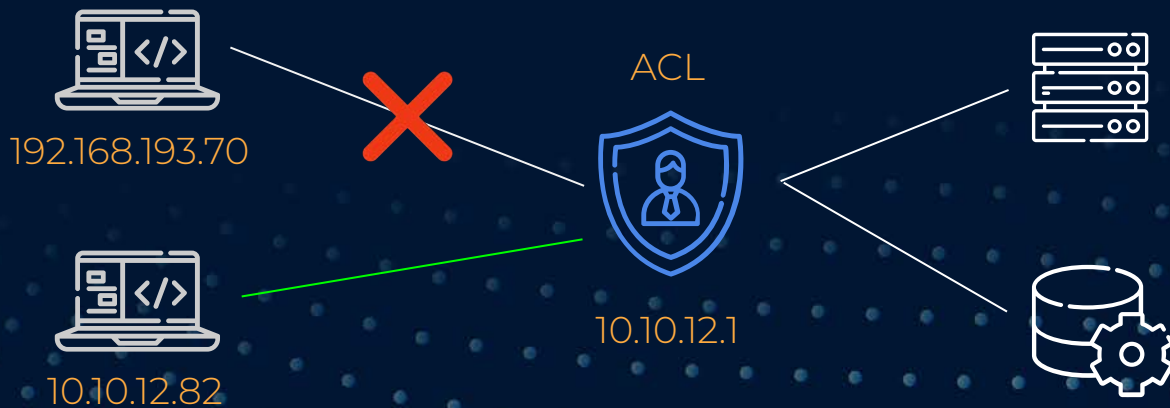
You could find freelancers, agencies, or full time pentesters to secure your website.

What if Pentests are out the Budget?

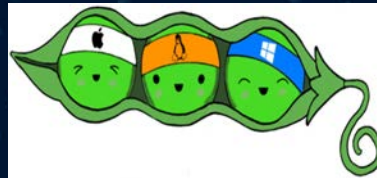
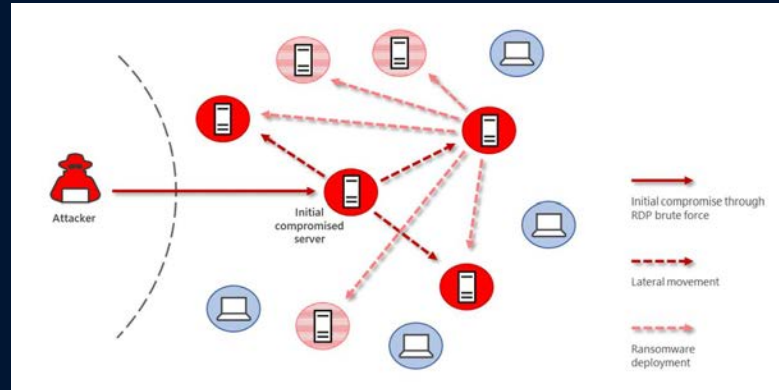


Setup an Access Control List

```
NSE: [mysql-vuln-cve2012-21 [CENSORED IP] 106] Connection attempt #1  
NSE: mysql-vuln-cve2012-2122 against [CENSORED IP] threw an error!  
Host '192.168.193.70' is not allowed to connect to this MySQL server
```



Prevent advanced Lateral Movement



```
Interesting Files
[+] SUID - Check easy privesc, exploits and write perms
[1] https://book.hacktricks.xyz/linux-unix/privilege-escalation#sudo-and-suid
-rwsr-xr-x 1 root root 63K Feb 7 2020 /usr/bin/passwd --> Apple_Mac_OSX(03-2006)/Solaris_8/9(12-2004)/SPM
MC_8/9/4ms_Solaris_2.2.10_2.2.1(02-1997)
-rwsr-xr-x 1 root root 44K Feb 7 2020 /usr/bin/passwd --> HP-UX_10.20
-rwsr-xr-x 1 root root 87K Feb 7 2020 /usr/bin/gpasswd
-rwsr-xr-x 1 root root 52K Feb 7 2020 /usr/bin/chsh
-rwsr-xr-x 1 root root 58K Feb 7 2020 /usr/bin/chfn --> Sudo_9.2/10
-rwsr-xr-x 1 root dip 395K Jan 7 01:10 /usr/sbin/pppd --> Apple_Mac_OSX_10.4.0(03-2007)
-rwsr-xr-x 1 root root 35K Feb 7 15:38 /usr/bin/mount --> BSD/Linux(08-1998)
-rwsr-xr-x 1 root root 71K Feb 7 15:38 /usr/bin/su
```

THANKS!

Do you have any questions?

fvecchi24@protonmail.com

<https://linkedin.com/in/fvecchi07>



CREDITS: This presentation template was created by Slidesgo, including icons by Flaticon, and infographics & images by Freepik.