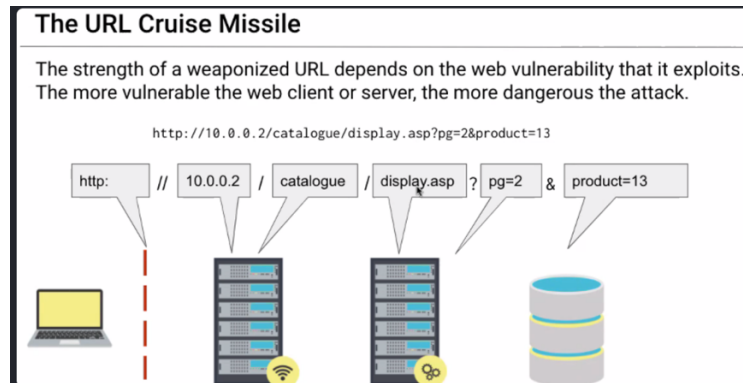


# Week 15 Homework: Web Vulnerabilities and Hardening

## Part 1: Q&A

### The URL Cruise Missile

The URL is the gateway to the web, providing the user with unrestricted access to all available online resources. In the wrong hands can be used as a weapon to launch attack.



Use the graphic below to answer the following questions:

Protocol Host Name Path Parameters  
`http:// www.buyitnow.tv /add.asp ?item=price#1999`

1. **Which part of the URL can be manipulated by an attacker to exploit a vulnerable back-end database system?**

Answer: Parameters

2. **Which part of the URL can be manipulated by an attacker to cause a vulnerable web server to dump the /etc/passwd file? Also, name the attack used to exploit this vulnerability.**

Answer: Path; the attack used to exploit the vulnerability is called Directory Traversal

3. **Name three threat agents that can pose a risk to your organization.**

Answer: Individuals, organizations or governments

4. **What kinds of sources can act as an attack vector for injection attacks?**

Answer: SQL Databases and HTML Code or anything similar that has those capabilities.

5. **Injection attacks exploit which part of the CIA triad?**

Answer: Confidentiality

6. **Which two mitigation methods can be used to thwart injection attacks?**

Answer:

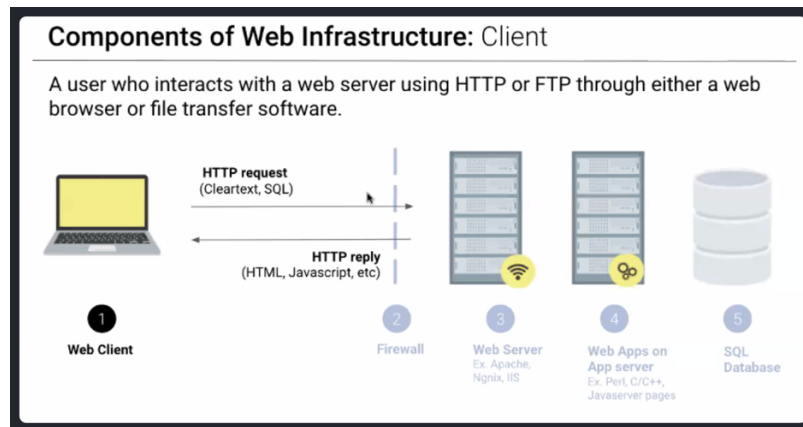
1. Never trust input
2. Implement filtering and monitoring tools
3. Craft error messages carefully
4. Patch and harden databases
5. Limit database privileges

<https://www.darkreading.com/risk/five-ways-to-stop-mass-sql-injection-attacks/d/d-id/1134277>

## Web Server Infrastructure

Web application infrastructure includes sub-components and external applications that provide efficiency, scalability, reliability, robustness, and most critically, security.

- The same advancements made in web applications that provide users these conveniences are the same components that criminal hackers use to exploit them. Prudent security administrators need to be aware of how to harden such systems.



Use the graphic below to answer the following questions:

Stage 1   Stage 2   Stage 3   Stage 4   Stage 5  
Client   Firewall   Web Server   Web Application   Database

- What stage is the most inner part of the web architecture where data such as, customer names, addresses, account numbers, and credit card info, is stored?**  
Answer: Stage 5-Database
- Which stage includes online forms, word processors, shopping carts, video and photo editing, spreadsheets, file scanning, file conversion, and email programs such as Gmail, Yahoo and AOL.**  
Answer: Stage 4-Web Application
- What stage is the component that stores files (e.g. HTML documents, images, CSS stylesheets, and JavaScript files) that's connected to the Internet and provides support for physical data interactions between other devices connected to the web?**  
Answer: Stage 3-Web Server
- What stage is where the end user interacts with the World Wide Web through the use of a web browser?**  
Answer: Stage 1-Client
- Which stage is designed to prevent unauthorized access to and from protected web server resources?**  
Answer: Stage 2-Firewall

## Server-side Attacks

In today's globally connected cyber community, network and OS level attacks are well defended through the proper deployment of technical security controls such as, firewalls, IDS, Data Loss Prevention, EndPoint and security. However, web servers are accessible from anywhere on the web, making them vulnerable to attack.

1. **What is the process called that cleans and scrubs user input in order to prevent it from exploiting security holes by proactively modifying user input.**

Answer: Input Sanitation

2. **Name the process that tests user and application-supplied input. The process is designed to prevent malformed data from entering a data information system by verifying user input meets a specific set of criteria (i.e. a string that does not contain standalone single quotation marks).**

Answer: Input Validation

3. **Secure SDLC is the process of ensuring security is built into web applications throughout the entire software development life cycle. Name three reasons why organization might fail at producing secure web applications.**

Answer:

1. Lack of Policy enforcement
2. Lack of expertise in risk reduction
3. Failure to create a culture of security

<https://www.veracode.com/sites/default/files/Resources/Whitepapers/top-3-reasons-appsec-programs-fail-veracode.pdf>

4. **How might an attacker exploit the robots.txt file on a web server?**

Answer: The file **robots.txt** is used to give instructions to web robots, such as search engine crawlers, about locations within the web site that robots are allowed, or not allowed, to crawl and index. The presence of the **robots.txt** does not in itself present any kind of security vulnerability. However, it is often used to identify restricted or private areas of a site's contents. The information in the file may therefore help an attacker to map out the site's contents, especially if some of the locations identified are not linked from elsewhere in the site. If the application relies on **robots.txt** to protect access to these areas, and does not enforce proper access control over them, then this presents a serious vulnerability.

<https://hackerone.com/reports/156182>

5. **What steps can an organization take to obscure or obfuscate their contact information on domain registry web sites?**

Answer: Privacy and/or Proxy service or private domain registration

<https://webmasters.stackexchange.com/questions/704/how-can-i-hide-whois-information>

<https://www.icann.org/resources/pages/faqs-f0-2012-02-25-en>

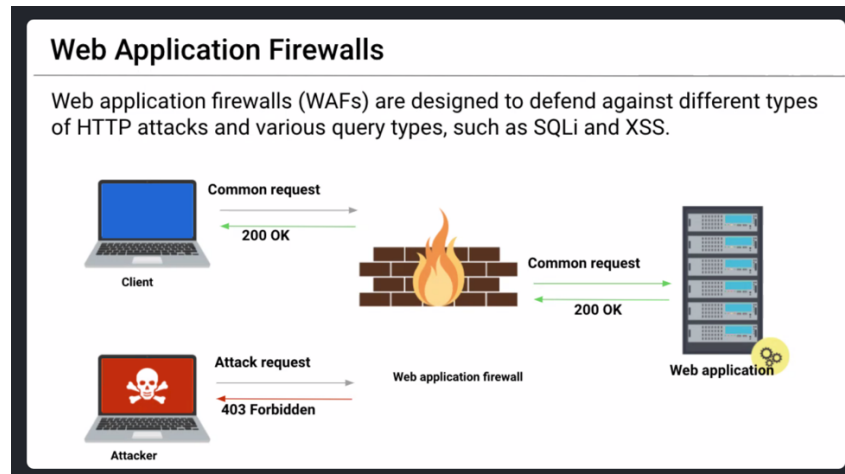
6. **True or False: As a network defender, Client-Side validation is preferred over Server-Side validation because it's easier to defend against attacks.**

- **Explain why you chose the answer that you did.**

Answer: False; Client-side validation is preferred over Server-side validation as it is easier to defend against attacks.

WAFs are designed to defend against different types of HTTP attacks and various query types such as SQLi and XSS.

WAFs are typically present on web sites that use strict transport security mechanisms such as online banking or e-commerce websites.



**1. Which layer of the OSI model do WAFs operate at?**

Answer: Layer 7-Application Layer

**2. A WAF helps protect web applications by filtering and monitoring what?**

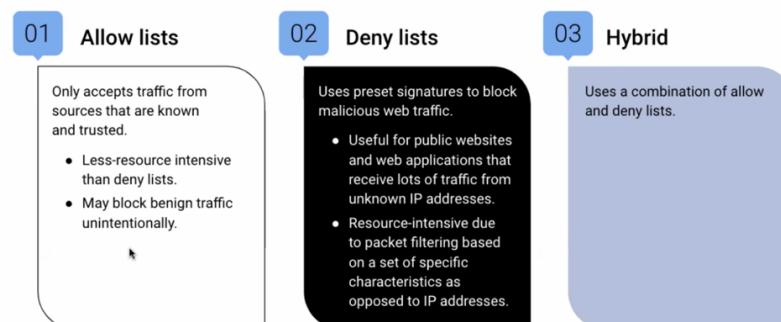
Answer: Incoming HTTP traffic

**3. True or False: A WAF based on the negative security model (Blacklisting) protects against known attacks, and a WAF based on the positive security model (Whitelisting) allows pre-approved traffic to pass.**

Answer: False; Client-side validation can be manipulated and bypassed more easily. You should use a hybrid system for filtering traffic for a WAF.

### WAF Analysis and Filtering

WAFs analyze and filter traffic based on three different techniques:



## Authentication and Access Controls

Security enhancements designed to require users to present two or more pieces of evidence or credentials when logging into an account is called multi-factor authentication.

- Legislation and regulations such as The Payment Card Industry (PCI) Data Security Standard requires the use of MFAs for all network access to a Card Data Environment (CDE).

- Security administrators should have a comprehensive understanding of the basic underlying principles of how MFA works.

1. Define all four factors of multifactor authentication and give examples of each:



- **Factor 1-Standard (Something you know)**

This is the most basic type and most used of the authentications. Passwords can be used in the string of letters, numbers or special characters. To protect yourself you need to create strong passwords that include a combination of all possible options. This factor is best if you combined with other methods. By using multi-factor authentication, you add multiple checkpoints so if there's a possible breach, it's harder for cybercriminals to access information.

*For example: Password, pins, answers to personal security questions*

- **Factor 2-Physical (Something you have)**

This method utilizes a physical asset or information explicitly given or sent to the individual. This next-level identifier is an effective way to prove identity as long.

*For example: Key card or fob, mobile or other device, badge*

- **Factor 3-Biometrics (Something you are)**

Answer: Biometrics authentication is a security process that relies on the unique biological characteristics of an individual. They can be easily compared to authorized features saved in a database. Biometric authentication can control physical access when installed on gates, doors, phones, tablet and computers.

*For example: Facial recognition, fingerprint scanners, voice identification, eye scanners*

- **Factor 4-Adaptive Authentication (Where you are)**

Is the newest and most sophisticated form of authentication, incorporating, location, time or behavior. This factor involves using AI and GPS to pinpoint a user's location or predicted activities and calculate a risk level.

*For example: GPS, MAC Address of the login point, types of network(s) utilizing, using identified or unidentified device(s)*

<https://www.sugarshot.io/what-is-multi-factor-authentication/>

<https://www.idrnd.ai/5-authentication-methods-that-can-prevent-the-next-breach/>

2. True or False: A password and pin is an example of 2-factor authentication.

Answer: False; you must utilize 2 separate factors of authentication for it to be 2-factor authentication.

**3. True or False: A password and google authenticator app is an example of 2-factor authentication.**

Answer: True

**4. What is a constrained user interface?**

Answer: Constrained user interface describes an authentication method that uses only a single type of authentication credentials. The term view-based access control (VBAC) refers to limiting access to databased views as opposed to allowing users to access data in a database tables directly. So, in retrospect it is an interface that is restricted based on privileges.

## Part 2: The Challenge

In this activity, you will assume the role of a pen tester hired by a bank to test the security of the bank's authentication scheme, sensitive financial data, and website interface.

### Lab Environment

We'll use the **Web Vulns** lab environment. To access it:

- Log in to the Azure Classroom Labs dashboard.
- Find the card with the title **Web Vulns** or **Web Vulnerability and Hardening**.
- Click the monitor icon in the bottom-right.
- Select **Connect with RDP**.
- Use Credentials (azadmin:p4ssw0rd\*)
- The lab should already be started, so you should be able to connect immediately.
- Refer to the [lab setup instructions](#) for details on setting up the RDP connection.

Once the lab environment is running, open the HyperV manager and make sure that the OWASPBWA and Kali box is running.

- Then, login to the Kali VM and navigate to the IP address of the OWASPBWA machine.
- Click the option for 'WebGoat' and start the WebGoat app.
- Use the credentials: guest:guest

On the bottom of the left side of the screen, click on Challenge and then choose The Challenge.

**Note:** A common issue with this lab is the Challenge activity failing to start successfully. Hit the Restart the Lesson button in the top right if you get an error starting the activity.

## The Challenge Instructions

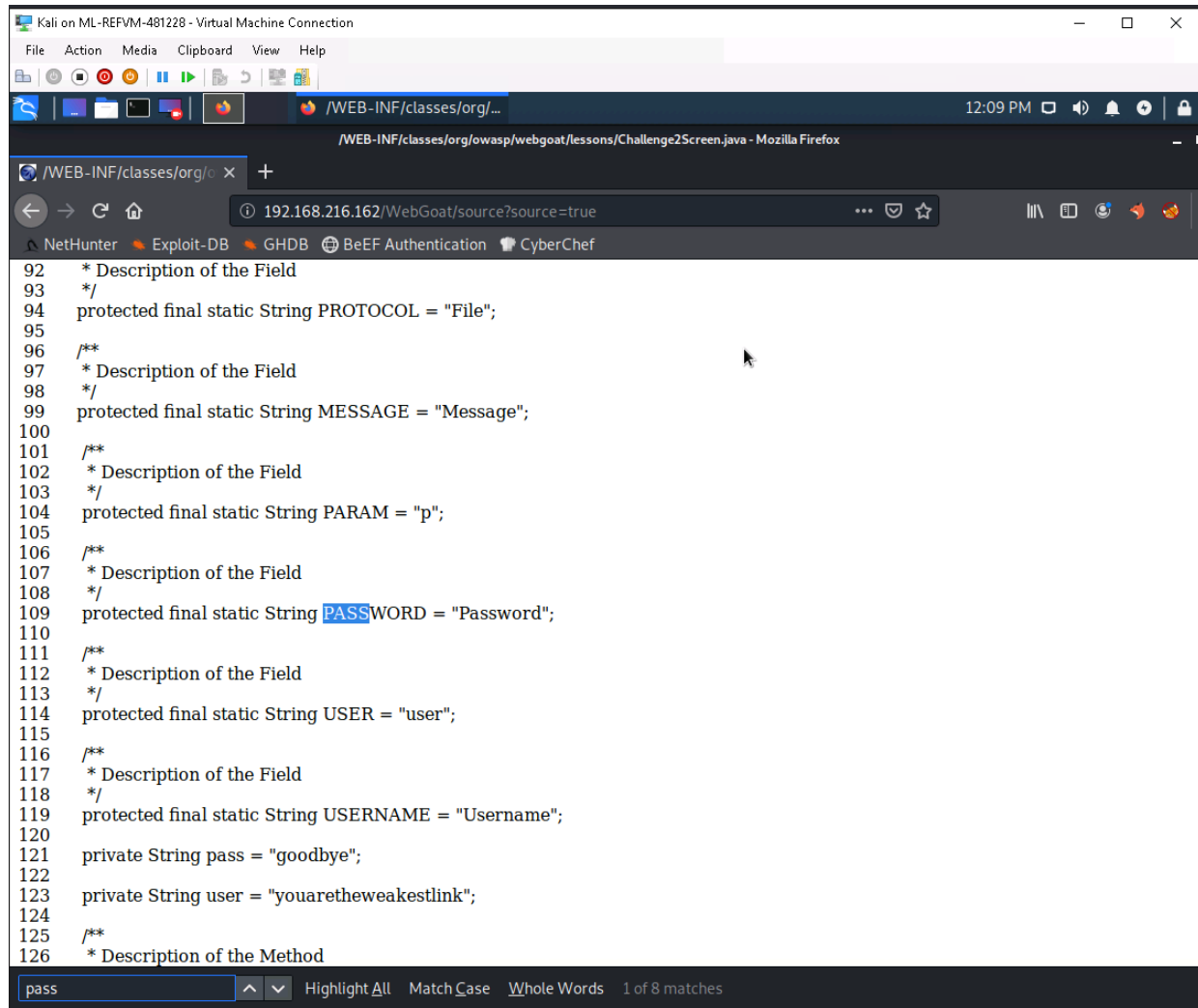
### Challenge #1

Your first mission is to break the authentication scheme. There are a number of ways to accomplish this task.

- **Hint #1:** Sometimes, form fields are shy!
- **Hint #2:** Find the hidden JavaScript.

- **Hint #3:** You can append source?source=true to the URL to read the source code.

Please include a screenshot here of the hidden JavaScript:



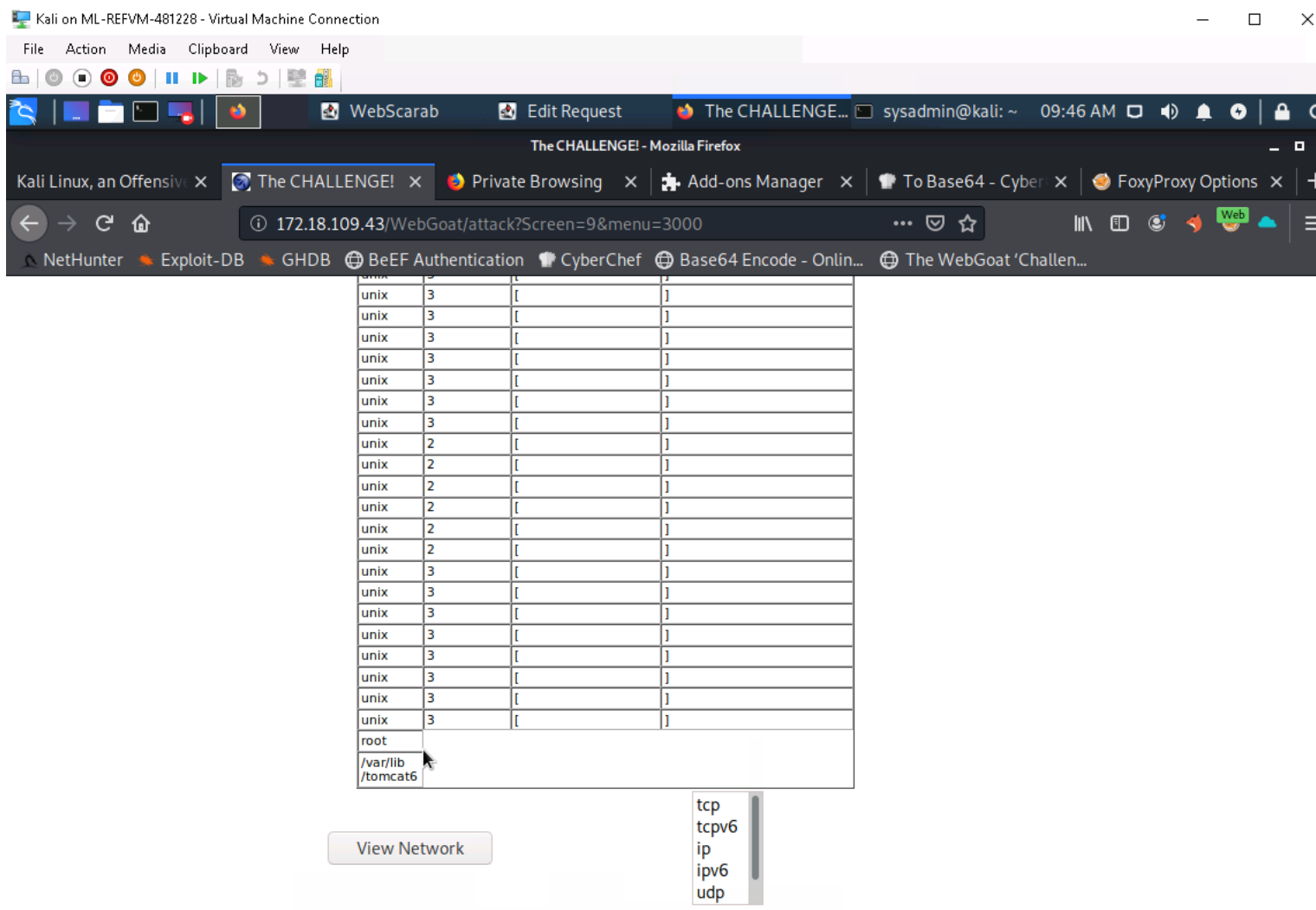
```
92  * Description of the Field
93  */
94  protected final static String PROTOCOL = "File";
95
96  /**
97  * Description of the Field
98  */
99  protected final static String MESSAGE = "Message";
100
101  /**
102  * Description of the Field
103  */
104  protected final static String PARAM = "p";
105
106  /**
107  * Description of the Field
108  */
109  protected final static String PASSWORD = "Password";
110
111  /**
112  * Description of the Field
113  */
114  protected final static String USER = "user";
115
116  /**
117  * Description of the Field
118  */
119  protected final static String USERNAME = "Username";
120
121  private String pass = "goodbye";
122
123  private String user = "youaretheweakestlink";
124
125  /**
126  * Description of the Method
```

## Challenge #2

Next, steal all of the credit card numbers from the database.

- **Hint #1:** Sometimes cookies wear different clothes to change their appearances.
- **Hint #2:** Break your way into the conversation and inject your own ideas.

Please include a screenshot here of all the credit card numbers from the database.



After completing the second challenge, you will be provided with an option to continue to the next challenge.

### Challenge #3

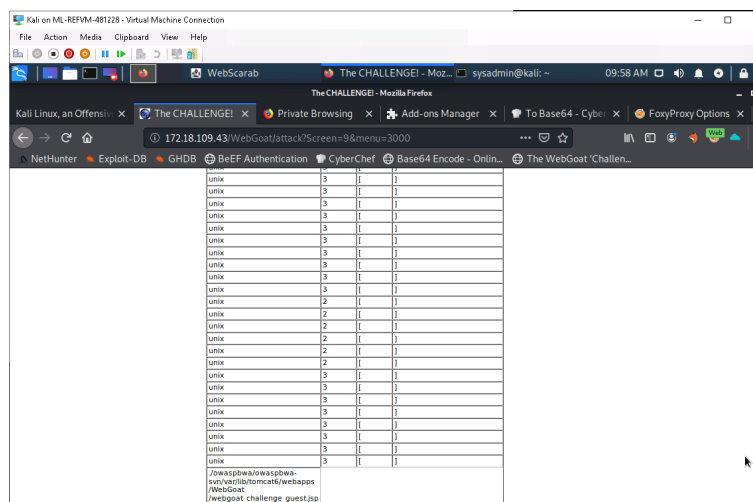
Your final act is to deface the website using command injection. Follow the walkthrough below to help you get started.

- After completing the second challenge, you will be provided with an option to continue to the next challenge.
- There should be two webpages at the bottom of the window. The one on top is the original, and the one on the bottom is the defaced webpage.
- Start Foxy Proxy (WebScarab) to send all GET/POST requests from Firefox to the WebScarab proxy intercept.
- Click **TCP** and then the **View Network** button and send the request to WebScarab.
- The WebScarab window will open.
  - In the **URL Encoded** tab, find the **File** and **Value** form fields.
  - This is where you will perform your command injection.
- Next, perform a test and see if this shell is vulnerable to command injection.
  - Type the following command into the Value field: `tcp && whoami && pwd.`
  - **Note:** Windows users can type `tcp && dir.` `dir` will return the directory as proof of vulnerability.
  - Click **Accept Changes**.



- 
- The screenshot shows a Kali Linux virtual machine interface. At the top, there's a menu bar with 'File', 'Action', 'Media', 'Clipboard', 'View', and 'Help'. Below it is a taskbar with various application icons. The main window is a Mozilla Firefox browser titled 'The CHALLENGE!'. The address bar shows the URL '172.18.109.43/WebGoat/attack?Screen=9&menu=3000'. The browser tabs include 'Kali Linux, an Offensiv...', 'The CHALLENGE!', 'Private Browsing', 'Add-ons Manager', 'To Base64 - Cyber...', and 'FoxyProxy Options'. The browser's address bar also shows several search engines and tools like 'NetHunter', 'Exploit-DB', 'GHDB', 'BeEF Authentication', 'CyberChef', and 'Base64 Encode - Onlin...'. The main content area displays a table with columns for 'ip', 'port', 'protocol', and 'type'. The table contains 20 rows of data, mostly for 'unix' type connections on ports 3 and 2. Below the table, there's a 'root' button and a '/var/lib/tomcat6' button. A 'View Network' button is located at the bottom left. On the bottom right, there's a network configuration menu with options: 'tcp', 'tcpv6', 'ip', 'ipv6', and 'udp'.
- | ip               | port | protocol | type |
|------------------|------|----------|------|
| unix             | 3    | [ ]      |      |
| unix             | 3    | [ ]      |      |
| unix             | 3    | [ ]      |      |
| unix             | 3    | [ ]      |      |
| unix             | 3    | [ ]      |      |
| unix             | 3    | [ ]      |      |
| unix             | 3    | [ ]      |      |
| unix             | 3    | [ ]      |      |
| unix             | 2    | [ ]      |      |
| unix             | 2    | [ ]      |      |
| unix             | 2    | [ ]      |      |
| unix             | 2    | [ ]      |      |
| unix             | 2    | [ ]      |      |
| unix             | 3    | [ ]      |      |
| unix             | 3    | [ ]      |      |
| unix             | 3    | [ ]      |      |
| unix             | 3    | [ ]      |      |
| unix             | 3    | [ ]      |      |
| unix             | 3    | [ ]      |      |
| unix             | 3    | [ ]      |      |
| unix             | 3    | [ ]      |      |
| unix             | 3    | [ ]      |      |
| root             |      |          |      |
| /var/lib/tomcat6 |      |          |      |
- View Network
- tcp  
tcpv6  
ip  
ipv6  
udp

- Scroll to the bottom of the **Current Network Status** window and observe the results for both of the whoami and pwd commands.
  - The results show that we are the root user and our current working directory is /var/lib/tomcat6.
  - This verifies the vulnerability, so proceed to the next step.
- Next, we'll locate the webgoat\_challenge\_guest.jsp file.
  - Type the following command: tcp && cd / && find . -iname webgoat\_challenge\_guest.jsp.
    - **Note:** Windows users will need to type: tcp && dir /s 'webgoat\_challenge\_guest.jsp'
  - The absolute path is: ./owaspbwa/owaspbwa-svn/var/lib/tomcat6/webapps/WebGoat/webgoat\_challenge\_guest.jsp.
  - Remember, our present working directory is /var/lib/tomcat6. Therefore, the relative path is webapps/WebGoat/webgoat\_challenge\_guest.jsp.



## Now it's your turn

- Now that we know where the webpage is, your task will be to deface the website. Keep in mind the following:
  - Use **WebScarab** to perform command injection.
  - When performing command injection, you will need to select a field that WebScarab can return commands to. These fields are typically located in a drop down.
  - You will also need to locate and edit the the webpage's source code: `webgoat_challenge_guest.jsp`
  - Your final command will:
    - Change to the location of the `webgoat_challenge_guest.jsp` file.
    - **and** echo You've been hacked by... followed by your name, to the `webgoat_challenge_guest.jsp` file.

