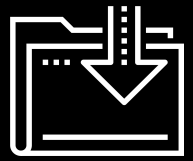
Web Application Tool Time

Cybersecurity

Web Vulnerabilities and Hardening, Day 3



Class Objectives

By the end of today's class, you will be able to:

Identify ways in which web application security tools can assist with testing security vulnerabilities.

Configure Burp Suite and FoxyProxy to capture and analyze an HTTP request.

Identify session management vulnerabilities using the Burp Suite Repeater function.

Conduct a brute force attack against a web application login page with the Burp Intruder function.

Attacking Web Applications Using Security Tools


Recap

Last Class

Web applications use **back-end components** to apply business logic to the application. Back-end components can provide file structure, content management, and access control and include back-end languages such as **PHP** and **Java**.

Back-end component vulnerabilities exist within the back-end components of web application servers.

Last Class

Directory Traversal

A web app back-end component vulnerability in which an attacker accesses files and directories from a web application outside a user's authorized permissions.

Local File Inclusion (LFI)

Another web app back-end component vulnerability in which an attacker tricks the application into running unintended back-end code or scripts that are LOCAL to the application's filesystem.

Remote File Inclusion (RFI)

A back-end component web app vulnerability in which an attacker tricks the application into running unintended back-end code or scripts—similar to LFI, except that the scripts are REMOTE to the application's filesystem.

Directory traversal, LFI, and RFI all fall under the OWASP risk broken access control.

OWASP explains broken access control as follows:

Restrictions on what authenticated users are allowed to do are often not properly enforced.

Attackers can exploit these flaws to access unauthorized functionality and/or data, such as access other users' accounts, view sensitive files, modify other users' data, change access rights, etc.

Today's Class

Web application security tools can assist security professionals by automating testing processes.

When we first executed SQL injection attacks, we were successful after our first payload attempt of:

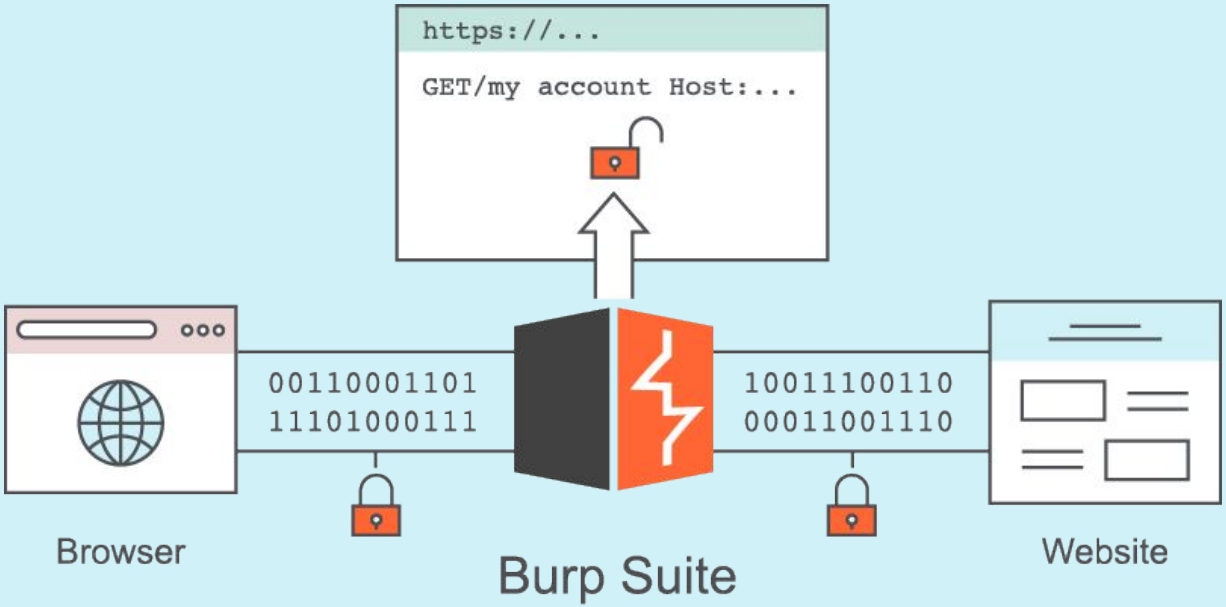
```
jsmith' OR '1' = '1
```

Realistically, we may need to test many different payloads, such as:

```
jsmith" or true--
jsmith OR "1" = "1
jsmith -- OR '1' = '1
jsmith') or true--
```

Today's Class

We will use **Burp Suite**, a popular web application security tool, to automate processes, capture, display, and modify web app requests and responses, and use other built-in features to test for vulnerabilities.



(portswigger.net)

While demonstrating Burp Suite, we will explore vulnerabilities that fall under the **OWASP risk of Identification and Authentication Failures**.

A01	Broken Access Control	A06	Vulnerable and Outdated Components
A02	Cryptographic Failures	A07	**Identification and Authentication Failures**
A03	Injection	A08	Software and Data Integrity Failures
A04	Insecure Design	A09	Security Logging and Monitoring Failures
A05	Security Misconfiguration	A10	Server-Side Request Forgery (SSRF)

Identification and authentication failures are risks that permit an attacker to either access or bypass the authentication methods that are used by a web application.

Today's Class

The class will proceed as follows:

01

We will begin by introducing **Burp Suite** and **FoxyProxy** and learning how to configure them to capture and analyze HTTP requests and responses.

02

Then, we will learn how to use the **Burp Repeater** feature to determine session hijacking vulnerabilities.

03

Lastly, we will show how to use the **Burp Intruder** feature to conduct an automated brute force attack.

⚠️ Important ⚠️

The techniques we will learn throughout this unit can be used to cause serious damage to an organization’s systems.

This is ILLEGAL when done without permission. All of the labs that we provide are safe locations to test the methods and tools taught during the week.

NEVER apply any of these methods to any web applications that you do not own or do not have clear, written permission to be interacting with.

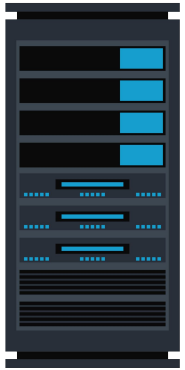
Introduction to Web Proxies and Burp Suite

How Web Apps Interact with Back-End Servers

01

A user's browser (the client) requests an image of a car to be displayed with an HTTP request.

HTTP request for car image



02

The web server (the server) responds with that car image by sending an HTTP response.

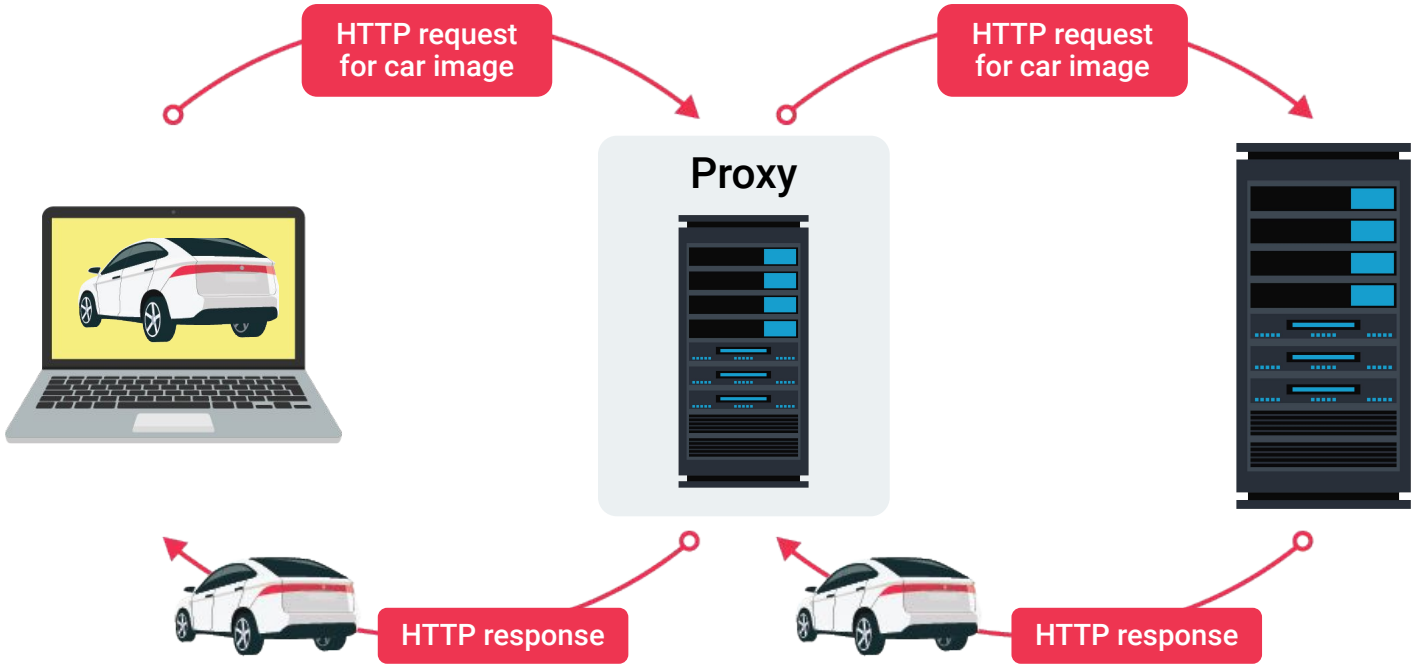
03

The browser receives the HTTP response and renders the image of the car to the user.

HTTP response

Web Proxy

A **web proxy** is an intermediary between the client and the server.
Internet traffic flows through the proxy on the way to its intended destination.



Web Proxy

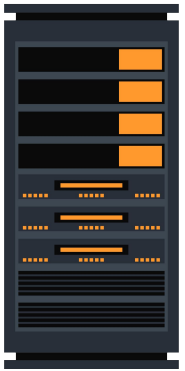
Web proxies can be used by:

Organizations

To monitor and block harmful web traffic, as some web proxies can be configured to block specific websites.

Individuals

To provide themselves anonymity when using the internet, as some web proxies can change the source IP address.



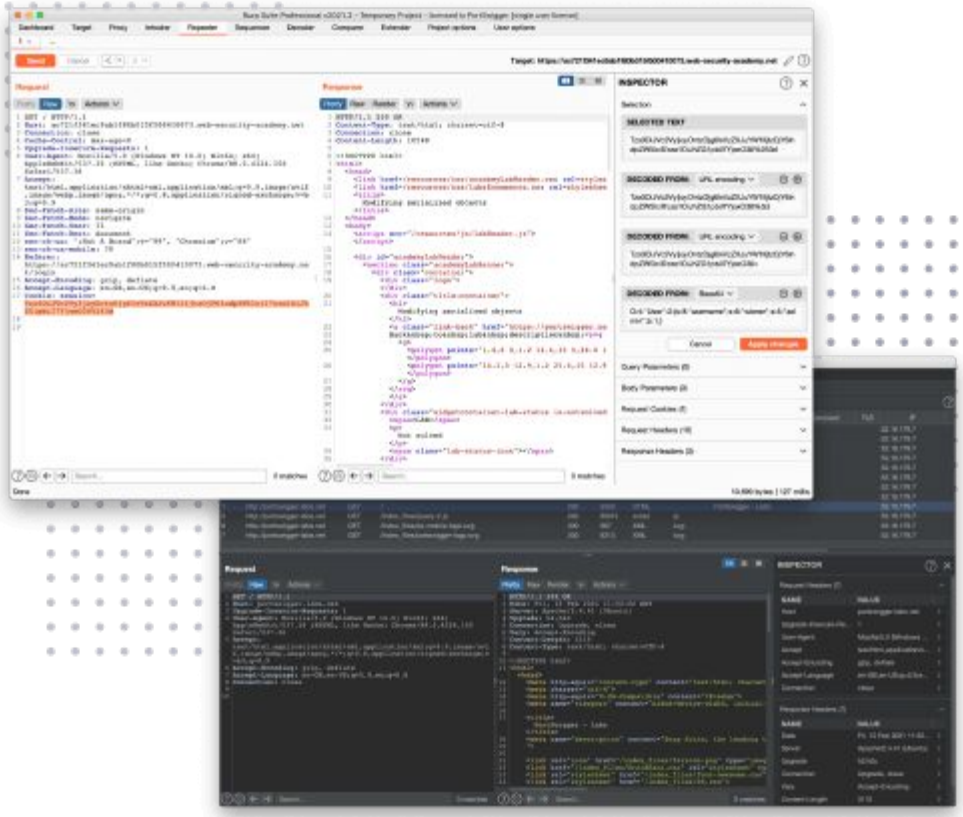
Burp Suite

Burp Suite is a web application security tool that lies between your browser and your target application.

Burp Suite

Community Edition

- Burp Suite intercepts raw HTTP traffic either from the browser or the server, thus functioning as a web proxy.
- Burp has many additional features and capabilities that allow a security professional to analyze, modify, and automate the HTTP traffic before passing it along to its final destination.



Burp Suite Demo

Burp Suite is a web application security tool that lies between your browser and your target application. In the following demo, we will:

01 Start and access Burp Suite

02 Navigate Burp Suite.

03 Configure the proxy on Burp Suite.

04 Configure the proxy on your browser.

05 Enable FoxyProxy  to send traffic to Burp Suite.

06 Review the captured traffic on Burp Suite.

Instructor Demonstration

Burp Suite

Questions?

Activity: Configuring Burp Suite

In this activity, you'll configure Burp Suite proxy settings in order to prepare for a future session hijacking attempt.

Suggested Time:

20 Minutes

Time’s Up! Let’s Review.

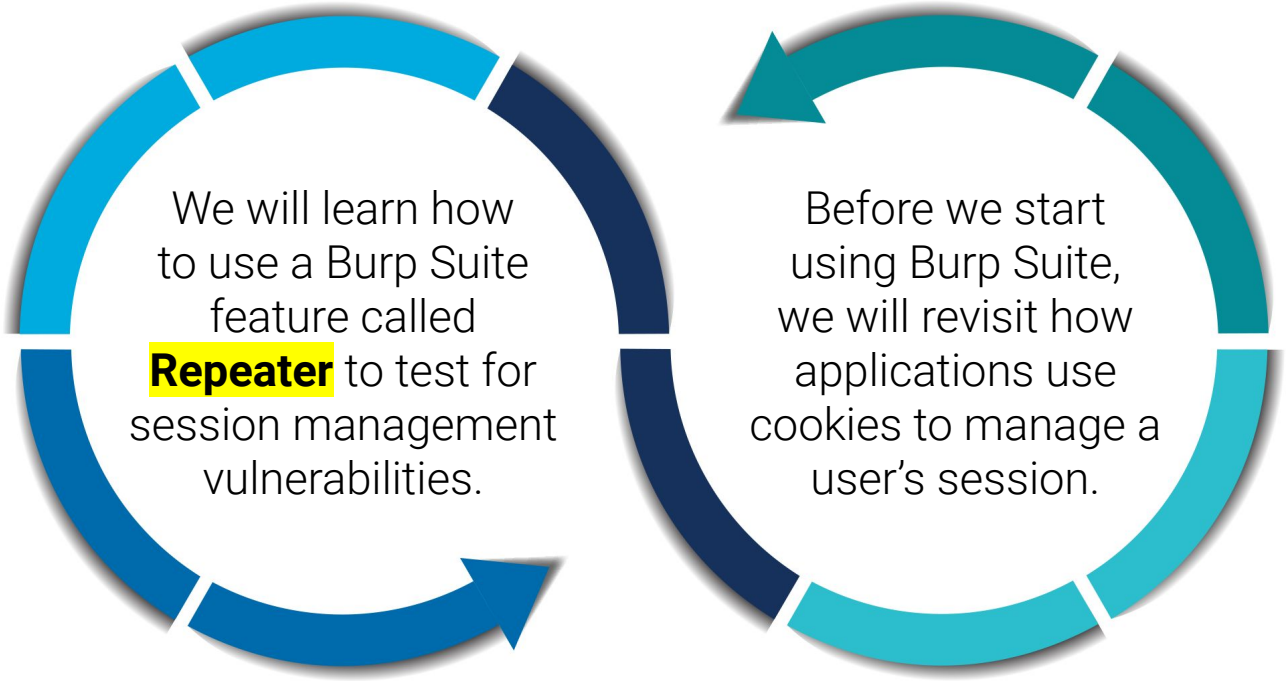
Questions?

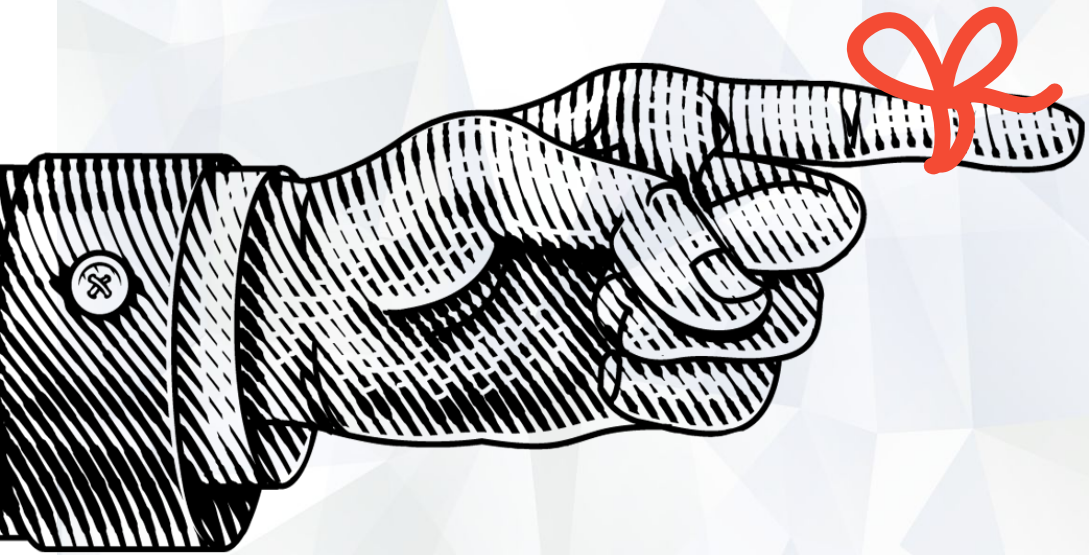

Break

Session Management Vulnerabilities

Session Management Vulnerabilities

Next…





Remember,

HTTP resources are inherently stateless, meaning that whenever your browser requests a webpage, there is no way for that webpage to distinguish you from anyone else.

Session Management and Cookies

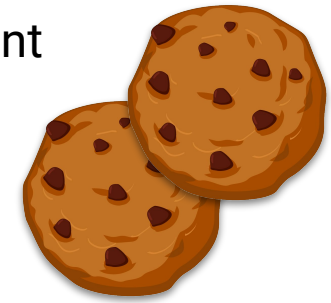
Websites need a way to deliver content that is specific to each user. To do so, they establish sessions by using cookies.

Sessions

Unique server-side sets of user data that are used to customize webpages for the specific user accessing them

Cookies

Small pieces of text data that, when sent by an HTTP server's response header, are saved by the user's HTTP client



Session Hijacking

While using session cookies is intended to keep a state between webpages when a user accesses a web application, a malicious user can obtain another user's unique session cookie and hijack the user's private session.

This in an example of **session hijacking**.

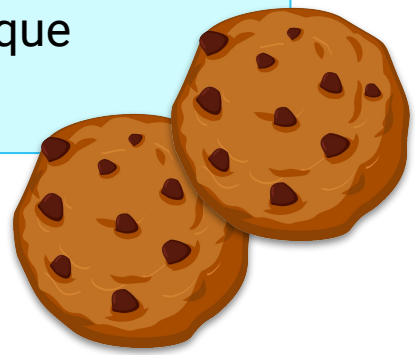
Remember, we conducted a session hijacking attack when we used the Chrome browser extension Cookie-Editor to swap sessions.

Session hijacking falls under the OWASP Top 10 risk of **identification and authentication failures**, because if an attacker has access to a user's session, they can bypass the application's authentication measures.

A01	Broken Access Control
A02	Cryptographic Failures
A03	Injection
A04	Insecure Design
A05	Security Misconfiguration
A06	Vulnerable and Outdated Components
A07	**Identification and Authentication Failures**
A08	Software and Data Integrity Failures
A09	Security Logging and Monitoring Failures
A10	Server-Side Request Forgery (SSRF)

Session Hijacking Methods

Sniffing traffic:	If a malicious user can sniff unencrypted traffic, then they can potentially capture the session cookie and take over a user's session.
Client-side attacks:	A malicious user can deploy a cross-site scripting attack to steal a user's session cookie.
Predictable sessions:	A malicious user can predict what a unique session cookie might be.



Predictable Session Scenario

Henry, a malicious user, is using a stock-trading website to buy and sell stocks and mutual funds.

Henry logs in with his own credentials on Monday, February 9, 2021, to the stock-trading site, then checks his session cookie in his browser settings:

`020921MON-1454`

He logs out and immediately logs back in again with his own credentials, then checks his session cookie again:

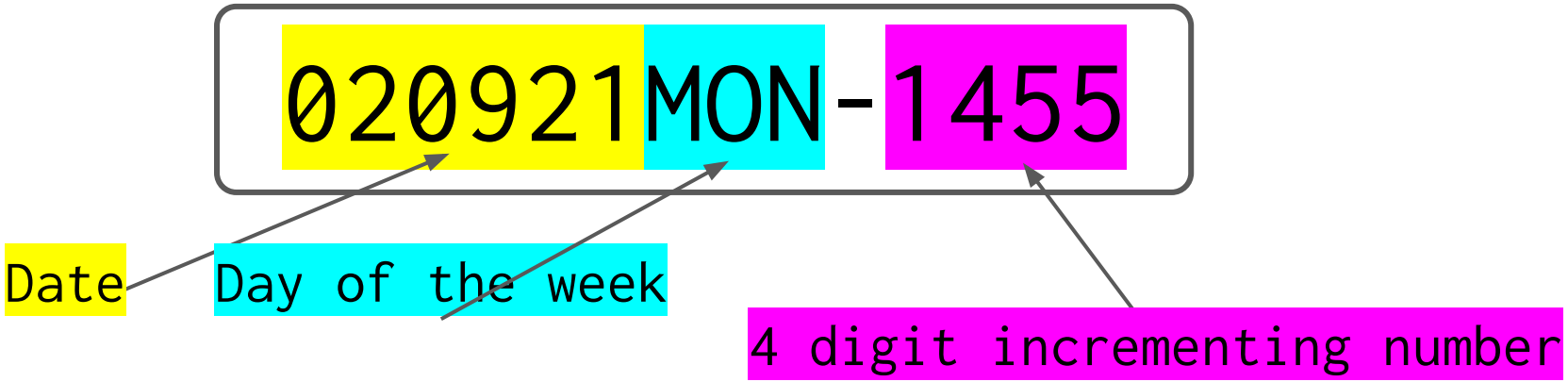
`020921MON-1455`

Henry logs in the following day, Tuesday, February 10, 2021, and his session cookie is now:

`021021TUE-3834`

Predictable Session Scenario

By examining the session cookies that were generated, Henry can determine that the algorithm used by the stock trading site likely comprises the following:



Predictable Session Scenario

By figuring out the algorithm, Henry can try and guess another user's session cookie, then hijack another user's session.

Henry might first try: `021121WED-1111`

If that didn't work, he could try: `021121WED-1112`

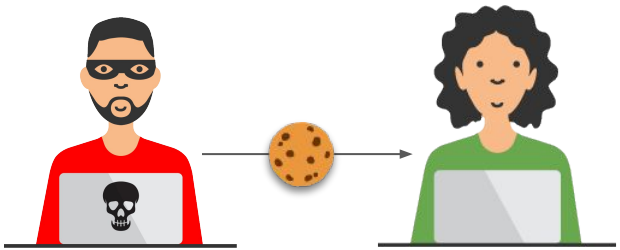
If that didn't work, he could try: `021121WED-1113`

If that didn't work, he could try: `021121WED-1114`

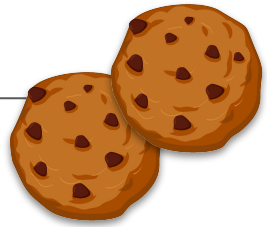
Eventually, after many tries, Henry tries the session cookie:

`021121WED-1118`

Then, he ends up inside the stock-trading account of another user, Julie Jones.



Predictable Session Scenario



In this scenario:

01 The session management vulnerability was that the session cookie was predictable.

02 Henry exploited the vulnerability by conducting a session hijacking attack, using a predicted session cookie to hijack Julie Jones's private session.

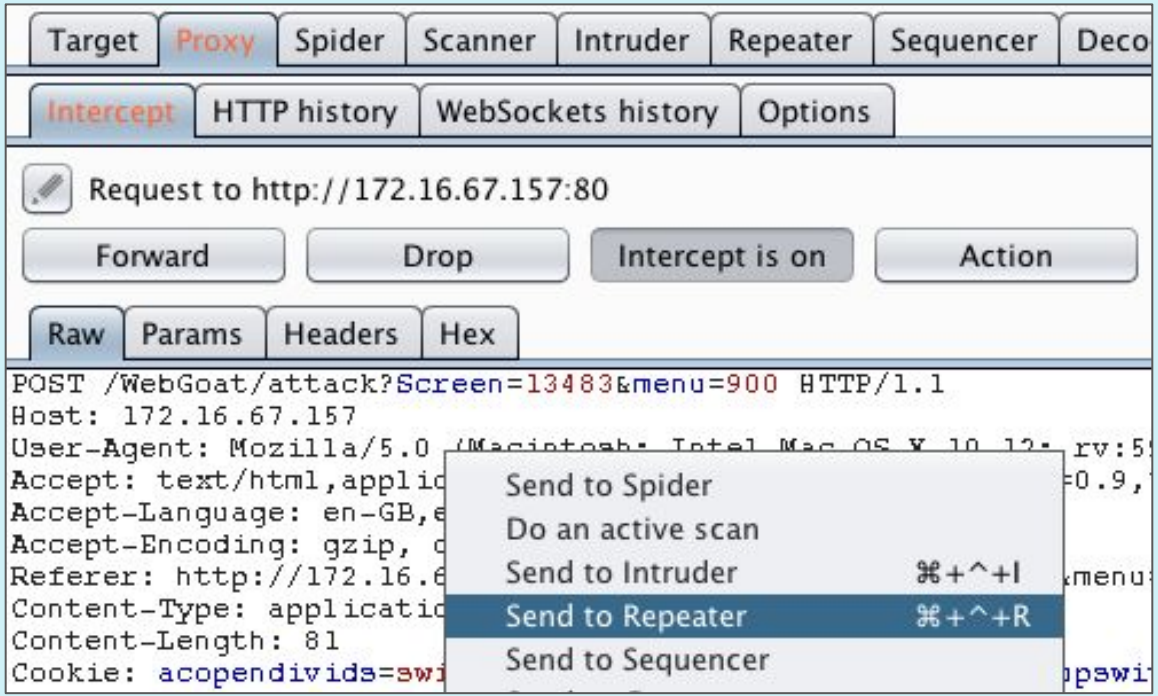
03 Note that Henry had to log out and log back in again to determine the pattern of the session cookies. This manual process can be time-consuming and inefficient.

Questions?

Analyzing Session Management Vulnerabilities with Burp Repeater

Burp Suite Repeater

Burp Suite Repeater can simplify the process detailed in the previous scenario by using an algorithm to generate session cookies.



Burp Repeater Demo

In this demonstration, we will continue to work on the Replicants web application and complete the following steps:

01 Access the session cookie generator and enable proxy settings.

02 Generate and review the session cookie.

03 Move the HTTP request to Burp Repeater.

04 Use Burp Repeater to review the HTTP response.

Instructor Demonstration

Burp Repeater

Real-World Challenges, Impact, and Mitigations

Real-World Challenges

While this demo illustrated how to determine session management vulnerabilities with the Burp Repeater feature, in the real world, we also need to be aware of several issues that we did not cover.

Generation of Session Cookies

- While we clicked a button to generate session cookies, real applications do not have a “generate” button.
- Most web applications generate the session cookie on successful authentication into the web application.
- Security professionals have to determine the HTTP request that generates the session cookie and use that request in Burp Repeater.

Use of Burp Repeater

- While we used Burp Repeater to simply reissue the same HTTP request, Burp Repeater can be used to modify the request one at a time.
- On the HTTP Request panel, an attacker can simply change any data in the request to check how it changes the response.
- Attackers can use this to try attacks against different hosts, resources, and types of HTTP requests.

Impact

The reason that **session hijacking** is considered harmful is due to the impact that it can have.

If a malicious actor can successfully launch a session hijacking against a web application's users, they can potentially:

- View a user's confidential data.
- Use features that are not intended to be accessed by unauthorized users within a secure application.

Mitigation Methods

To mitigate against:

	Sniffing attacks	Client-side attacks	Predictable sessions
The application's web developer can:	**Use an HTTP header to have a Secure cookie attribute.**	**Use an HTTP header to have an HttpOnly cookie attribute.**	**Prevent the generated session cookie from being predicted or brute forced.**
This ensures that:	The client only sends the session cookie encrypted, and it cannot be exposed in the web traffic.	• The client does not allow scripts, such as JavaScript, to access the session cookie. • This can protect against a cross-site scripting attack trying to steal a cookie.	• The cookie is long enough to make it difficult to brute force. • The cookie is random enough to be unpredictable. • The cookie doesn't contain any details about the user.

Session Management Vulnerability Summary

Intended purpose of the original function:	Session cookies are INTENDED to maintain a user's session within a web application.
Vulnerability and method of exploit:	With a session management vulnerability such as a predictable session, an attacker could predict a future session cookie and conduct a session hijacking attack.
Unintended consequence of the exploit:	The unintended consequence is that if an attacker determines a user's session cookie, they can access the user's private session.
Mitigation of the vulnerability:	Mitigations can vary depending on the session management vulnerability you are protecting against. Mitigations can include using protective HTTP headers and generating session cookies that are difficult to predict.
Potential impact of the exploit:	The impact could include viewing a user's confidential data within the application or conducting unauthorized activities within the application.

Questions?

Activity: Analyzing Session Management Vulnerabilities with Burp Repeater

In this activity, you will use Burp Suite to analyze session IDs that are generated from your application and determine whether they are randomly generated and thus at risk of session hijacking.

Suggested Time:

20 Minutes

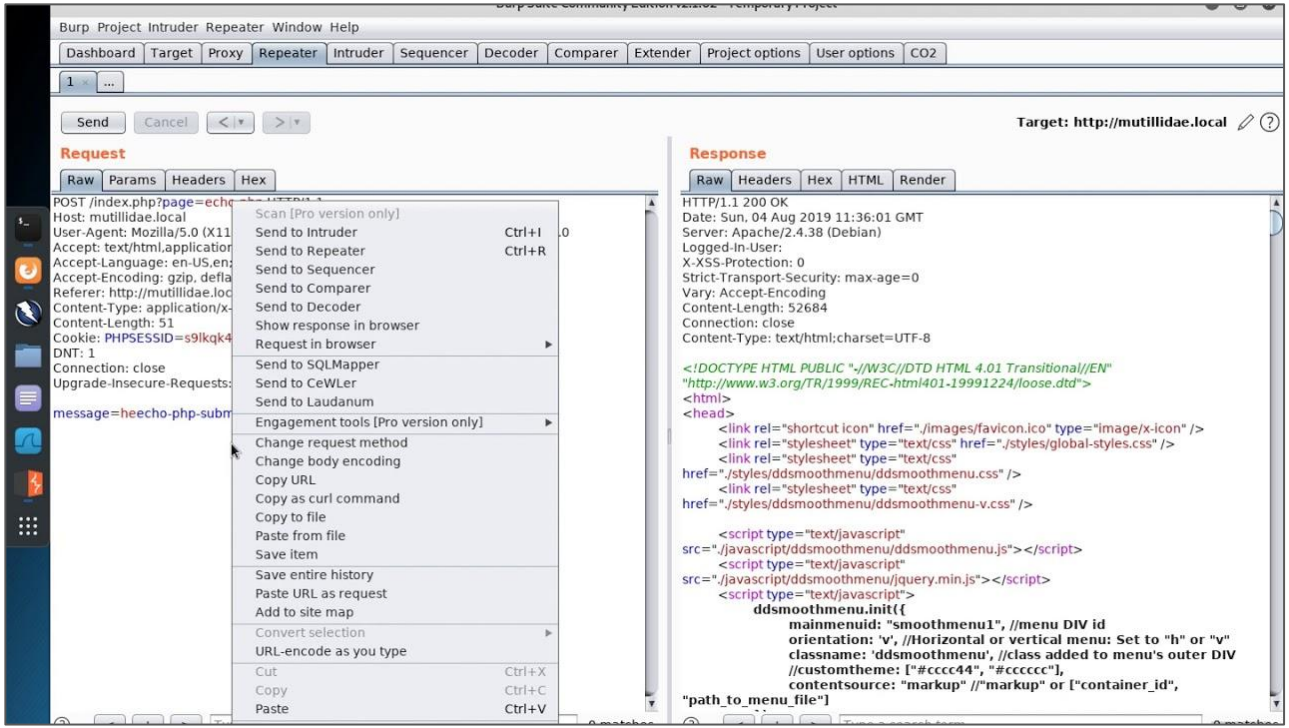
Time's Up! Let's Review.

Questions?

Conducting Brute Force Attacks with Burp Intruder

Conducting Brute Force Attacks with Burp Intruder

Burp Repeater is limited to reissuing *individual* HTTP messages.



Sometimes it is useful to issue multiple HTTP messages and automate MULTIPLE HTTP messages with a single request.

Scenario

An attacker is trying to attack a web application form field with a SQL injection attack.

The attacker attempts a SQL injection attack, with the common payload of:

```
' OR '1' = '1
```

Unfortunately for the attacker, the common payload

```
' OR '1' = '1
```

doesn't work.

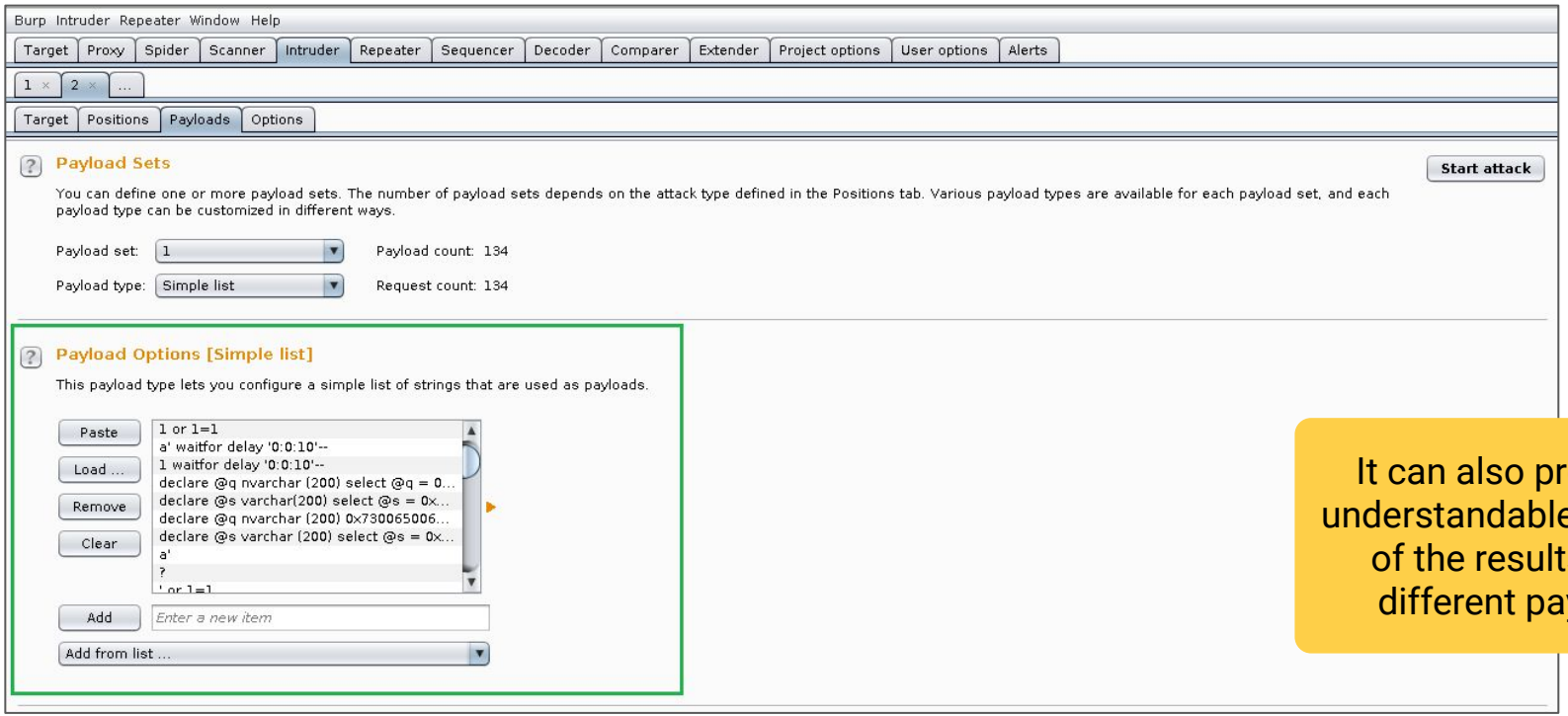
The attacker now wants to try many different payloads, such as:

```
* ' or "
-- or #
' OR '1
' OR 1 -- -
" OR "" = "
" OR 1 = 1 -- -
' OR '' = '
```

Rather than trying this list one at a time and checking the results individually, it would be more efficient for the attacker to use a tool to try all of these payloads with a single request.

Burp Intruder

Burp Intruder can automate issuing multiple HTTP messages.



It can also provide an understandable summary of the results of the different payloads.

We will demonstrate Burp Intruder by conducting a brute force attack against a web application's login page.

A **brute force attack** is defined as systematically attempting many passwords, or username/password combinations, with the hope of determining the correct result to obtain access in an unauthorized area.

Brute Force Attacks

Other variations of brute force attacks include:

Credential Stuffing

Using lists of usernames to conduct brute force attacks that were obtained from breaches of other websites.

Password Spraying

Using a single weak password, such as “123456”, against a large list of usernames.

Brute force attacks also fall under the **OWASP Top 10 risk of broken authentication**, as the attacker aims to bypass an application’s method to securely authenticate a user.

Brute Force Attacks

An attacker can use a brute force attack to continuously attempt username and password combinations.

Burp Intruder Demonstration

In the following demonstration, we will:

- Access the Replicants administrator login page and enable proxy settings.
- Generate and display a login request.
- Move the HTTP request to Burp Intruder.
- Identify the payload positions.
- Update the intruder payloads.
- Launch the brute force attack, and analyze the results.

Instructor Demonstration

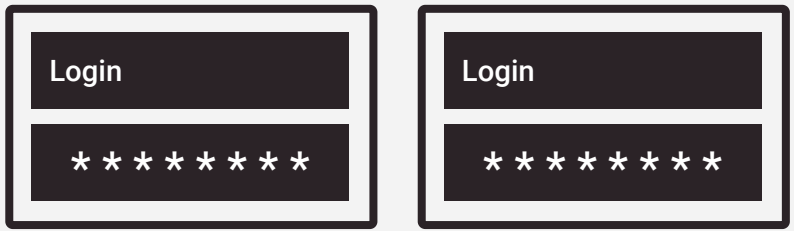
Burp Intruder

Real-World Challenges, Impact, and Mitigations

Real-World Challenges

Uses of Burp Intruder:

While we used Burp Intruder to test a small list of two users and two passwords, most brute force tests would include much larger lists.



We used Burp Intruder to conduct a brute force attack, but Burp Intruder can be used to conduct a variety of other web application attacks that test different payloads and payload combinations.

Impact

If a malicious actor can successfully launch a brute force attack against a web application's users and gain access to users' accounts, they can do the following:

- View a user's confidential data.
- Use features that are not intended to be accessed by unauthorized users within a secure application.

Mitigation Methods

Developers can use several mitigation methods to protect against a brute force attack:

	01	02	03
	Require complex usernames and passwords.	**Lock out accounts after a number of failed attempts.**	**Use multi-factor authentication (MFA).**
For example:	Require the user to include special characters, upper- and lowercase, and numbers in the username and password.	After three failed login attempts, the user's account gets locked.	Require users to have a password and a secondary form of authentication, like a pin generated by an external token.

Brute Force Attack Summary

Intended purpose of the original function:	Web applications use username and password input fields to authenticate a user before they can access a secure part of a web application.
Vulnerability and method of exploit:	Without proper protections on the login page, an attacker can apply a brute force attack to systematically test many user/password combinations to attempt to gain access to unauthorized accounts.
Unintended consequence of the exploit:	The unintended consequence is that if an attacker can determine a correct username/password combination, they can access an account that they do not have permission to access.
Mitigation of the vulnerability:	Mitigations can include requiring complex usernames and passwords, using MFA, and enabling a lockout after a certain amount of failed login attempts.
Potential impact of the exploit:	The impact could include viewing a user's confidential data within the application or conducting unauthorized activities within the application.

Activity: Brute Force Attacks with Burp Intruder

In this activity, you will use Burp Suite to determine if weak user passwords are vulnerable to brute force attacks.

Suggested Time:

20 Minutes

Time's Up! Let's Review.

Questions?

Instructor Demonstration

Set Up Next Week's Lab Environment

Next Week's Lab Environment

We will return to Azure Lab Services and use a new Pen testing lab environment.

- Switch to your local computer environment.
- You will be provided with a registration link for the Pen testing environment.
- Once you click the link, the Pen testing environment card will be added to your Azure dashboard.
- Make sure you can set up and access this environment. Reach out if you need help with any troubleshooting issues prior to the next class.


The End