RSAConference™2024
San Francisco | May 6 – 9 | Moscone Center

THE ART OF POSSIBLE

SESSION ID: LAB3-T09

# Web Application Exploration

*"Web Application Hacking 101' No Tools"*

#RSAC

**Joseph Mlodzianowski**
Chief Exploiter
Darknets.org

**Lee McWhorter**
Director
McWhorter Technology

**Sandra Stibbards**
Owner, Private Investigator
Camelot investigations, OSINT

# Welcome

- Introductions
- Setting up your environment
- Browser Plugins
- What is a Web Application
- Web Application Lab Frameworks
- Accessing online system
- Performing Exercises
- What's Next?

# Who we are:

- Joseph Mlodzianowski
  - Cisco CCIE, CISSP, ITILv4, Author, Trainer and 30-year veteran of the Networking/Infosec/Cyber field.
  - *"OPEN" Seeking a CISO/CTO Role – Joseph.Mlodzianowski@gmail.com Twitter: @cedoxx – Linkedin: https://linkedin.com/in/mlodzianowski phone 210.885.4188*
  - *MORE About Joseph: https://CyberLearningPath.org for all my courses at Pearson/O'Reilly & https://darknets.org for dark web investigations*

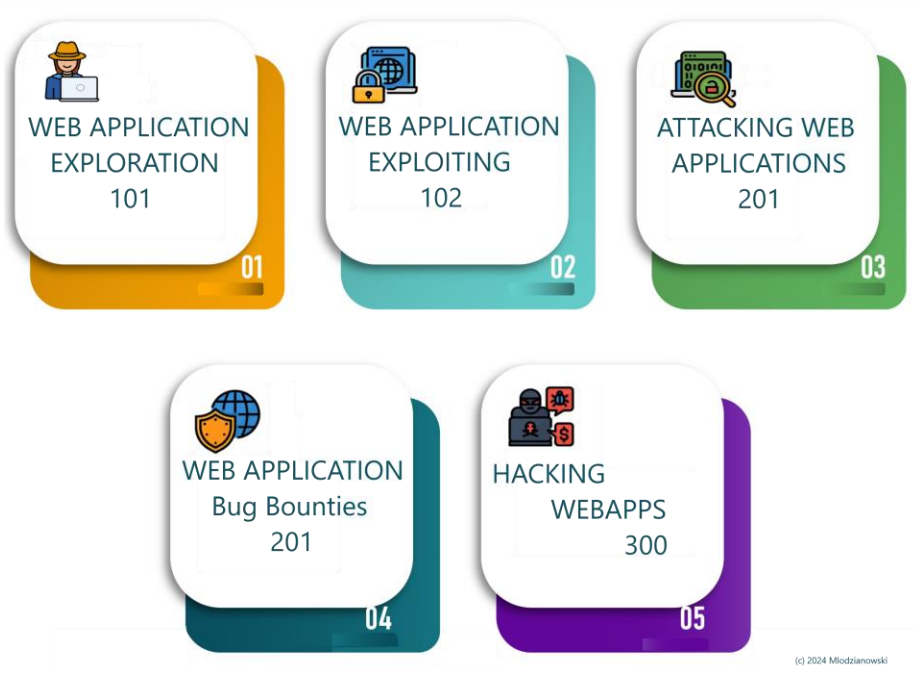
- Lee McWorther
  - Lee holds an MBA and more than 20 industry certifications in such areas as System Admin, Networking, Programming, Linux, IoT, and Cybersecurity. He is a highly sought after professional in identifying weaknesses in computer networks and systems.

- Sandra Stibbards
  - Sandra opened her investigation agency, Camelot Investigations, in 1996. Currently, she maintains a private investigator license. Sandra specializes in financial fraud investigations, competitive intelligence, counterintelligence, business and corporate espionage. Sandra has spoken at RSAC, Texas Cyber Summit and other conferences.

# Course information and disclaimer

- *This training is for educational purposes only. If you attempt these techniques without proper authorization, or on your Corporate network you are likely to get caught. If you are caught engaging in unauthorized hacking or exploration, most companies will fire you and/or you will be arrested.*

- This is an introduction course to 'Web Application Exploration' and should be considered an essentials, introductory course. This course is intended to take someone who is new to web application testing from a basic "Novice" level to at least a beginner level.






# Course information and disclaimer

- We start with the basic introduction as to what makes up a web applications, how they function, then we guide you in setting up your testing environment. The good news is you can practice the techniques we teach you online, or here on prebuilt target systems and frameworks right on your laptop that can be used for web application testing and exploitation, and over 300 labs.

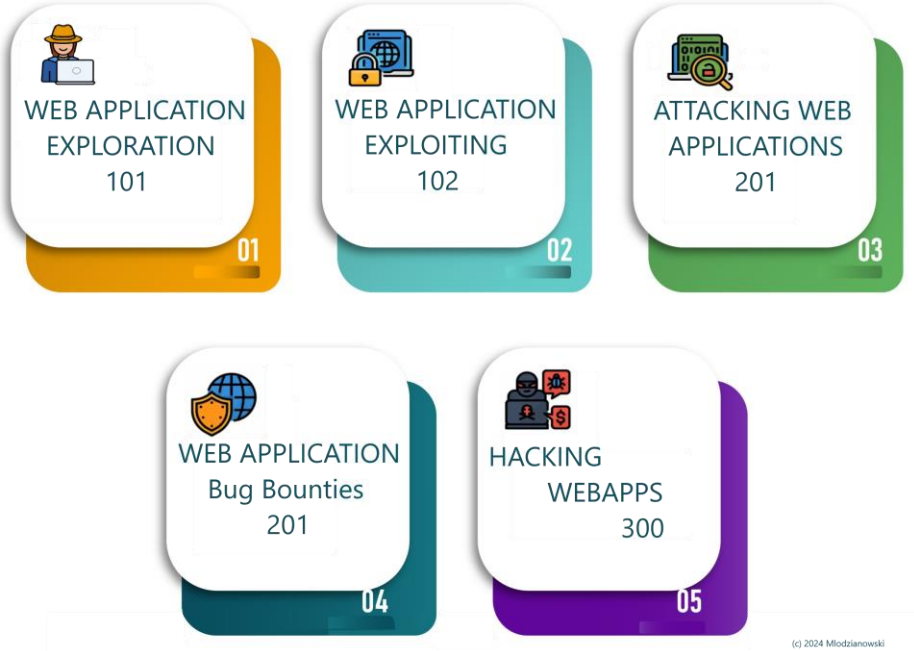
The **Lab Guide** that will be used through-out this course can be obtained here:

- https://darknets.org/lab/RSAC_WebAppLabGuide.pdf

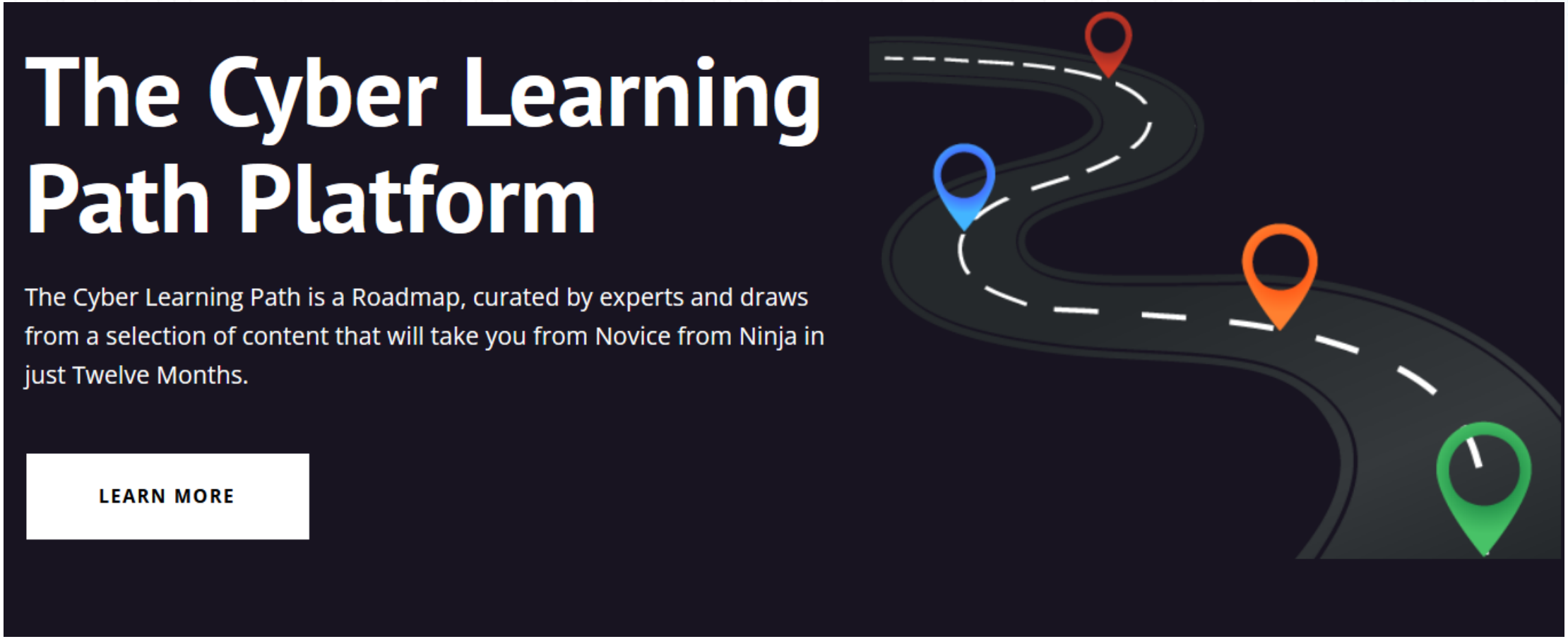
*You Can continue your Cyber Learning Path at the link below:*

*https://CyberLearningPath.org/*

Warning: Don't perform any of these activities on your corporate network.




# CyberLearningPath.org




Visit the ‘Cyber Learning Path’ for more Training Opportunities

# Cyber Learning Path

## How the Cyber Learning Path Work's

Full Cycle Learning environment, where you can Learn, Grow and build social connections and relationships.


### SecureVPN

Learn how to design, configure and deploy secure VPN networks using many different technologies.


### Dark Nets & Dark Web

Learn hands-on how to execute an investigation on the Dark Web, Darknets and the Deep Web.


### Bug Hunting 101

Bug Hunting 101 is a Hands-on hit the ground running with over 101 exersies for beginners.

### Webapp Hacking 101

Web Application Hacking takes you through applications and learning thier

### Recon 101

Learn the Basics of Recon: A required learning course to more advanced

### Pentesting 101

Get your hands-on with Penetration testing 101, taking you from 0 to 101

## Live, Pre-Record and Virtual Training

We help your value shine

### Red Team Tools & Tactics

Hands-on in-person or virtual courses available.

### Wireless Networking

Hacking Wireless Networks, Devices, performing Pentesting, Asessments & More

### Scada Device Hacking

Hacking IoT Internet of Things, Voice activated devices, routers, lighting, toilets and more.

### AI: Chat, Bots, LLM's, Gen.AI

Build your Own Chat Bot

### Darknet Investigations

Investigations

### Bug Bounty 101

Introduction to Bug Bounties

# False Cyber' Profits

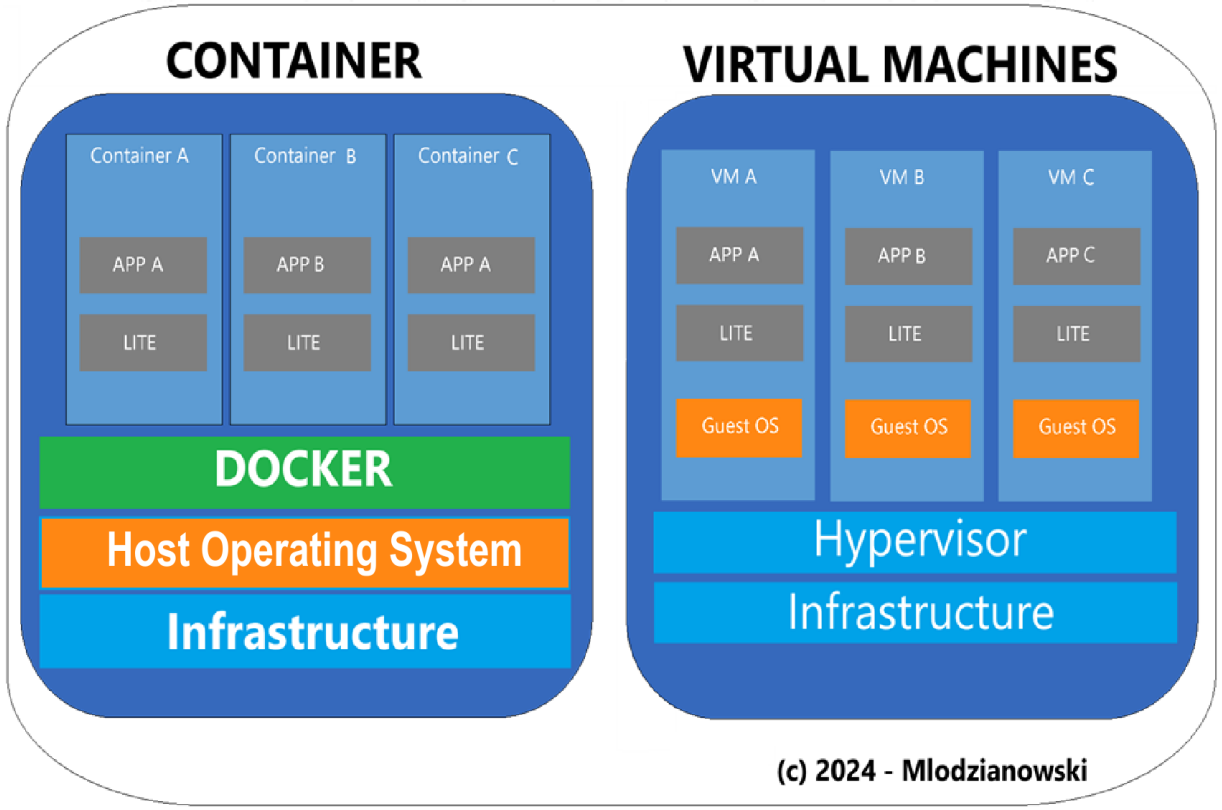
***WARNING:*** *THERE ARE PEOPLE THAT ONLY CARE ABOUT THEMSELVES AND HOW THEY CAN TAKE WHAT IS YOURS –AND MAKE IT THEIRS, AND HOW THEY CAN MAKE A BUCK OFF UNWITTING FRIENDS, STUDENTS, ATTENDEES AND THE COMMON PUBLIC.*

***THESE PEOPLE ARE: FALSE 'CYBER PROPHET'***

# Web Application Exploration

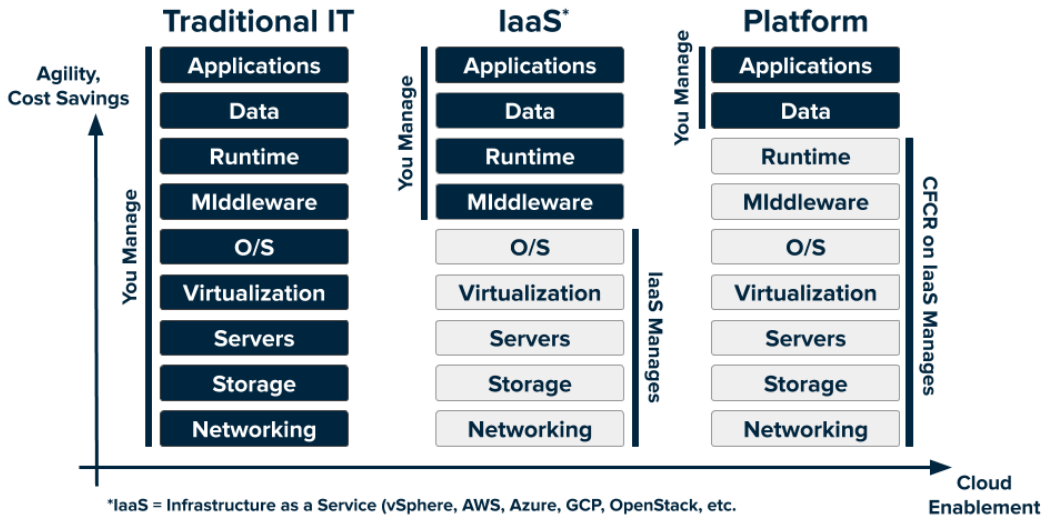
**LET'S GET STARTED LAB3-T09**

# Container's –vs- Virtualization




- A virtual machine (VM) is a software-based computer that behaves like a real physical computer. VMs are created by borrowing resources from a physical host computer or a remote server. They can run in a window as a separate computing environment, and have a CPU, memory, storage, and can connect to the internet.
- A Container is an abstracted instance of an application running as a virtualized operating system. One of the more common container types is Docker; A Container is a standard unit of software that packages up code and all its dependencies, so the application runs quickly and reliably from one computing environment to the next.
- Containers -vs- Virtual machines; as you can see a container does not require a bundled OS to be included with each instance, such as a virtual machine does, it makes use of the primary host OS, making a container easier to maintain, spin-up and tear down.

# Other Container Technology




CLOUD FOUNDRY

https://www.cloudfoundry.org/

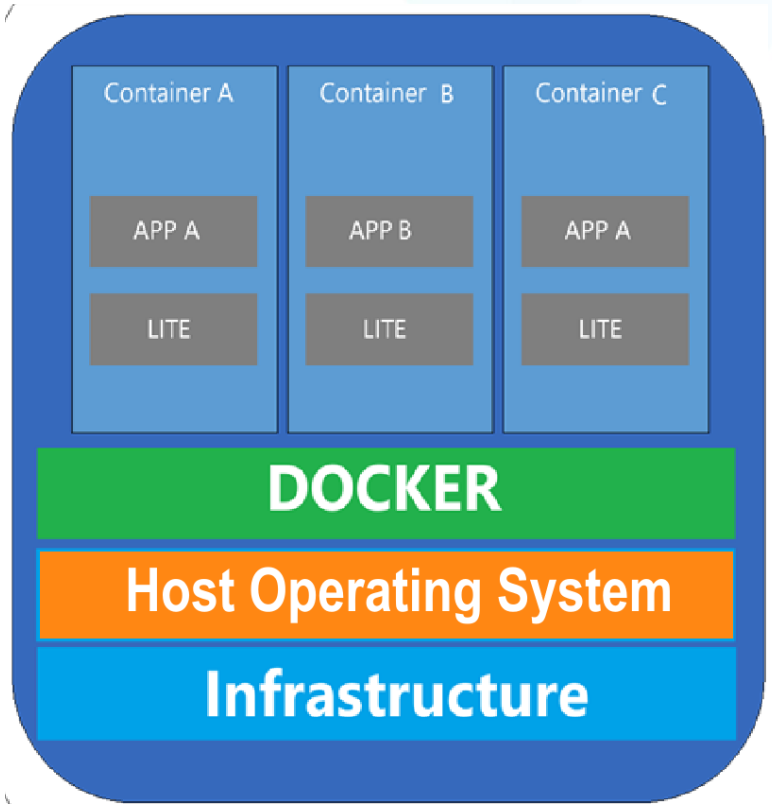
Cloud Foundry is an open-source, industry-standard cloud application platform that supports the most popular programming languages and developer frameworks right out of the box

- LXC is a traditional containerization technology that predates Docker. It is a lightweight and efficient way to isolate applications from each other.
- rkt is a security-focused container runtime developed by CoreOS (now part of Red Hat). It is designed to be easy to use and secure.
- Podman is a lightweight container tool that focuses on compatibility with Docker. It can be used to run Docker images and containers without the need for the Docker daemon.
- Containerd is an industry-standard core container runtime that powers Docker, Kubernetes, and other container platforms. It is a lightweight and efficient way to run containers.
- Kubernetes K8 is primarily known as a container orchestration platform, but it can also be used as an alternative to Docker. It can be used to run containers on a large scale
- Hyper-V Has an Isolation mode that offers enhanced security. Each container runs inside a highly optimized virtual machine and gets its own kernel.

# Exploitable Frameworks and Virtual Machines

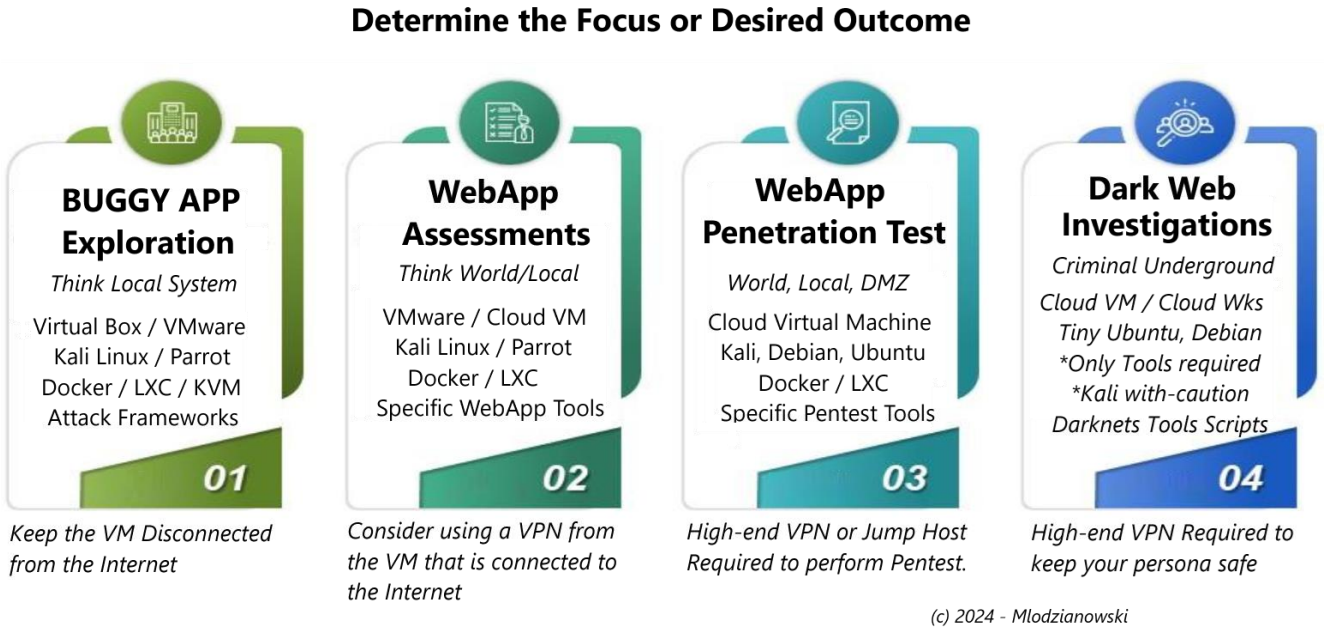
NOTICE: These VM's contain vulnerable software, hence the name, remember to use extreme caution as you use the systems. NEVER connect these vulnerable frameworks or Virtual Systems to a production environment, and never leave them unattended.

The purpose of these "Containers" are to provide you with a portable learning and exploitable environment with web applications and penetration testing tools to allow you to learn from anywhere.




# Setup your “Exploration” Workstation

- Download a Virtual Machine software
  - Virtual Box 7.0 - https://www.virtualbox.org/wiki/Downloads
  - Vmware Workstation or Player
  - Hyper-V & Azure
  - Qemu - Download
  - Vmware Fusion for Mac
  - Proxmox - Download
- Install a Linux (Debian) Based OS
  - Ubuntu Ubuntu-Download
  - Debian Lite – Minimum Image
  - Kali Linux, Parrot, BlackArch

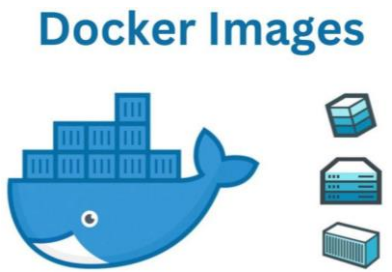



**Every System you build should have a purpose – not a Swiss Armyknife because what do we know about Swiss –cheese**

# Setup your Workstation Container Infrastructure

## Download and Install **Docker** on Ubuntu 22

- sudo apt update
- sudo apt install apt-transport-https ca-certificates curl software-properties-common
- curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg
- echo "deb [arch=$(dpkg --print-architecture) signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
- sudo apt update
- apt-cache policy docker-ce
- sudo apt install docker-ce
- sudo systemctl status docker | start|stop|restart|status
- sudo usermod -aG docker ${USER}
- Log out, then login and perform: su - ${USER}




- https://www.digitalocean.com/community/tutorials/how-to-install-and-use-docker-on-ubuntu-22-04

# Setup your Cloud Workstation

## Virtual Cloud Services (cloud systems)

- Azure Cloud: https://azure.microsoft.com/en-us/pricing/details/virtual-machines/series/
- AWS EC2 Cloud: https://aws.amazon.com/ec2/pricing/
- Vultr Virtual Machines 2.50/Mo - https://www.vultr.com
- Google Compute Engine - https://cloud.google.com/compute/
- Google Cloud Workstation - https://cloud.google.com/workstations
- Digital Ocean VM - https://try.digitalocean.com/virtualmachines
- Alibaba Cloud $2.50/Mo - https://www.alibabacloud.com/

# bWAPP Framework: Options

Option 1 – Use your own Kali or Debian Linux VM and run the darksetup.sh script
Option 2 – Use the Cloud based AWS, GCP, Azure or Digital Ocean
Option 3 – Use the Online bWAPP only instance

Option 1

- Download and install Kali 2024.1 - https://www.kali.org/get-kali/
- On the Darknets.org website located/download the darknets.org/setup.sh script
- Execute the script - ./darksetup.sh
- Execute the docker management script - ./darknets.sh

Option 2

- Connect to the service of your choice, login and access cloud-based workstations
- AWS: https://www.kali.org/docs/cloud/aws/
- Digital Ocean: https://www.kali.org/docs/cloud/digitalocean/

Option 3

- Online Option for bWAPP only | https://bwapp.hakhub.net/portal.php | https://bwapp.hakhub.net/install.php
- Also available at Pentester Academy | https://attackdefense.pentesteracademy.com/


Container:

```
docker run -d -p 80:80 raesene/bwapp
```

http://localhost/install.php

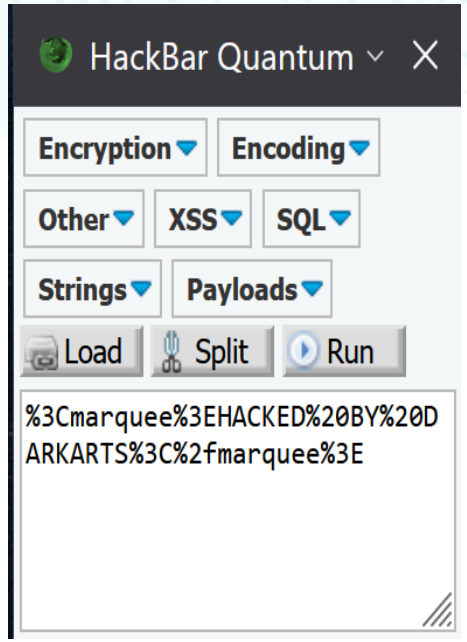
# Lab Setup – Browser Plugins

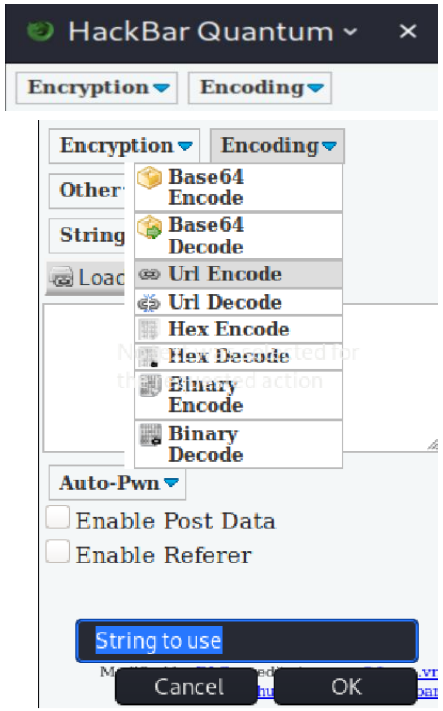
URL Encoder/Decoder - https://addons.mozilla.org/en-US/firefox/addon/url-encoder-decoder

Chrome Hackbar https://chromewebstore.google.com/detail/hackbar/ginpbkfigcoaokgflihfhhmglmbchinc

HackBar Quantum - https://addons.mozilla.org/en-US/firefox/addon/hackbartool/

Web Developer Tools: https://addons.mozilla.org/en-US/firefox/addon/web-developer/



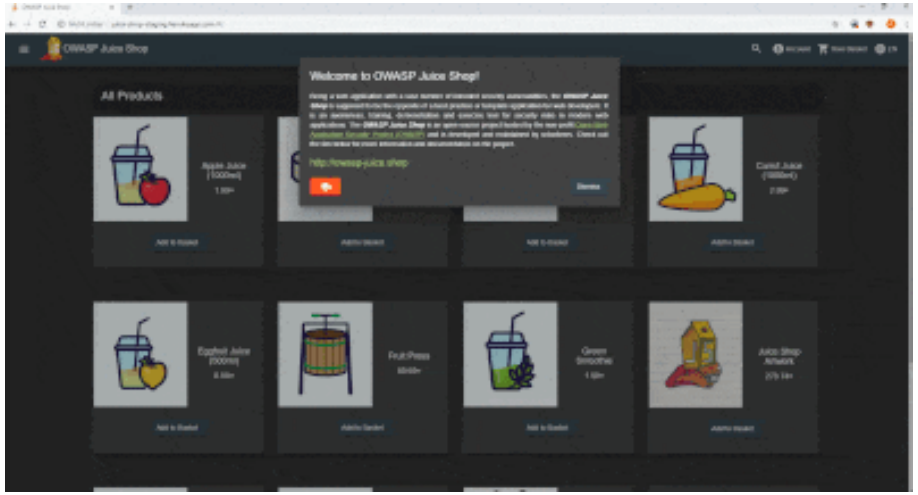

https://github.com/notdls/hackbar

- Now You could try a URL Encoded string and see if that works… Select F9
- Launches the **HackBar Quantum** plugin from Firefox, this has a build in encoders
- Select Encoding, Url Encode, replace “String to use” with marquee html config

# Juice Shop

## OWASP Juice Shop – Container

One of my favorite playgrounds is the Juice Shop, it is regularly updated, maintained and has a host of features that make it suitable for a one-stop lab. Juice Shop has capabilities to be run as a “Capture the Flag” environment as well.




You can deploy Juice Shop as a docker image downloaded from GitHub, or from Source. To run Juice Shop locally you need to have Node.js installed on your computer. The Juice Shop officially runs on versions 10.x, 12,x and 13.x of Node.js. Closely follow the official Node.js long-term support release schedule to avoid issues.

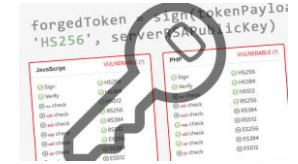
**Juice Shop: From Docker and Docker Image**

1.Install Docker on your computer.
2.On the command line run docker pull bkimminich/juice-shop to download the latest image.
3.Run docker run -d -p 3000:3000 bkimminich/juice-shop to launch the container with that image.
4.Browse to http://localhost:3000. (You can change the port to anything available on your system)

If you are using Docker on Windows - inside a **VirtualBox** VM - make sure that you also enable port forwarding from host 127.0.0.1:3000 to 0.0.0.0:3000 for TCP.

# Exploitable Frameworks and Virtual Machines

- Java Vulnerable Lab - https://github.com/CSPF-Founder/JavaVulnerableLab/
- Xtreme Vulnerable Web Application - https://github.com/tuxotron/xvwa_lamp_container
- Vulnerable SAML infrastructure - https://github.com/yogisec/VulnerableSAMLApp
- Vulnerable JWT implementations - https://github.com/Sjord/jwtdemo/
- DVNA Damn Vulnerable NODEJS Application - https://github.com/appsecco/dvna
- BWAPP – Extremely Buggy WebApp - https://sourceforge.net/projects/bwapp/
- Metasploitable 2 VM here: https://sourceforge.net/projects/metasploitable/
- WebGoat 7 - https://s3.amazonaws.com/webgoat-war/webgoat-standalone-7.1-SNAPSHOT-exec.jar




bWAPP, or a buggy web application, is a free and open source **deliberately insecure** web application developed by MME. It helps security enthusiasts, developers and students to discover and to prevent web vulnerabilities. bWAPP prepares one to conduct successful penetration testing and ethical hacking projects. "A security testing framework made for educational purposes".

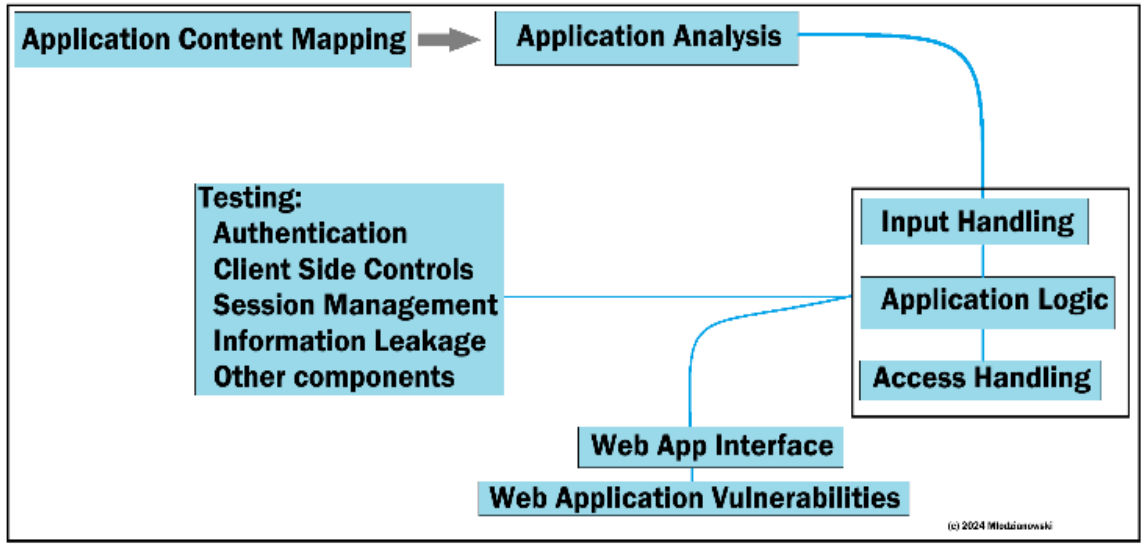
# Web Application tools & tactics

1.2.0

**Adversary Tactics:**
Throughout this course we teach real-world adversarial ***tactics, techniques and procedures, known as TTP's*** We also give you various opensource tools required to conduct an effective Web Application vulnerability assessment. When it comes to working with any type of assessment, non-traditional and out-side-the-box thinking is required to be most effective, we will explore how to simulate threat actors that will provide your defensive team with visibility into how an adversary would engage against you.

**Software Security** testing is the process of assessing and testing a system to discover security risks and vulnerabilities of the system and its data. There is no universal terminology but for our purposes, we define assessments as the analysis and discovery of vulnerabilities without attempting to actually exploit those vulnerabilities. We define testing as the discovery and attempted exploitation of vulnerabilities.
**Security testing** is often broken out according to the type of vulnerability being tested or the type of testing being done. One of the most common breakouts is:

- **Vulnerability Assessment** – The system is scanned and analyzed for security issues.
- **Penetration Testing** – The system undergoes analysis and attack from simulated malicious attackers.
- **Runtime Testing** – The system undergoes analysis and security testing from an end-user.
- **Code Review** – The system code undergoes a detailed review and analysis looking specifically for security vulnerabilities.




# HTML Injection

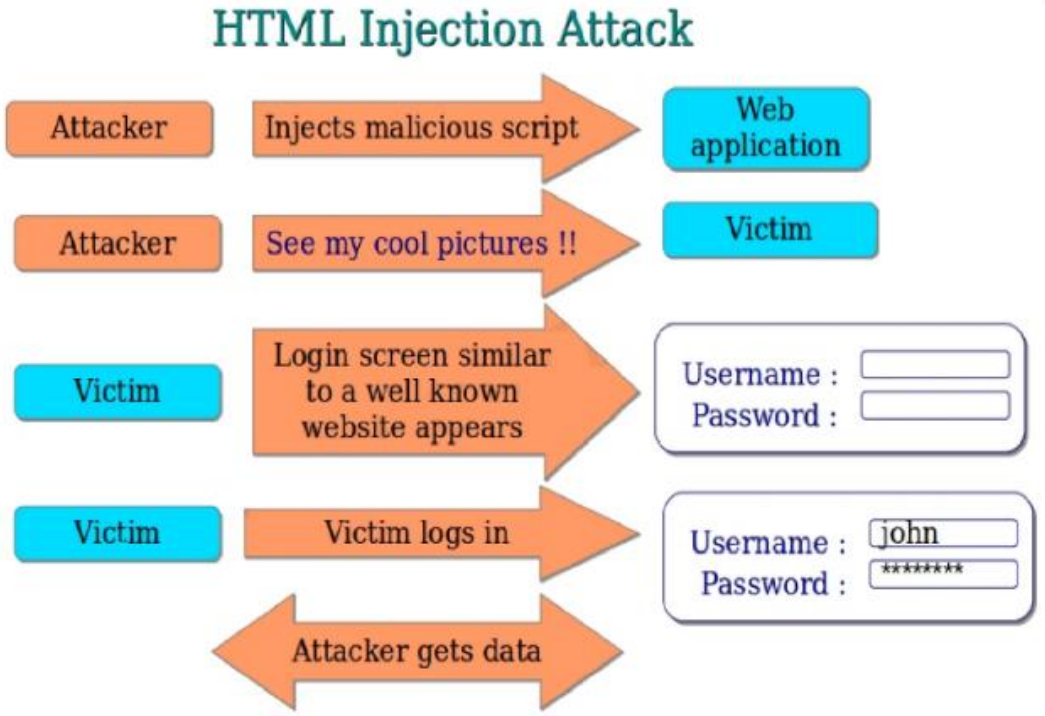
Injecting HTML code through vulnerable parts of the website. The attacker sends crafted HTML code through any vulnerable field with the purpose of changing the websites design and other information. The result will be loss of integrity to the system, displaying and/or exfiltration of data.
The Data that is sent during this type attack is very different from the intended input, the browser usually interprets the malicious (input) user data as legitimate and tries to display it. This can lead to a variety of issues, from minor website defacement to serious data breaches. Unlike other web vulnerabilities, HTML injection targets the markup language that forms the backbone of most websites.
This attack differs from other web vulnerabilities that exploit server or database weaknesses because it focuses on manipulating the structure and content of a webpage

**The Main types are:**
- Stored HTML Code
- Reflected HTML Injection

**Stored Injection** attacks occur when malicious HTML code is saved in the **webserver** and is being executed every time a user calls a linked functionality; this action is normally expected.

**The Reflected Injection,** an attack occurs when the
- website immediately responds to the (bad) mal input,
- this code is not stored permanently on the **webserver**.

Reflected HTML Injection Types:
- GET
- POST
- URL




# HTML Injection

Depending on the HTTP method, a reflected injection attack can have different results depending on GET/POST/URL, - First let's take a look at what these requests look like:

**Malicious HTML code** can "get" into the source code by *innerHTML*. innerHTML is the property of DOM document and with innerHTML, we can write dynamic HTML code. It is used mostly for data input fields like comment fields, questionnaire forms, registration forms, forums and other entry etc. Therefore, those elements are most vulnerable to HTML attack.

Now let's start with a questionnaire form, where we are filling appropriate answers and our name, and when the questionnaire is completed, an acknowledgment message is being displayed.

In the acknowledgment message, the response indicates a user's (%) name that is also being displayed.

*var user_name=location.href.indexOf("user=");*
*document.getElementById("Thank you for filling our questionnaire").innerHTML=" Thank you for filling our questionnaire, "+user;*

In the questionnaire form we would type our **malicious "HTML" code**, its message would be displayed on the acknowledgment page, thereby *injecting* that into the HTML page of the user

The same happens with the comment fields as well. If you have a comment form, you might be able to perform an injection attack, if the form is vulnerable to the HTML attack.

# Reflected HTML

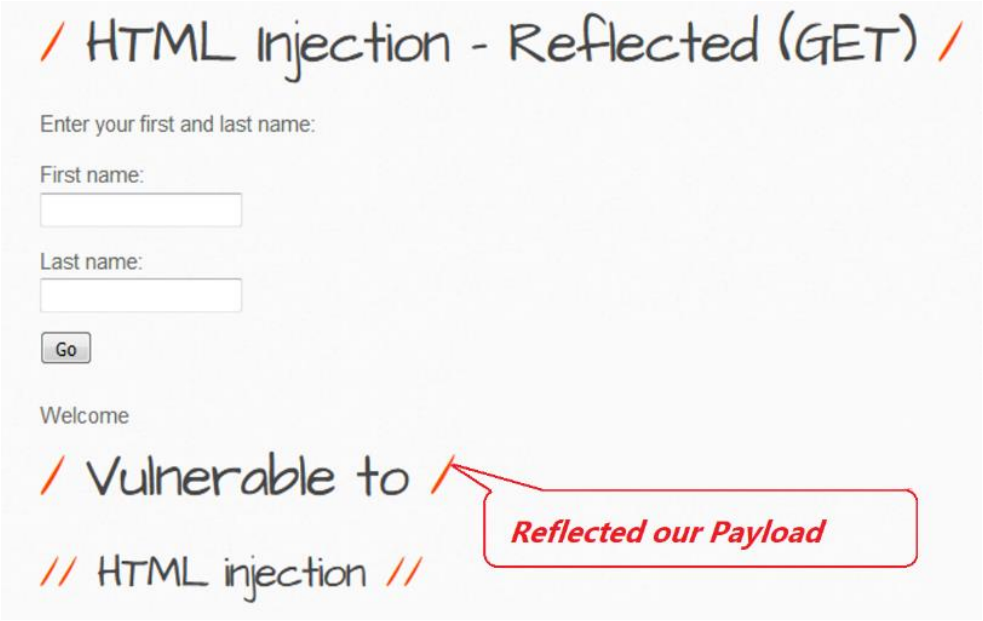
1.2.1

The reflected HTML is also known as non-Persistence. It occurs when the web application responds immediately on user's input without validating the inputs. Because the malicious script does not get stored inside the web Application database, therefore attacker will send the malicious link through watering holes or phishing attempts to trick the user.
We can test forms quite easily simply entering HTML content, and see if it is reflected back to us.

**Firstname <h1>Vulnerable to</h1>**
**Lastname <h2>HTML injection</h2>**

Normally you would see "Welcome" showing the user first/last name.

Instead we see the *inner html* data Being displayed through the html tags we used.




# HTTP/ Response, Request

1.2.1

The Most commonly used HTTP (methods) are POST, GET, PUT, and DELETE. These correspond to create, read, update, and delete operations, respectively. There are a number of other verbs but are utilized less frequently. Of those less-frequent methods, OPTIONS and HEAD are used more often than others.

- GET The HTTP GET method is used to retrieve (or read) a representation of a resource GET returns a representation in XML or JSON and an HTTP response code of 200
- (OK). In an error case, it most often returns a 404 (NOT FOUND) or 400 (BAD REQUEST).
- POST - The HTTP POST request message has a content body that is normally used to send parameters and data. Unlike using the request URI or cookies, there is no upper limit on the amount of data that can be sent and POST must be used if files or other variable length data has to be sent to the server.
- PUT -The PUT is most-often utilized for update capabilities, PUT-ing to a known resource URI with the request body containing the newly-updated representation of the original resource.
- DELETE - DELETE is pretty easy to understand. It is used to delete a resource identified by a URI.
- OPTIONS - The OPTIONS method is used by the client to find out the HTTP methods and other options supported by a web server.

# HTTP/ Response, Request

1.2.1

**TRACE** - The TRACE method is used to echo the contents of an HTTP Request back to the requester which can be used for debugging purpose at the time of development.

***What are the Status Codes?***

HTTP status codes are returned by web servers to describe if and how a request was processed.

The codes are grouped by the first digit:

1xx – Informational
2xx – Successful
200 - Code is used when a request has been successfully processed
3xx – Redirection
302 - The requested resource has been temporarily moved and the browser should issue a request to the URL supplied in the Location response header.
304 - The requested resource has not been modified and the browser should read from its local cache instead. The Content-Length header will be zero or absent because content is never returned with a 304 response
4xx – Client Error
401 - Anonymous clients are not authorized to view the requested content
404 - The requested resource does not exist on the server
5xx - Server Error
500 - An internal error occurred on the server. This may be because of an application error or configuration problem
503 - The service is currently unavailable, perhaps because of essential maintenance or overloading

# HTTP/ Response, Request

1.2.2

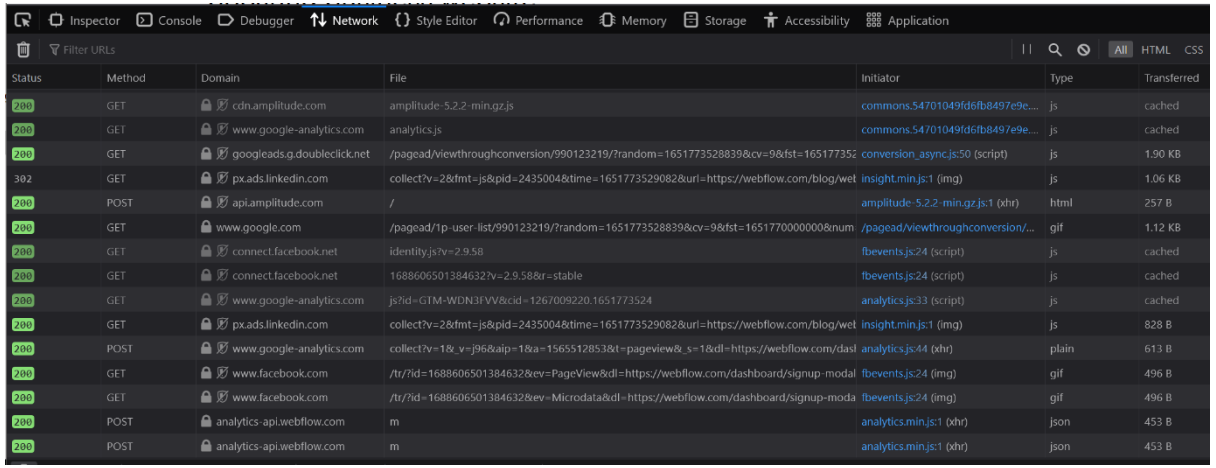
You can examine the HTTP request message with Wireshark a network analysis tool or BurpSuite.

As show below:

Sample HTTP Request message:

POST /index.html HTTP/1.1
Host: http://www.darknets.org
Connection: Keep-Alive
Accept: image/gif
Accept-Language: us-en

*Brupsuite HTTP/ Response, Request*




- HTTP Request form https://www.tutorialspoint.com/http/http_requests.htm
- HTTP Response form https://www.tutorialspoint.com/http/http_responses.htm
- HTTP Response codes https://www.tutorialspoint.com/http/http_status_codes.htm
- HTTP URL Encoding https://www.tutorialspoint.com/http/http_url_encoding.htm

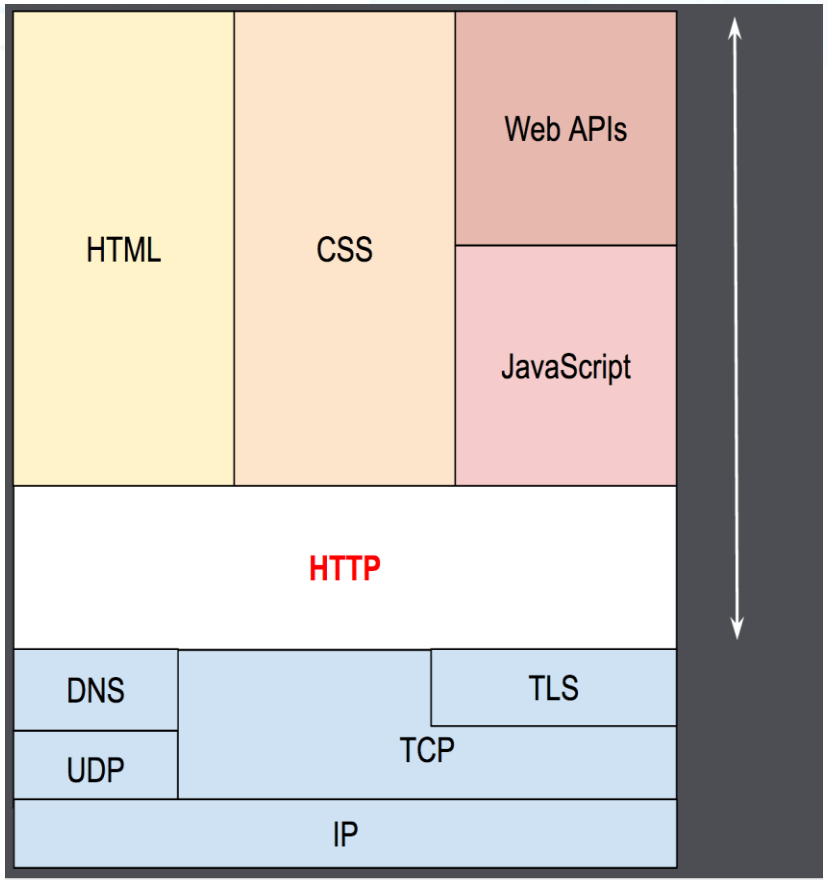
# HTTP/ Reference

1.2.2

- **HTML:** HTML is very easy to learn and there are a ton of free resources for it. If you are interested in learning more about XSS this should be your first step. If you prefer an interactive tool to learn about Javascript, I highly recommend Codecademy! https://www.codecademy.com/learn/learn-html https://www.w3schools.com/html/

- **JavaScript:** Once you have familiarized yourself with HTML, the next step is understanding Javascript since you will be using it to exploit XSS vulnerabilities. The usage of Javascript isn't just limited to when you are exploring XSS, so it's a very handy programming language to know. If you prefer an interactive tool to learn about Javascript, I highly recommend Codecademy! https://www.codecademy.com/learn/introduction-to-javascript

- **SQL:** Not part of this course, however you won't be able to exploit complex SQL injection vulnerabilities before having any SQL knowledge. As always, if you prefer an interactive course, feel free to use Codecademy! https://www.codecademy.com/learn/learn-sql http://www.sqlcourse.com/
- To determine which method is used by the website, you can start with checking the source code of the webpage, this is done by right click'ing (inspect element/view source) depending browser, It will clearly show various GETS. Going forward make sure you execute all commands (inside the VM) and not your desktop.

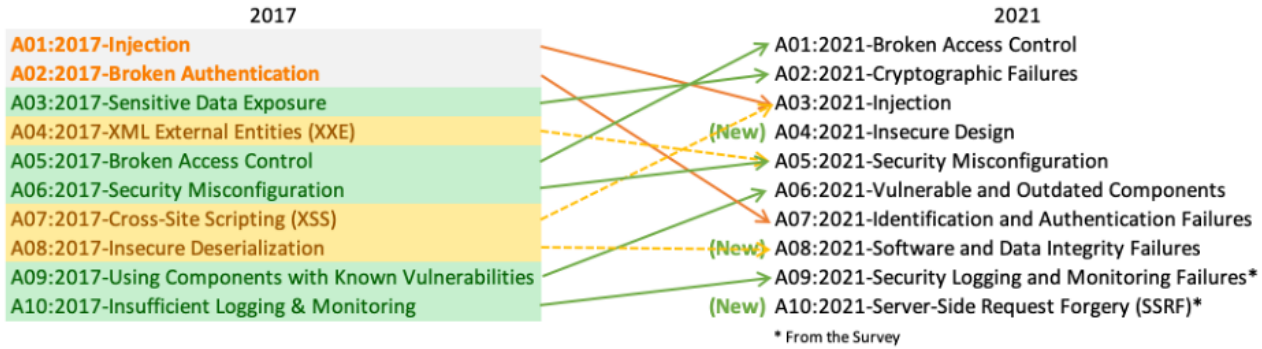



# OWASP



**OWASP Top 10**

- The Open Web Application Security Project (OWASP) is a nonprofit foundation that works to improve software security. The have a host of projects (254) of them, and many of them are active development projects.




The OWASP Top 10 is a standard awareness document for developers and web application security. It represents a broad consensus about the most critical security risks to web applications.

**https://owasp.org/www-project-top-ten/**

# What is bWAPP

/ Home /

bWAPP, or a *buggy web application*, is a free and open source deliberately insecure web application.
It helps security enthusiasts, developers and students to discover and to prevent web vulnerabilities.
bWAPP prepares one to conduct successful penetration testing and ethical hacking projects.

What makes bWAPP so unique? Well, it has over **100 web vulnerabilities**!
It covers all major known web bugs, including all risks from the OWASP Top 10 project.

bWAPP is a PHP application that uses a MySQL database. It can be hosted on Linux/Windows with Apache/II and MySQL. It can also be installed with WAMP or XAMPP.
Another possibility is to download the *bee-box*, a custom Linux VM pre-installed with bWAPP.

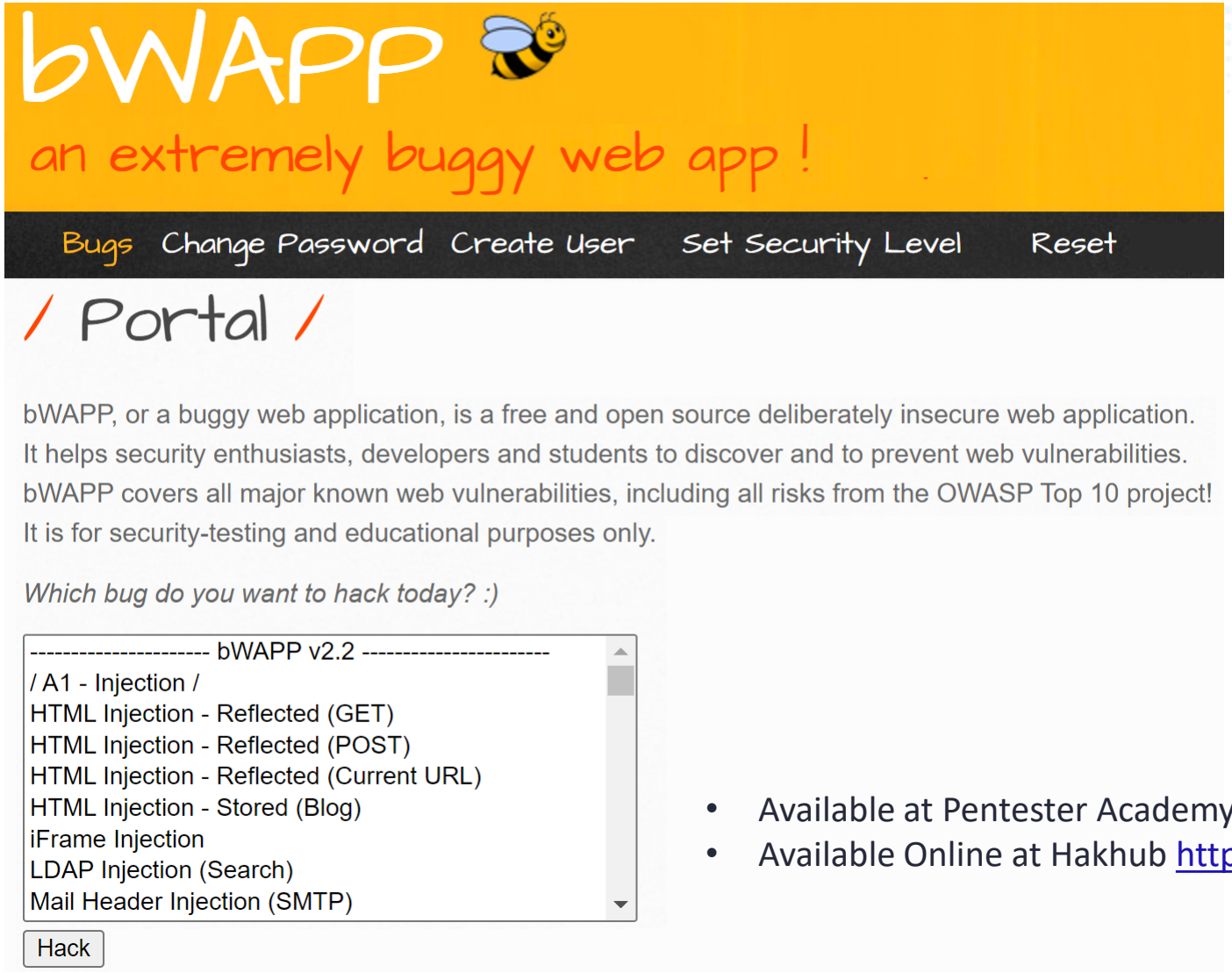
Download our **What is bWAPP?** introduction tutorial, including free exercises...

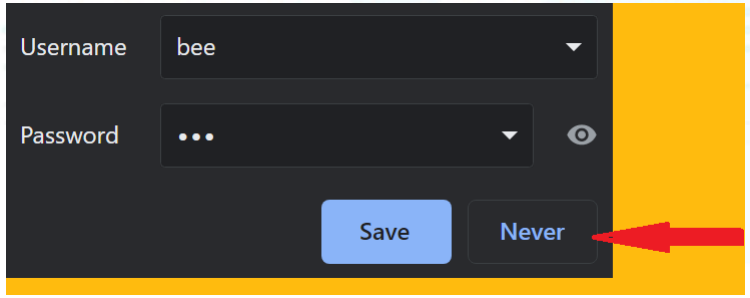
bWAPP is for web application security-testing and educational purposes only.
Have fun with this free and open source project!

Cheers, Malik Mesellem

http://www.itsecgames.com/downloads/vulnerabilities.txt

# bWAPP Online




- Available at Pentester Academy | https://attackdefense.pentesteracademy.com/
- Available Online at Hakhub https://bwapp.hakhub.net/login.php

# Local: Starting bWapp container

2.0.1

```
cd Darknets
./darknets.sh start bwapp
```

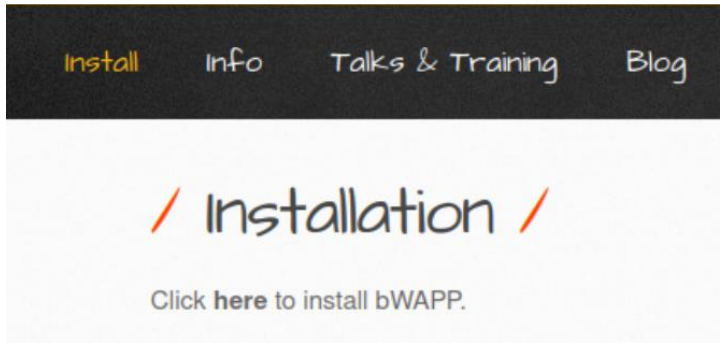
1. Start the bWAPP Container
2. Install/setup the database
3. Access via the web browser
4. Start Labs

```
┌──(root💀 darknets)-[~]
└─# ./darknets.sh start bwapp
Starting bWAPP
Adding bwapp to your /etc/hosts
127.5.0.1       bwapp was added succesfully to /etc/hosts
Running command: docker start bwapp
bwapp
DONE!

Docker mapped to http://bwapp or http://127.5.0.1

Remember to run install.php before using bwapp the first time.
at http://bwapp/install.php
Default username/password:  bee/bug
bWAPP will then be available at http://bwapp
```

# Local: Setup First run bWAPP

2.0.1

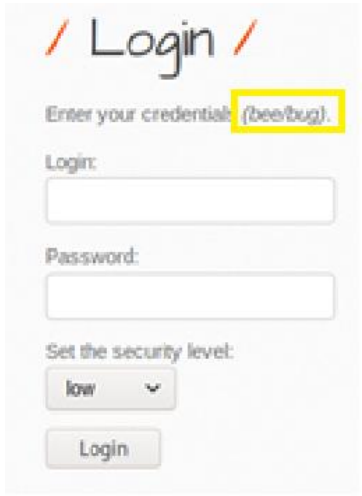
Notice the first step is to click start http://bwapp/install.php

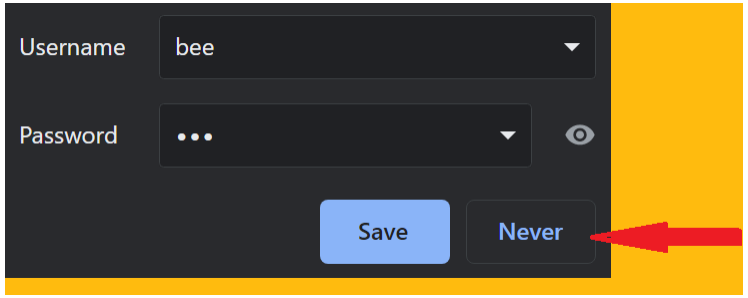


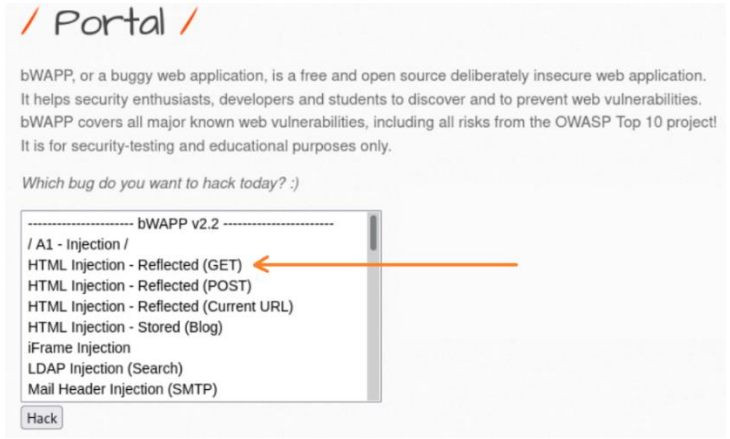

- You will need to Click the **'Here"** under installation | That will install the **database,** you may notice a "already installed" notice if you previously ran the system. That is fine just continue and select login.
- If you forget to run the installation, you will get an error mess: Connection failed: Unknown database 'bWAPP'
- All activities going forward will be done inside the VM - LAB Virtual Machine, using the web browser inside the VM, be sure to shut down the VM when you are done for the day, or if you choose on the online system.
- Note: if you did not "START BWAPP" properly you will end up at a Chinese website, that should be a good indicator you missed a step, usually not properly starting bWAPP, which enters bwapp name into the host file.

# Local: Starting the bWAPP Lab

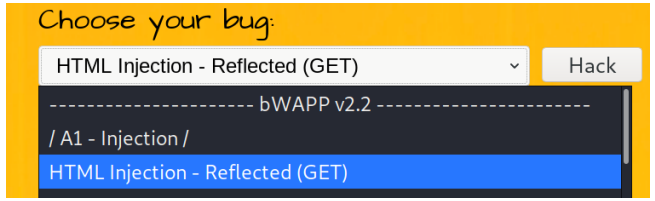
- Now let's log in to bWAPP – Click Login On Top | **user:** bee **pwd:** bug | after login we will select our Bug
- The main things we need to be concerned with in this framework is: 1. The Bug and 2. the Security Level



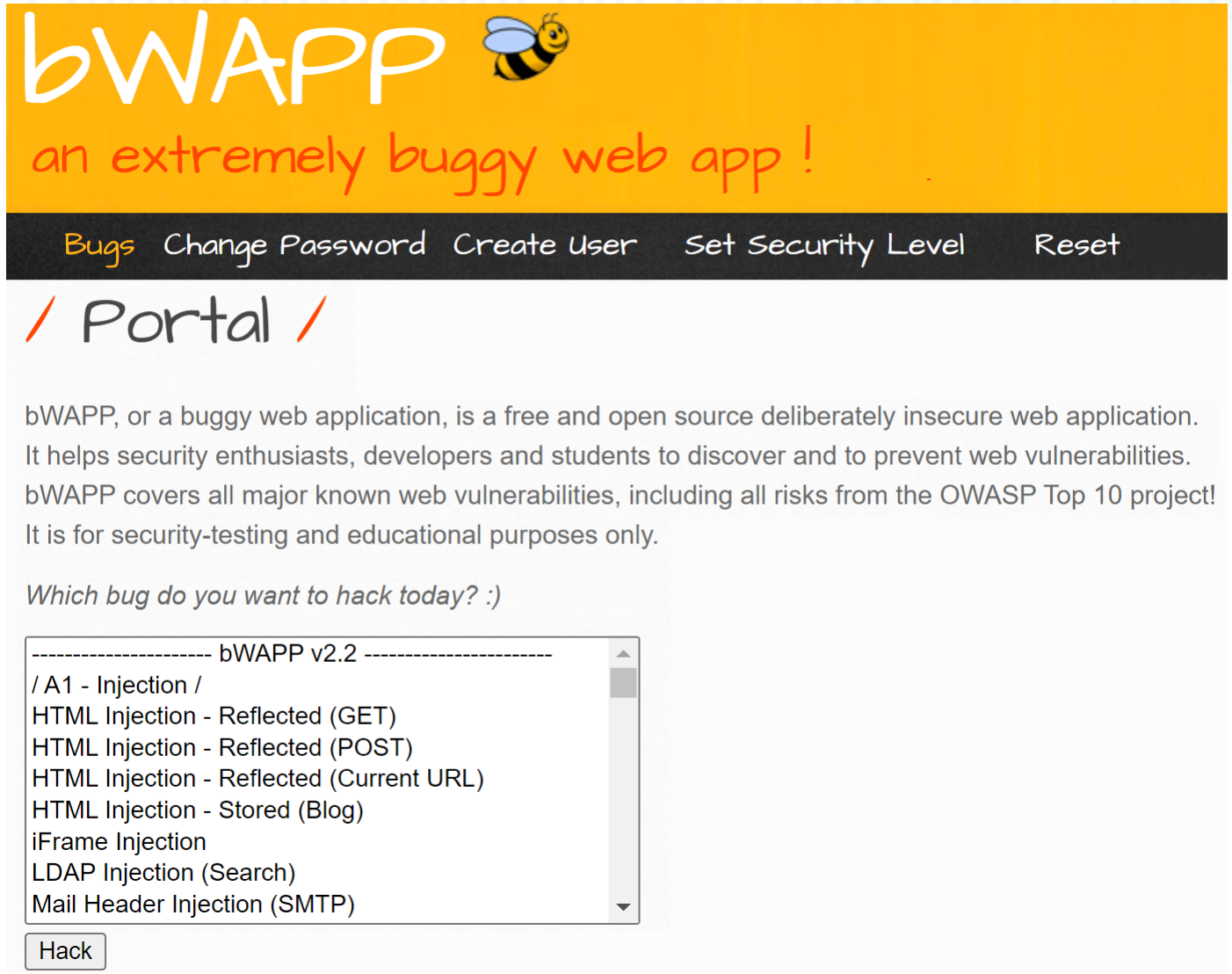





# Lab Guide

You can now switch to the

Lab Guide Page 19




# HTML Injection – Reflected (GET)

2.0.4

A. Here we enter: **First name:** darknets **Last name:** class and then go
B. If you examine the fields “first and last name” in the **URL bar**, as plain text - As you can see from the browser line
C. Try this with all three security Settings**, Low, Medium and High**, notice any difference?
D. You shouldn’t – this is a valid function of this form, normal operations, however notice that both fields require data in them in order to continue.

/htmli_get.php?firstname=darknets&lastname=class&form=submit


**Observable:** Here we want to see what the actual HTML code that was entered into this page was: You can right click on the page and select “View” or “View Source” then search for “first or last” you should see the method=”GET” you can see these fields in plain text.

Wireshark Capture of the “GET”

```
25 1.372830436   172.17.0.1      172.17.0.2      TCP      66 58598 → 80 [ACK] Seq=501 Ack=23779 Win=61440 Len=0 TSval=3701423343 TSecr=2574687446
26 1.384306918   172.17.0.1      172.17.0.2      HTTP     482 GET /htmli_get.php HTTP/1.1
27 1.384369091   172.17.0.2      172.17.0.1      TCP      66 80 → 58598 [ACK] Seq=23779 Ack=917 Win=64384 Len=0 TSval=2574687457 TSecr=3701423354
Frame 26: 482 bytes on wire (3856 bits), 482 bytes captured (3856 bits) on interface 0
Ethernet II, Src: 02:42:00:8f:8b:8a (02:42:00:8f:8b:8a), Dst: 02:42:ac:11:00:02 (02:42:ac:11:00:02)
Internet Protocol Version 4, Src: 172.17.0.1, Dst: 172.17.0.2
Transmission Control Protocol, Src Port: 58598, Dst Port: 80, Seq: 501, Ack: 23779, Len: 416
    Source Port: 58598
    Destination Port: 80
    [Stream index: 0]
    [TCP Segment Len: 416]
    Sequence number: 501    (relative sequence number)
    [Next sequence number: 917    (relative sequence number)]
    Acknowledgment number: 23779    (relative ack number)
    1000 .... = Header Length: 32 bytes (8)
  Flags: 0x018 (PSH, ACK)
    Window size value: 501
    [Calculated window size: 64128]
    [Window size scaling factor: 128]
    Checksum: 0x59ec [unverified]
    [Checksum Status: Unverified]
    Urgent pointer: 0
  Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
```

```
<div id="main">

    <h1>HTML Injection - Reflected (GET)</h1>

    <p>Enter your first and last name:</p>

    <form action="/htmli_get.php" method="GET">

        <p><label for="firstname">First name:</label><br />
        <input type="text" id="firstname" name="firstname"></p>

        <p><label for="lastname">Last name:</label><br />
        <input type="text" id="lastname" name="lastname"></p>

        <button type="submit" name="form" value="submit">Go</button>
```

# HTML Injection – Reflected (GET) Marquee

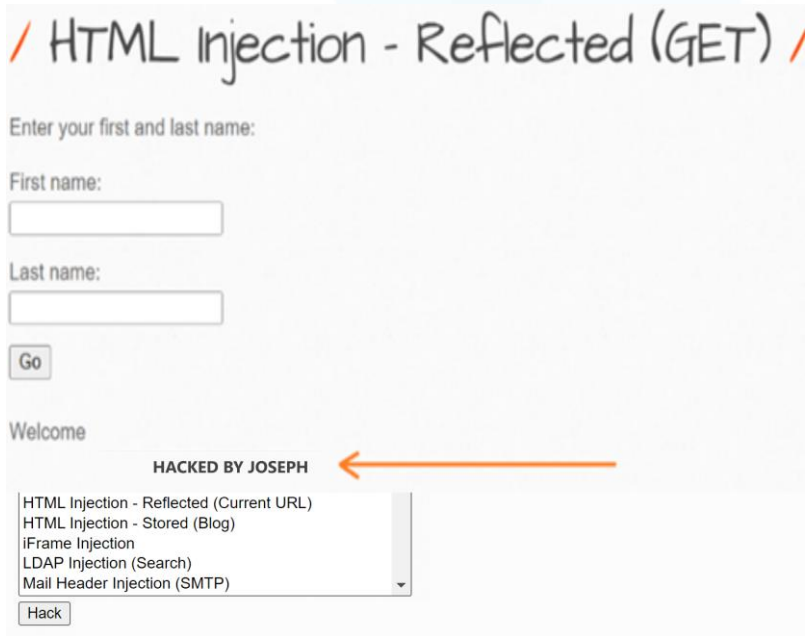
2.1.9

NOW let's try replacing the firstname= Darknets & lastname= <marquee>HACKED BY JOSEPH</marquee>

A. So lets copy or type: <marquee>HACKED BY JOSEPH</marquee> for the first name

B. Since this form validates that two fields are required be sure to fill both spaces, anything can be placed in the second field, here I use the word darknets.

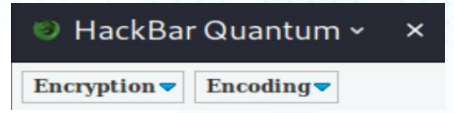
C. Try this on all three levels of security, now you will need to use the plugin

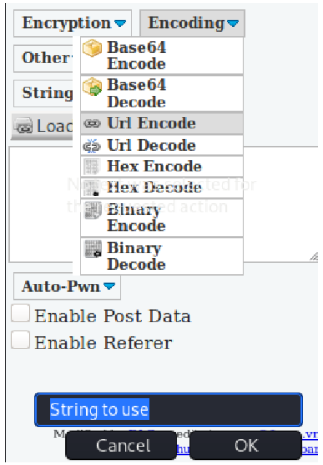


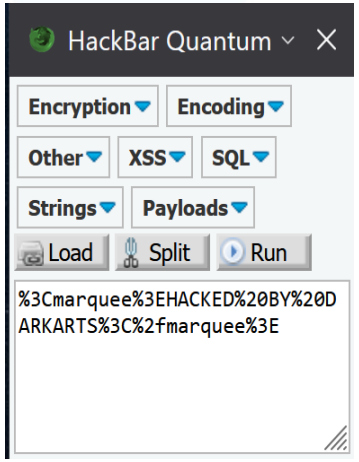

Notice what Marquee does - it reflects back your "HTML" and forces words to scroll across as a marquee. You should see something similar to what we placed, hardly a hack but is meant to show you the how easy it is to manipulate HTML forms, and when low security settings are in place it's easy to manipulate the form.

# HTML Injection – Reflected (GET) Marquee (esc) 2.1.9

- Currently our security level is set to low (as noted under “Set your Security Level” ) **[low] [Set]**
- Next lets try this same bug and wording with **medium and then high security setting**.
- You will notice they do not work.
- Now You could try a URL Encoded string and see if that works… Select F9
- Launches the **HackBar Quantum** plugin from Firefox, this has a build in encoders
- Select Encoding, Url Encode, replace “String to use” with marquee html config

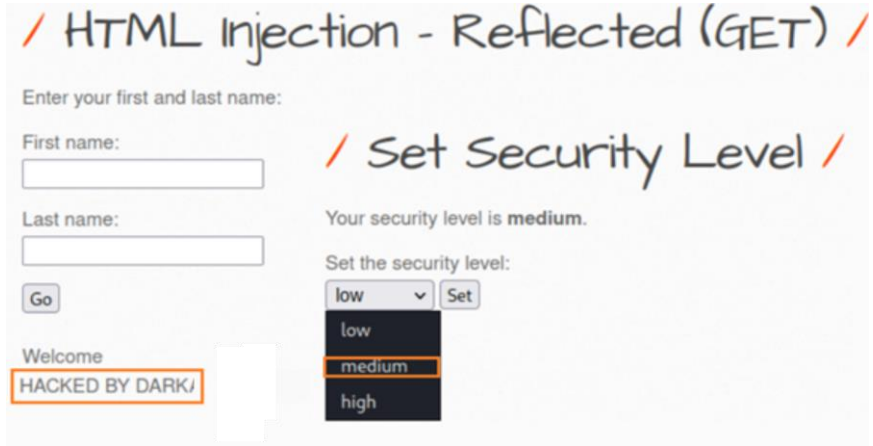



So let copy or type: <marquee>HACKED BY DARKNET</marquee> for the first name and place that in “String to use” - select ok – and notice the encoded results in the box Select and highlight that code, copy it to clipboard (ctrl c) and past it into the form for first name.

Since this form validates that two fields are required be sure to fill both spaces, anything can be placed in the second field,
here we use the word HACKED BY DARKNET


# HTML Entity Encoding



An online service is: www.url-encode-decode.com or the us the Quantum *hackbar* or your favorite URL encoder. And then past the results back into the same place first name, with the security level set to medium

After taking the input as-is The Webpage is reflecting it back. This is called html entity encoding, and it renders the code into html so the server attempts to rectify the HTML injection process. It is one of the techniques used to filter the input from the user. **URLs** can only be sent over the Internet using the ASCII character-set. Since **URLs** often contain characters outside the ASCII set, the **URL** has to be converted into a valid ASCII format. **URL encoding** replaces unsafe ASCII characters with a "%" followed by two hexadecimal digits

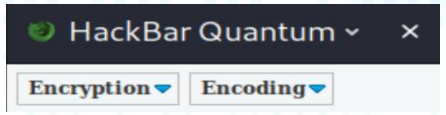
https://www.w3schools.com/tags/ref_urlencode.ASP

UTF-8 is a popular character encoding scheme for representing Unicode characters in variable length byte sequences used through-out html web pages. UTF-8 strings by web servers promiscuous interpretation results in the translation of multiple sequences into the same ASCII character (e.g. ’/’ or ’.’). This technique is known as ”Redundant UTF-8 Encoding”. When a webserver detects a request in which redundant UTF-8 occurred, this violation is generated.

- %2e%2e%2f which translates to ../
- %2e%2e/ which translates to ../
- ..%2f which translates to ../
- %2e%2e%5c which translates to ..\

**Lab Guide 23**

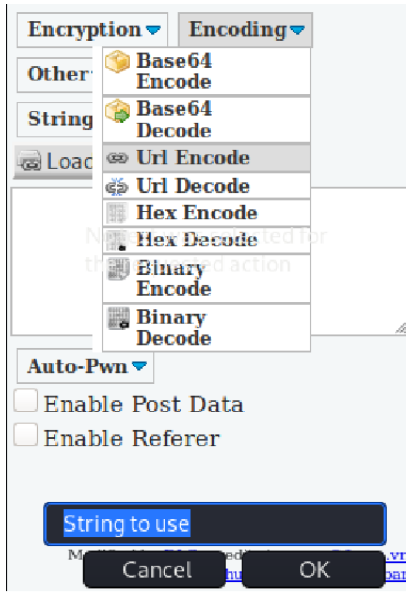
# Browser Plugins

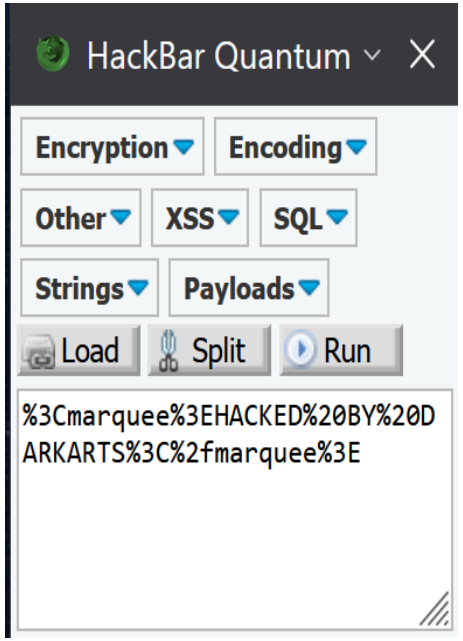



Quantum HackBar - https://addons.mozilla.org/en-US/firefox/addon/hackbartool/

https://github.com/notdls/hackbar

- Now You could try a URL Encoded string and see if that works… Select F12
- Launches the **HackBar Quantum** plugin from Firefox, this has a build in encoders
- Select Encoding, Url Encode, replace “String to use” with marquee html config



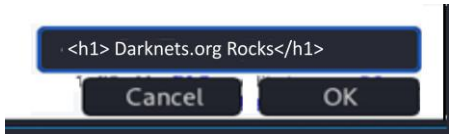




# HTML Injection - Reflected POST

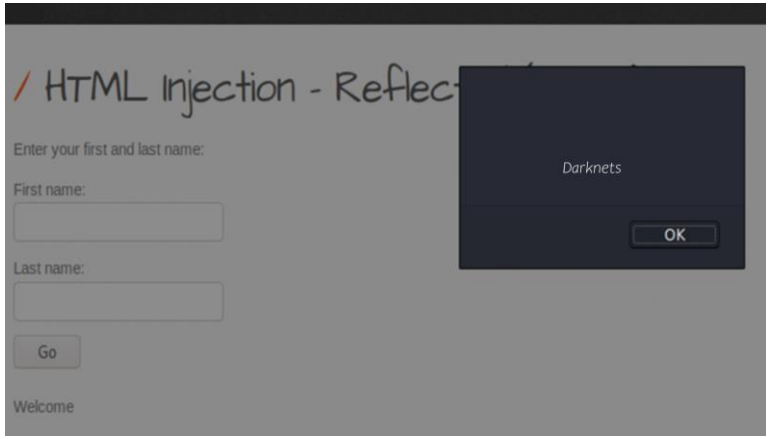
## Exercise 3 – HTML Injection – Reflected (POST)

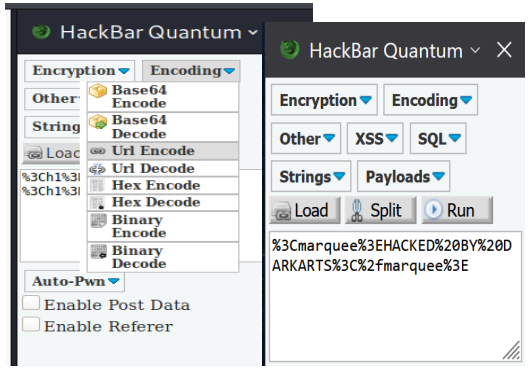
Now Let's see how we can bypass the **POST** reflective HTML injection in Bwapp. This is similar to the GET request and again we don't need burpsuite for this simple task. It can be easily done with the help of a browser. In this case we use firefox with Hackbar Quantum tool called URL encoder, which encodes the special characters in URL encoding.

1. We will start with
   a. With the security level set to medium.
   b. **First name: <h1>HackedBy</h1>**
   c. **Last Name: <h1>Darknets.org</h1>**




2. After entering Go – we notice the output is reflected as it was entered, meaning there is a filter not allowing execution of our html tag's
3. So next let's try to bypass this medium level filter using url encoding, as we did in the previous exercise we will utilize the HackBar Quantum to do a URL Encode
4. You can go to url encode/decode or select **F9** from Firefox to bring up the HackBar Quantum. Select encoding, enter both strings to encode: Encoding>URL Encode> at the bottom where "string to use" shows up, enter the two strings




# XSS Reflective (GET)



## Chapter 2.3 XSS – Reflective (GET)

**What is Cross Site Scripting – XSS?**

Cross-site scripting (XSS) is a type of computer security vulnerability typically found in web applications. XSS enables attackers to *inject client-side scripts* into web pages viewed by other users. XSS is Injecting a script into the parameter of a url - A cross-site scripting vulnerability may be used by attackers to bypass access controls such as the same-origin policy. In 2007 Cross-site scripting carried out on websites accounted for roughly 84% of all security vulnerabilities.

Let's walk through the environment we are going to use for these labs and explain each of these three types of attacks.

**Types of XSS**

- Reflective XSS
- Stored XSS
- DOM Based XSS

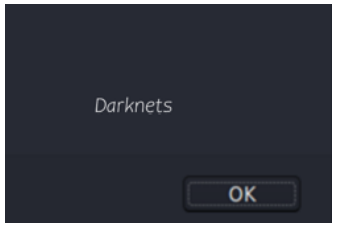
The environment is Kali Linux, and tools from the OWASP broken web applications project, it is a suite of servers/services, we will continue to use bwapp and webgoat through-out these labs, but there are many others.

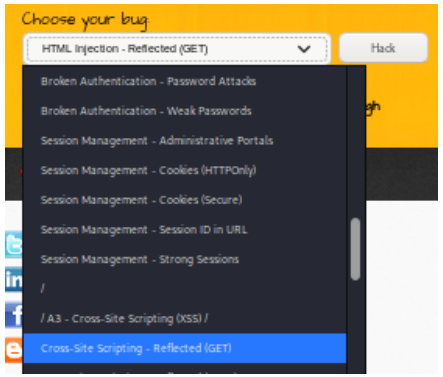
# Reflected XSS - alert

**Reflected XSS**
Reflected XSS is the most common type of XSS. It occurs when the malicious payload is part of the request that the victim's browser sends to the vulnerable site. This type of attack is called "reflected" because an input field of the HTTP request sent by the browser, is immediately repeated on the output page. The attacker uses Phishing emails and other social engineering techniques to convince the victim to open the malicious link.
**Reflected XSS** isn't a persistent attack, so the attacker needs to deliver the payload to each victim. We will continue to use bWAPP for these exercise's as well.
Start the bWAPP and open your browser and connect to bee-box login.
**Select: A3** - *Cross-Site Scripting – Reflected (GET) with security settings set to Low*

**<script>alert('Darknets')</script>**

To test if the input fields are vulnerable, we place the following script:




Since both fields are required insert the script in First name field and in Last name field we can insert anything we want.
If it is vulnerable, it will show us an "popup" alert box that says: **Darknets** Notice it shows us an popup "alert box" this means that it is vulnerable.

As a side note we are using firefox for a reason, Google Chrome uses an Anti-XSS filter.

# SQL Injection (GET/Search)



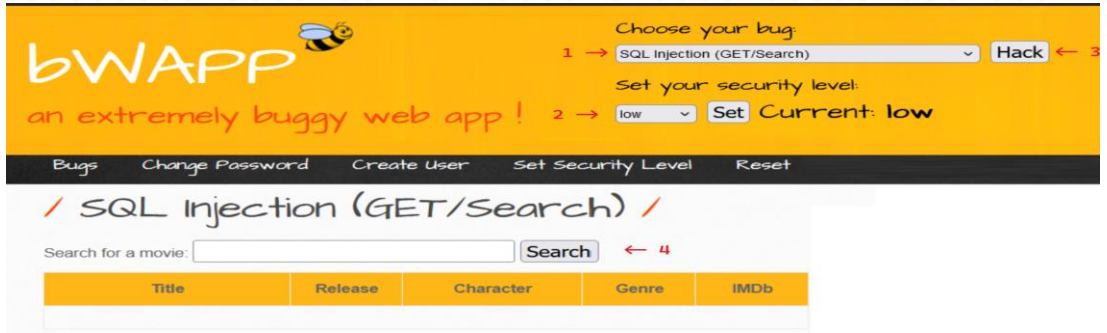
SQL injection, or SQLi, is a common attack on database-driven websites that occurs when an attacker uses unintended data from an untrusted source to construct a SQL query. This allows the attacker to interfere with the queries that an application makes to its database, and view data that they are not normally able to retrieve

A successful SQL injection attack can:

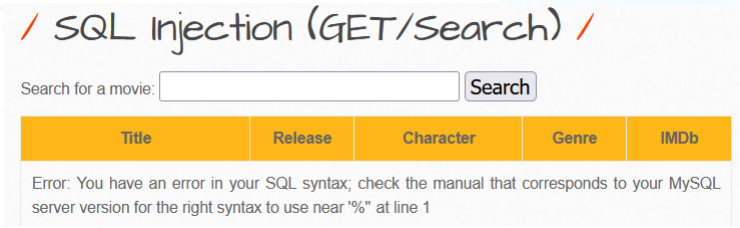
- Read sensitive data from the database
- Modify database data
- Execute administration operations on the database
- Spoofing identity
- Tamper with existing data
- Cause repudiation issues
- Expose all data on the system
- Destroy data or make it unavailable
- Become administrators of the database server

https://owasp.org/www-community/attacks/SQL_Injection/

# SQL Injection (GET/Search)




1




If you enter an ‘ you will notice an error

2


now lets try ‘ order by 5 -- -

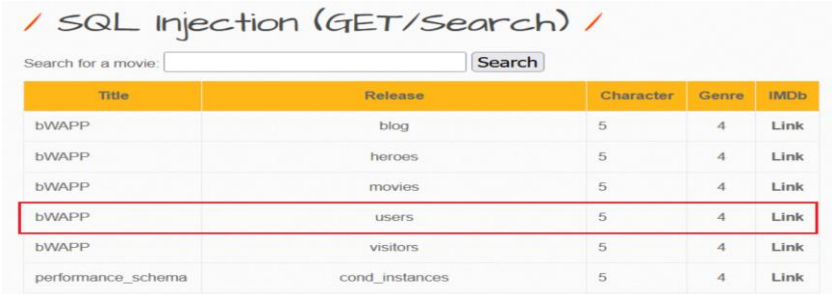
Setup bWAPP for SQL Injection (GET/Search)

- 1. Select the appropriate bug – SQL Injection (GET/Search)
- 2. Select the appropriate Security Level (LOW)
- 3. Select Hack “SQL Search Movie Box Appears”
- 4. Enter your SQL commands to search for users

**Lets explore the SQL database and see if maybe bWAPP has a user table, use the following code:**

' and 1=0 union all select 1,table_schema,table_name,4,5,6,7 from information_schema.tables where table_schema != 'mysql' and table_schema != 'information_schema' -- -

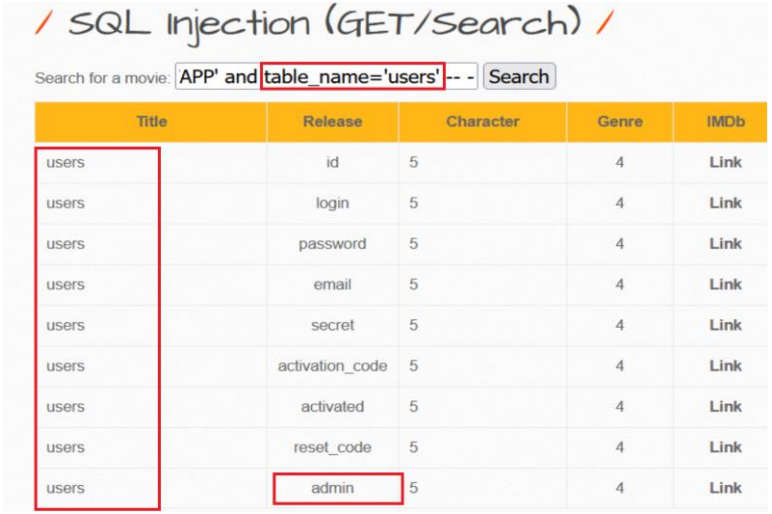
3



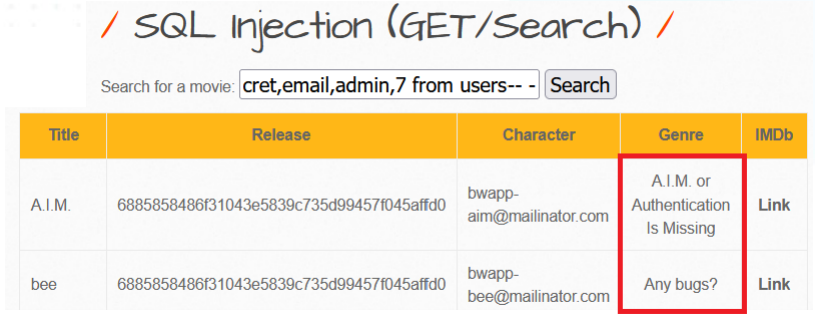

# SQL Injection (GET/Search)



4




5




' and 1=0 union all select 1,login,password,secret,email,admin,7 from users-- -

Here we can see the hashed password, you can use “John the Ripper” to crack it.

6

```
adminx@darkweb:~$ john --format:raw-sha1 /home/adminx/hashes.txt
Using default input encoding: UTF-8
Loaded 1 password hash (Raw-SHA1 [SHA1 256/256 AVX2 8x])
Proceeding with wordlist:/usr/share/john/password.lst
Proceeding with incremental:ASCII
bug              (?)
1g 0:00:00:00 DONE 3/3 (2024-04-28 14:11) 3.333g/s 763326p/s 763326c/s
```

**Another way to see users, and specific names is with the table name ‘users’**

' and 1=0 union all select 1,table_name, column_name,4,5,6,7 from information_schema.columns where table_schema != 'mysql' and table_schema != 'information_schema' and table_schema='bWAPP' and table_name='users' -- -

7

https://owasp.org/www-community/attacks/SQL_Injection
https://book.hacktricks.xyz/pentesting-web/sql-injection

# Webgoat – 7.1

- https://github.com/WebGoat/WebGoat/releases
- https://hub.docker.com/r/webgoat/webgoat-7.1/


- Browse to http://localhost:8080/WebGoat and happy hacking !

**WARNING 1:** *While running this program your machine will be extremely vulnerable to attack. You should disconnect from the Internet while using this program.* WebGoat's default configuration binds to localhost to minimize the exposure.
**WARNING 2:** *This program is for educational purposes only. If you attempt these techniques without authorization, you are very likely to get caught. If you are caught engaging in unauthorized hacking, most companies will fire you. Claiming that you were doing security research will not work as that is the first thing that all hackers claim.*

# Additional Tools:

- WebApp Analyzer - https://addons.mozilla.org/en-US/firefox/addon/wappalyzer/
- Foxy Proxy - https://getfoxyproxy.org
- HTTP Header Live - https://addons.mozilla.org/en-US/firefox/addon/http-header-live/
- HackBar Quantum - https://addons.mozilla.org/en-US/firefox/addon/hackbartool/
- Mitaka Plugin - https://addons.mozilla.org/en-US/firefox/addon/mitaka/
- Cookie Editor - https://addons.mozilla.org/en-US/firefox/addon/cookie-editor/
- BuildWith Profiler - https://addons.mozilla.org/en-US/firefox/addon/builtwith/
- User Agent Switcher - https://addons.mozilla.org/en-US/firefox/addon/uaswitcher/
- Sputnik Osint - https://addons.mozilla.org/en-US/firefox/addon/sputnik-osint/
- Cookie Quick - https://addons.mozilla.org/en-US/firefox/addon/cookie-quick-manager/
- Easy XSS - https://addons.mozilla.org/en-US/firefox/addon/easy-xss/
- Tamper ff - https://addons.mozilla.org/en-US/firefox/addon/tamper-data-for-ff-quantum/

# Apply What you have learned Today

Next week you should:

- Revisit the bWapp and WebGoat frameworks in your lab VM and review what you learned, continue to apply what you have learned by working the 100 exercises that are part of these two frameworks.
- Review your own internal websites (with permission) for these simple but often overlooked vulnerability and coding mistakes
- Visit with your team and discuss the programming Vulnerability class types that common and make sure the team understands their significance.

In the next three months:

- You should engage and review the additional frameworks we provided for you to study on and “Expand” your Web Application Vulnerability view.
- Install in your lab environment and start using automated tools such as nmap, OpenVAS, BurpSuit and ZAP (OWASP). Visit owasp and the top 10. https://owasp.org/www-project-top-ten/
- Continue to Learn the Process, the impact and advancements being made in WebApp Exploitation

# Continue Your Learning

1. With Team and leadership approval, work to identify issues in your own environment, especially with home grown applications
2. Remember the Wapp and WebGoat frameworks in your lab VM and review have over 100 exercises that are part of these two frameworks, so continue to work through the labs.
3. Engage and review the additional frameworks we provided for you to study on and  in the first couple slides and “Expand” your Web Application Vulnerability view.
4. Install in your lab environment and start using automated tools such as nmap, OpenVAS, BurpSuit and ZAP (OWASP). Visit owasp and the top 10. https://owasp.org/www-project-top-ten/
5. Continue to Learn the Process, the impact and advancements being made in WebApp Exploitation
6. Engage with us on Discord, Telegram, Facebook, Linkedin and continue to learn.

# Complete this Course and Obtain your Certificate:

## Complete the LAB3-T09 Course and All the Labs?

Download your Certificate: https://darknets.org/lab/WebAppExploration.png


# Thank you !

**@cedoxx – Twitter – Joseph Mlodzianowski**