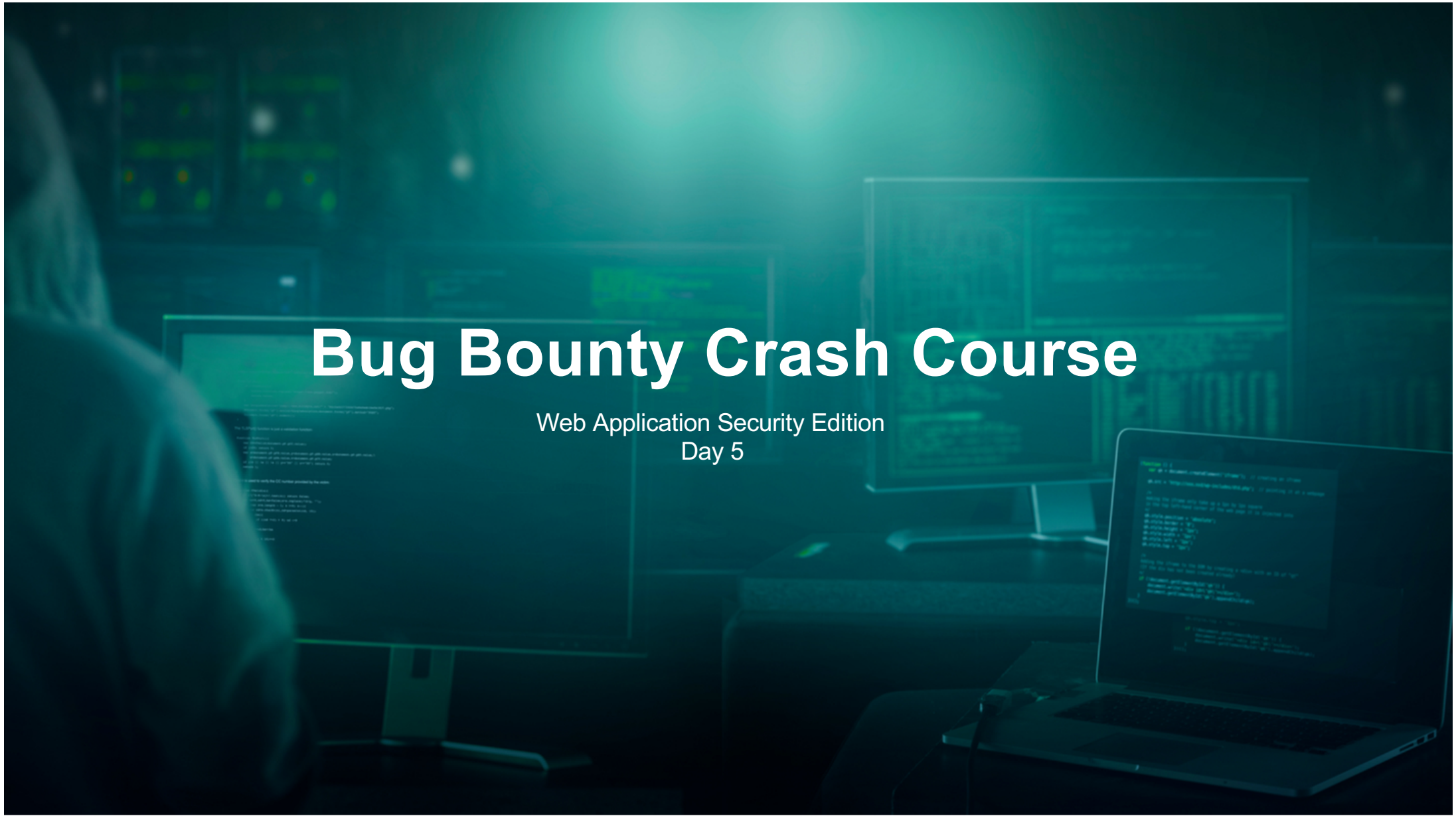


# Bug Bounty Crash Course

Web Application Security Edition
Day 5

# OWASP Top 10 : A1 Injection

**A1 :2017**

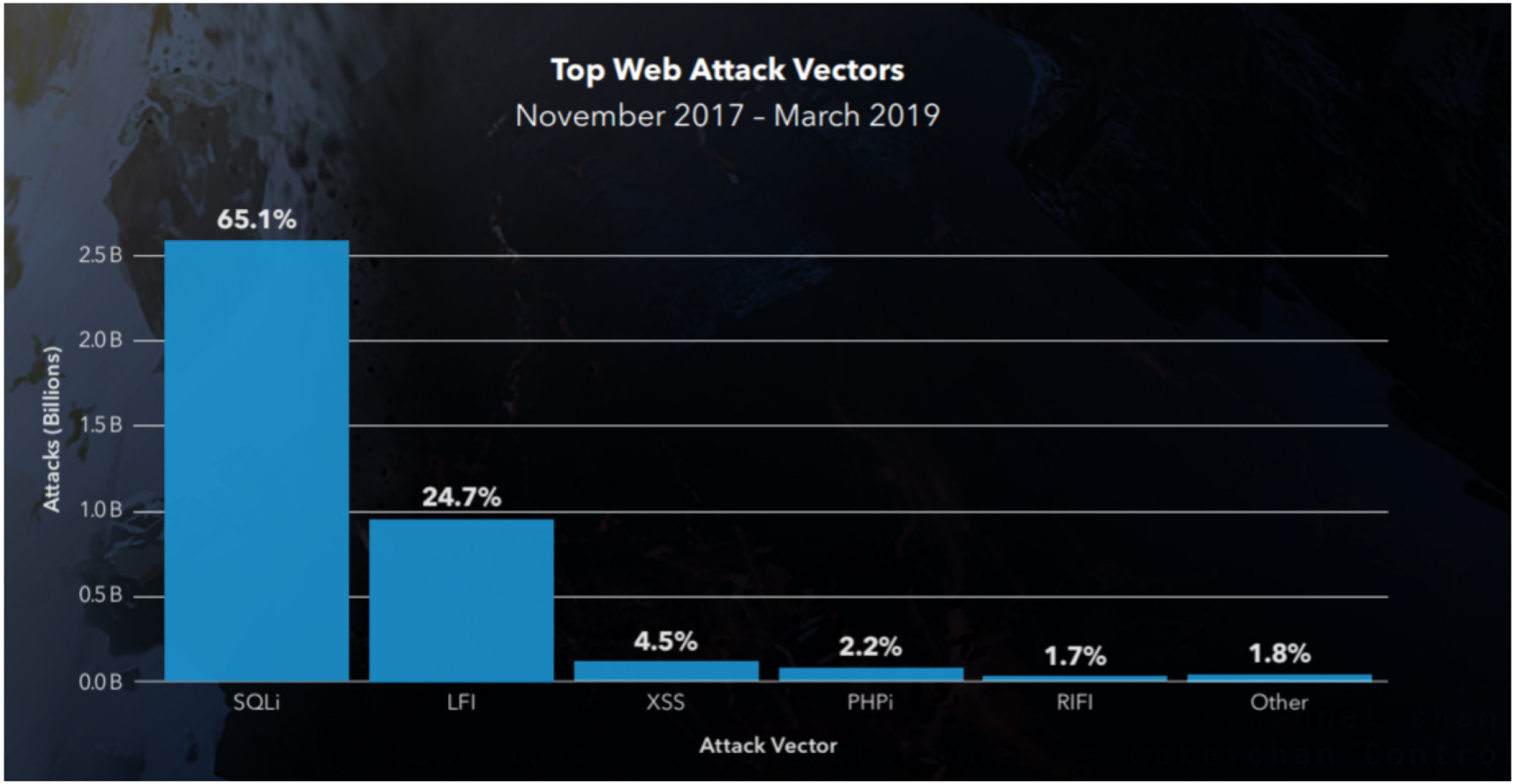
## Injection




| App. Specific | Exploitability: 3 | Prevalence: 2 | Detectability: 3 | Technical: 3 | Business ? |
|---|---|---|---|---|---|
| Almost any source of data can be an injection vector, environment variables, parameters, external and internal web services, and all types of users. Injection flaws occur when an attacker can send hostile data to an interpreter. | | Injection flaws are very prevalent, particularly in legacy code. Injection vulnerabilities are often found in SQL, LDAP, XPath, or NoSQL queries, OS commands, XML parsers, SMTP headers, expression languages, and ORM queries.<br><br>Injection flaws are easy to discover when examining code. Scanners and fuzzers can help attackers find injection flaws. | | Injection can result in data loss, corruption, or disclosure to unauthorized parties, loss of accountability, or denial of access. Injection can sometimes lead to complete host takeover.<br><br>The business impact depends on the needs of the application and data. | |

Source: OWASP

# OWASP Top 10 : A1 Injection




Source: https://www.cbronline.com/news/sql-injection-attacks

# OWASP Top 10 : A1 Injection

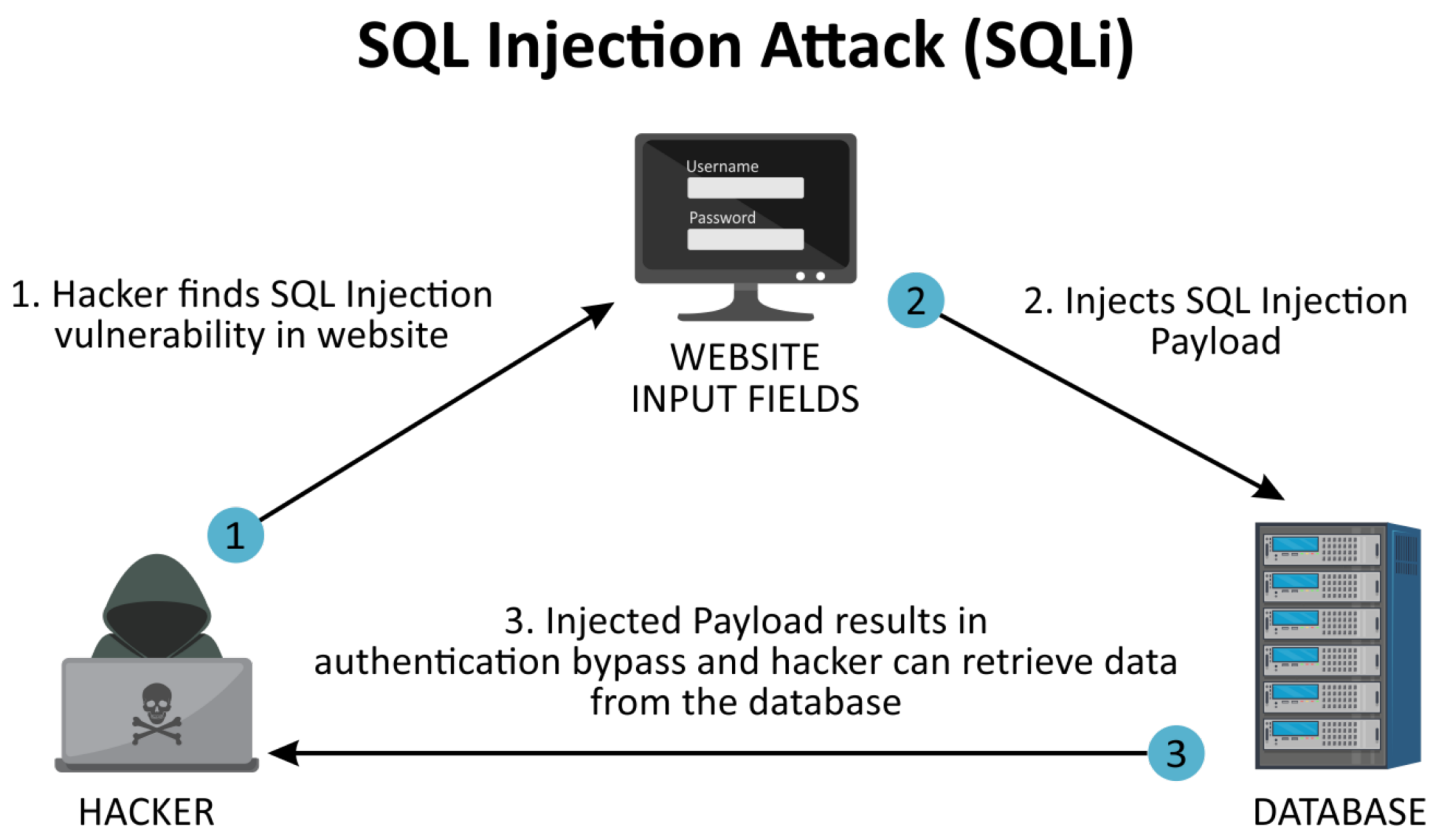

## SQL Injection Attacks Represent Two-Third of All Web App Attacks

**When Local File Inclusion attacks are counted, nearly nine in 10 attacks are related to input validation failures, Akamai report shows.**

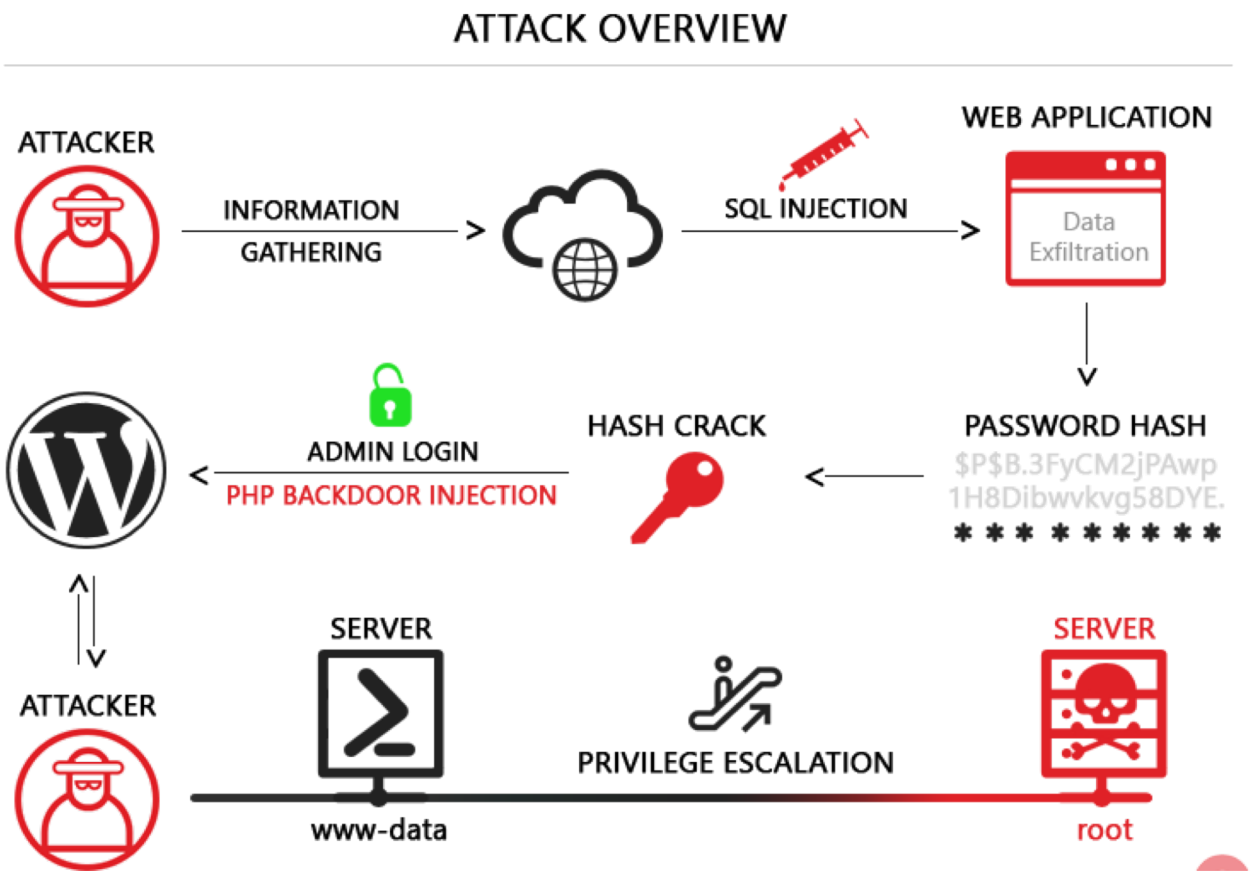
## Hacker breached 60+ unis, govt agencies via SQL injection

A hacker tied to the November 2016 penetration of the US Election Assistance Commission and subsequent database sale has successfully targeted 60+ government agencies and universities by leveraging the same attack method: SQL injection.


# SQL Injection



# Total Compromise




Source: https://www.acunetix.com/blog/articles/exploiting-sql-injection-example/

# Databases – SQL Based

- Structured storage and query of data
- Relational DB System - MySQL, PostgreSQL
- Embedded DB - SQLite



# Lab: SQL Basics

https://www.attackdefense.com/paredirect?cid=1801

# Authentication Bypass

## SQL Injection

WEB PAGE

USERNAME: John

PASSWORD: **********

select * from users where username='john' and password ='doe'

WEB PAGE

USERNAME: 1' or '1'='1

PASSWORD: **********

select * from users where username='1' or '1'='1' and password ='1' or '1'='1'

# Authentication Bypass

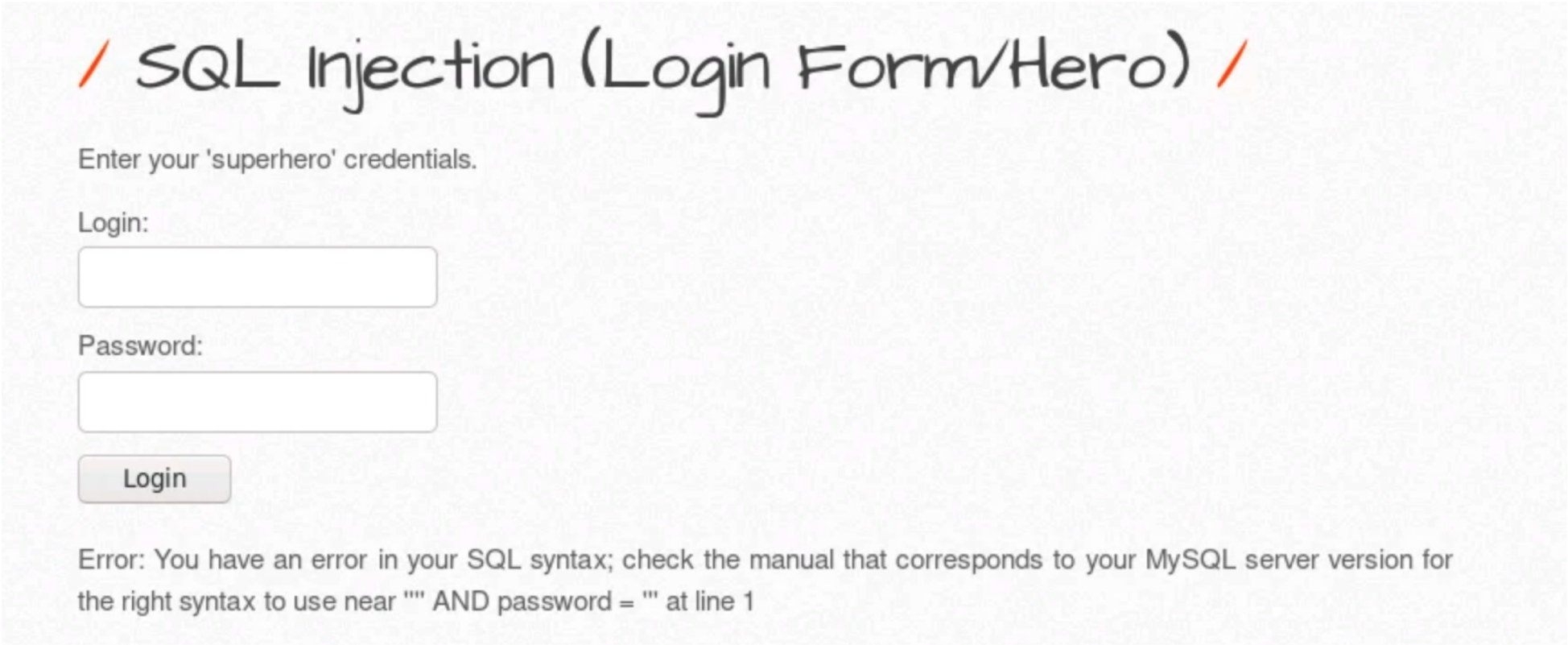
- Identifying vulnerability

- Trying generic SQLI Payloads
  - ' or '1'='1
  - ' or '1'='1' --  (Comment)
  - ' or '1'='1' #  (Comment)

- Resultant SQL Query:
  - select * from users where login='' or '1'='1' and password='' or '1'='1'; ← always True
  - select * from users where login='' or '1'='1' -- ' and password=''; ← always True
  - select * from users where login='' or '1'='1' # ' and password=''; ← always True

# Authentication Bypass

- Checking for SQLI vulnerability by injecting "'" payload in login text field:



# Lab: Free Article Submissions

Lab URL: https://attackdefense.com/challengedetails?cid=315

# Lab: OpenSupports

Lab URL: https://attackdefense.com/challengedetails?cid=437

# Homework Lab: Basic SQL Injection

Lab URL: https://attackdefense.com/challengedetails?cid=1901

# SQL Injection Types

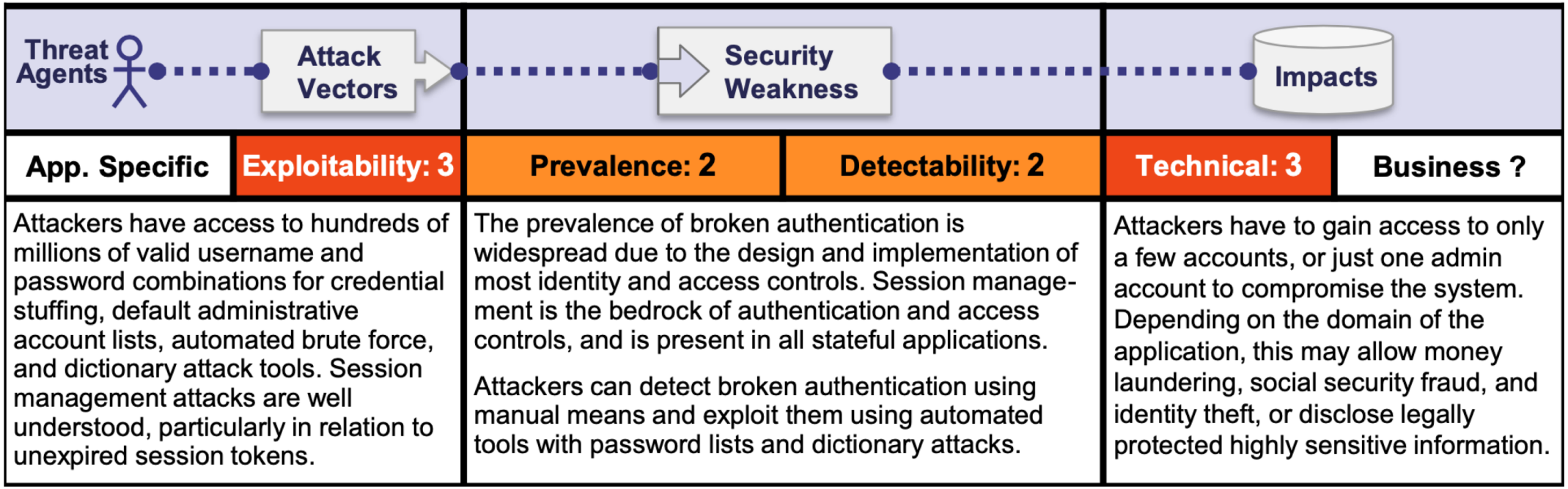
- Classic SQL Injection
  - Union Based SQL Injection
  - Error Based SQL Injection
- Blind SQL Injection
  - Boolean Based
  - Time Based
- Out of bound SQLI

# OWASP Top 10 : A2 Broken Authentication

**A2 :2017**

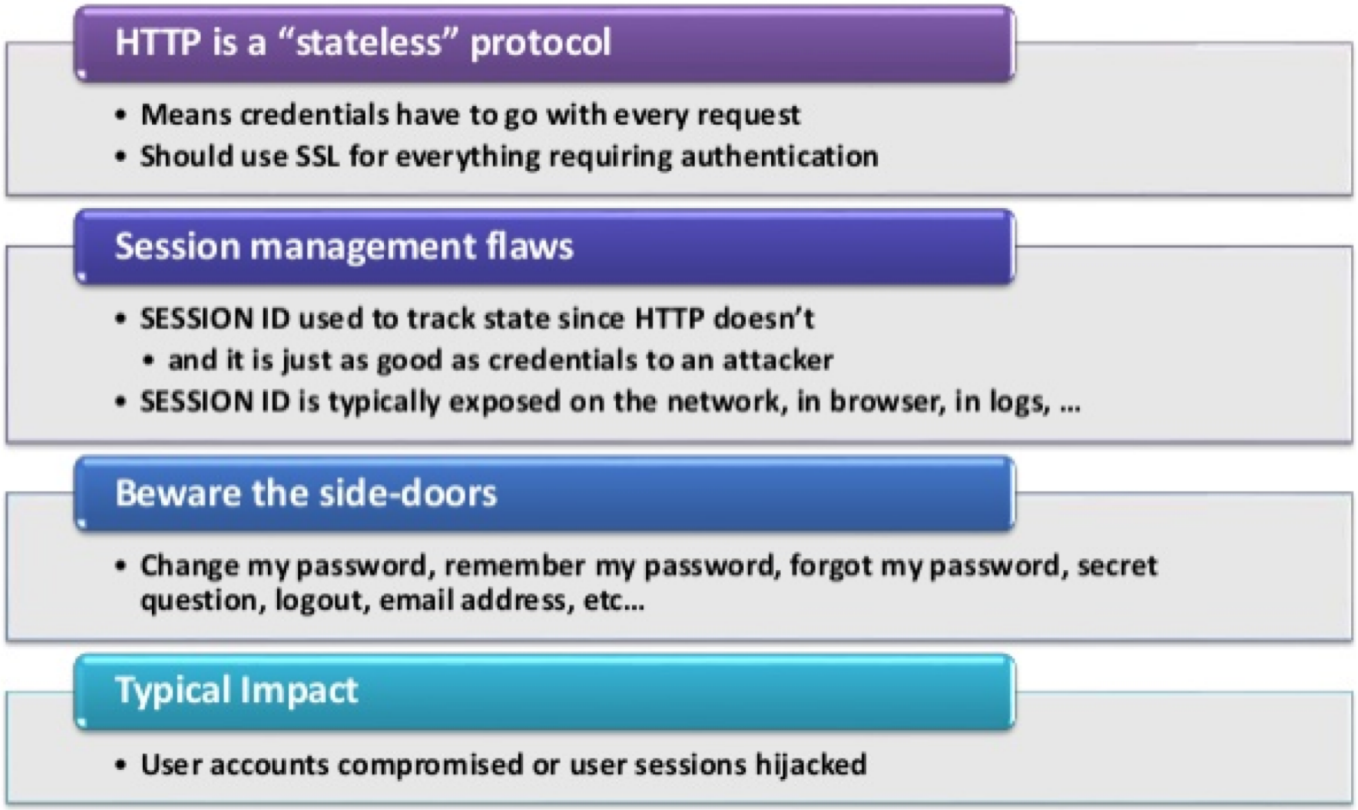
## Broken Authentication






Source: OWASP

# OWASP Top 10 : A2 Broken Authentication



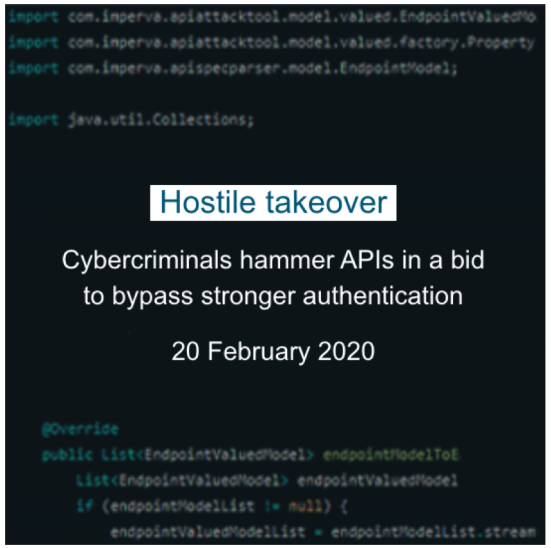

Source: OWASP

# OWASP Top 10 : A2 Broken Authentication

- Improper Authentication ranked 13th on Mitre CWE list

| Rank | ID | Name | Score |
|---|---|---|---|
| [1] | CWE-119 | Improper Restriction of Operations within the Bounds of a Memory Buffer | 75.56 |
| [2] | CWE-79 | Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting') | 45.69 |
| [3] | CWE-20 | Improper Input Validation | 43.61 |
| [4] | CWE-200 | Information Exposure | 32.12 |
| [5] | CWE-125 | Out-of-bounds Read | 26.53 |
| [6] | CWE-89 | Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection') | 24.54 |
| [7] | CWE-416 | Use After Free | 17.94 |
| [8] | CWE-190 | Integer Overflow or Wraparound | 17.35 |
| [9] | CWE-352 | Cross-Site Request Forgery (CSRF) | 15.54 |
| [10] | CWE-22 | Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal') | 14.10 |
| [11] | CWE-78 | Improper Neutralization of Special Elements used in an OS Command ('OS Command Injection') | 11.47 |
| [12] | CWE-787 | Out-of-bounds Write | 11.08 |
| [13] | CWE-287 | Improper Authentication | 10.78 |
| [14] | CWE-476 | NULL Pointer Dereference | 9.74 |

# OWASP Top 10 : A2 Broken Authentication




## Critical Authentication Bypass Vulnerability in InfiniteWP Client Plugin

**This entry was posted in** Vulnerabilities, WordPress Security **on January 14, 2020 by** Matt Barry 8 Replies


thehackernews.com › computer security › hacking

### Authentication Bypass Vulnerability Found in Auth0 Identity ...

Apr 7, 2018 - A critical **authentication bypass vulnerability** has been discovered in ... due to improper validation of the JSON **Web** Tokens (JWT) audience ...

# Terminologies

- Authentication
  - the process or action of verifying the identity of a user or process

- Authorization
  - security mechanism used to determine user/client privileges or access levels related to system resources

# When is the application vulnerable?

- Permits Automated attacks such as credential stuffing
- Permits brute force or other automated attacks
- Permits default, weak or well-known passwords
- Uses Weak or ineffective credential recovery and forgot-password processes
- Uses plain-text, encrypted or weakly hashed password
- Has missing or ineffective Multi-factor authentication

Source: OWASP

# When is the application vulnerable?

- Exposes Session ID in URL
- Does not rotate session id after successful login
- Does not properly invalidate session ID

# Installing Burp Suite and FoxyProxy

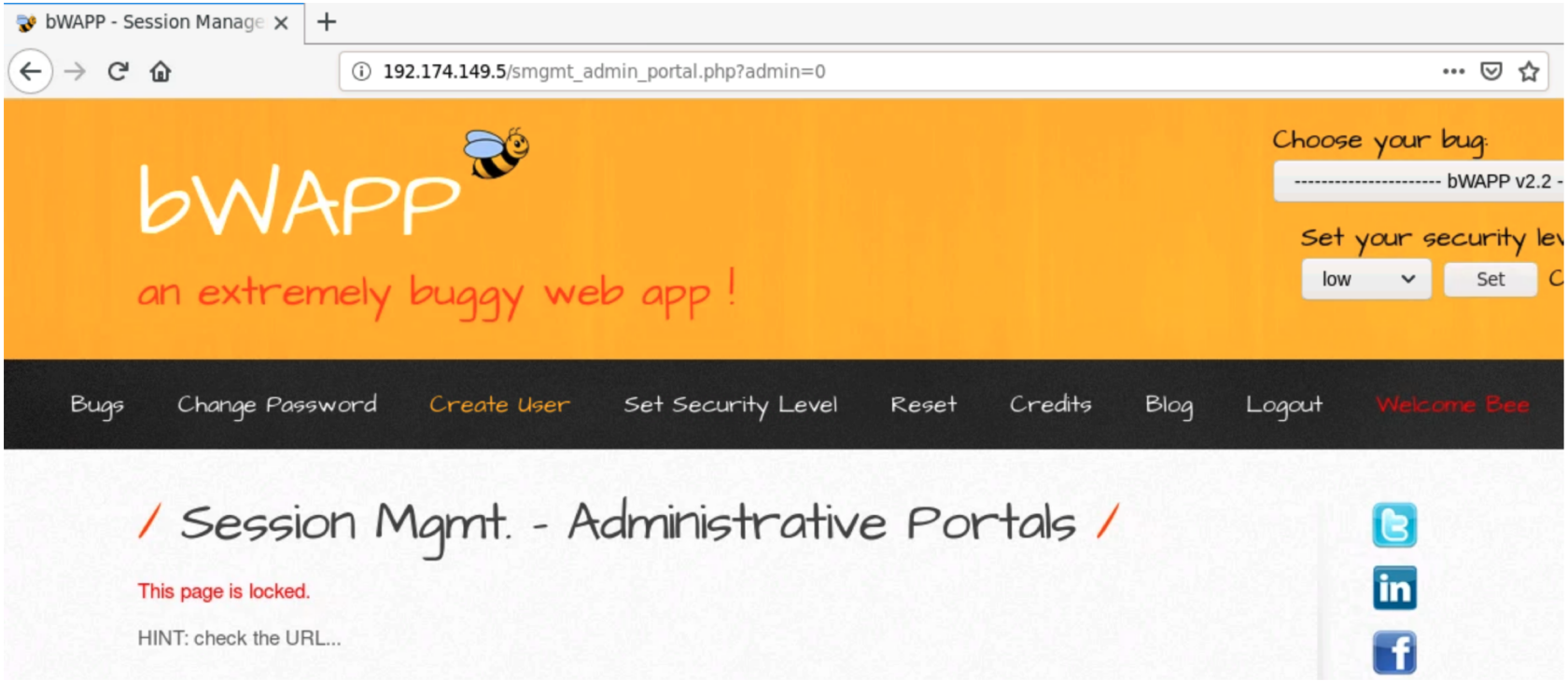
Video URL: https://youtu.be/-JLUw3fr-ro

# Lab: Online Airline Booking System

Lab URL: https://attackdefense.com/challengedetails?cid=438

# Improper Session Management

- bWAPP - Session Management - Administrative Portals
  - Admin session control parameter is URL

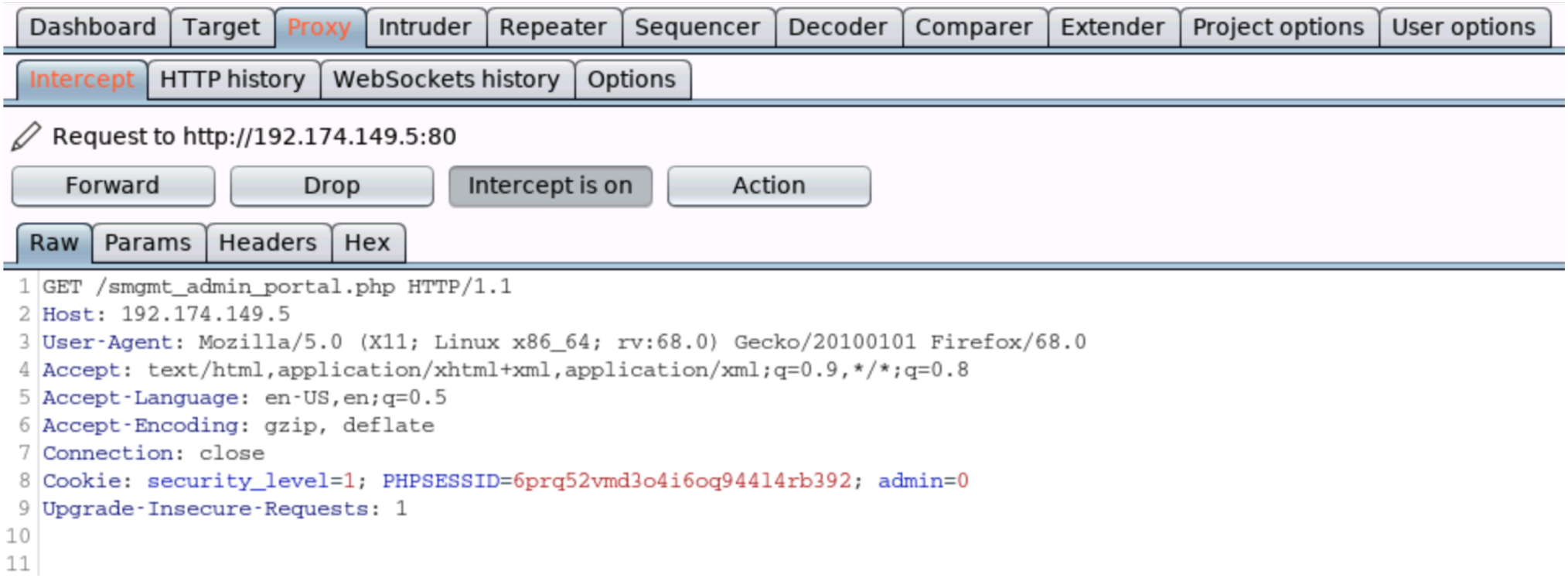


# Lab: Improper Session Management

Lab URL: https://attackdefense.com/challengedetails?cid=1899
Video: https://youtu.be/Yt-5Fgg-u_4

# Improper Session Management - Cookie based

- bWAPP - Session Management - Administrative Portals
  - Admin Cookie



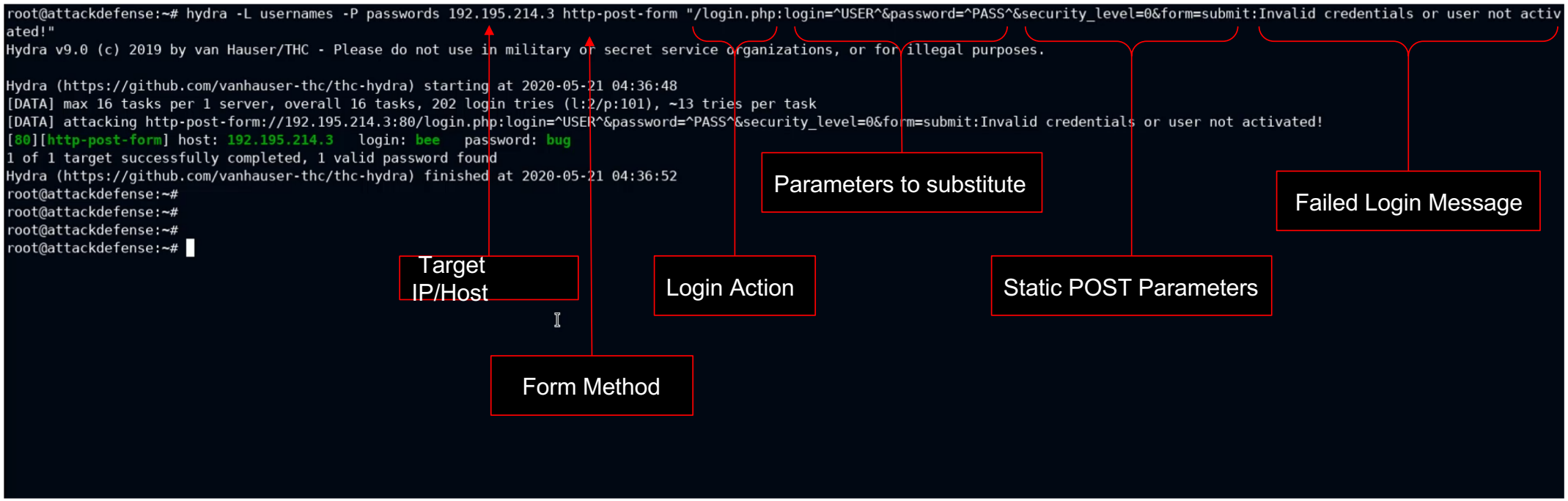
# Lab: Improper Session Management II

Lab URL: https://attackdefense.com/challengedetails?cid=1899
Video: https://youtu.be/BvrKOK9Z7ok

# Weak Password

- bWAPP - Broken Authentication - Password Attacks
- Default passwords / Easy to remember
- Might be present in popular wordlists
- Depending Upon length, password can be brute forced.

# HTTP Login Form : Hydra



# Lab: Attacking HTTP Login Form with Hydra

Lab URL: https://attackdefense.com/challengedetails?cid=1895

# Prevention

- Where possible, implement multi-factor authentication to prevent automated, credential stuffing and dictionary/brute force.
- Do not ship or deploy with any default credentials, particularly for admin users.
- Implement weak-password checks, such as testing new or changed passwords against a list of the top 10000 worst passwords
- Align password length, complexity and rotation policies with NIST 800-63 B's guidelines in section 5.1.1 for Memorized Secrets or other modern, evidence based password policies.

Source: OWASP

# Prevention

- Ensure registration, credential recovery, and API pathways are hardened against account enumeration attacks by using the same messages for all outcomes.
- Limit or increasingly delay failed login attempts. Log all failures and alert administrators when credential stuffing, brute force, or other attacks are detected.
- Use a server-side, secure, built-in session manager that generates a new random session ID with high entropy after login. Session IDs should not be in the URL, be securely stored and invalidated after logout, idle, and absolute timeouts.

Source: OWASP

# OWASP Top 10 : A3 Sensitive Data Exposure

- Information Exposure ranked 4th on Mitre CWE list

| Rank | ID | Name | Score |
|---|---|---|---|
| [1] | CWE-119 | Improper Restriction of Operations within the Bounds of a Memory Buffer | 75.56 |
| [2] | CWE-79 | Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting') | 45.69 |
| [3] | CWE-20 | Improper Input Validation | 43.61 |
| [4] | CWE-200 | Information Exposure | 32.12 |
| [5] | CWE-125 | Out-of-bounds Read | 26.53 |
| [6] | CWE-89 | Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection') | 24.54 |
| [7] | CWE-416 | Use After Free | 17.94 |
| [8] | CWE-190 | Integer Overflow or Wraparound | 17.35 |
| [9] | CWE-352 | Cross-Site Request Forgery (CSRF) | 15.54 |
| [10] | CWE-22 | Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal') | 14.10 |
| [11] | CWE-78 | Improper Neutralization of Special Elements used in an OS Command ('OS Command Injection') | 11.47 |
| [12] | CWE-787 | Out-of-bounds Write | 11.08 |
| [13] | CWE-287 | Improper Authentication | 10.78 |
| [14] | CWE-476 | NULL Pointer Dereference | 9.74 |

# OWASP Top 10 : A3 Sensitive Data Exposure


ATTACKS/BREACHES

3/13/2020
12:15 PM

## Princess Cruises Confirms Data Breach

**The cruise liner, forced to shut down operations due to coronavirus, says the incident may have compromised passengers' personal data.**

DARK Reading

Dark Reading Staff
Quick Hits

Carnival-owned Princess Cruises — the cruise line forced to suspend operations after two ships became hot spots for coronavirus — reports that a breach may have compromised passenger data.


threatpost.com › Breach

## Millions of Guests Impacted in Marriott Data Breach, Again ...

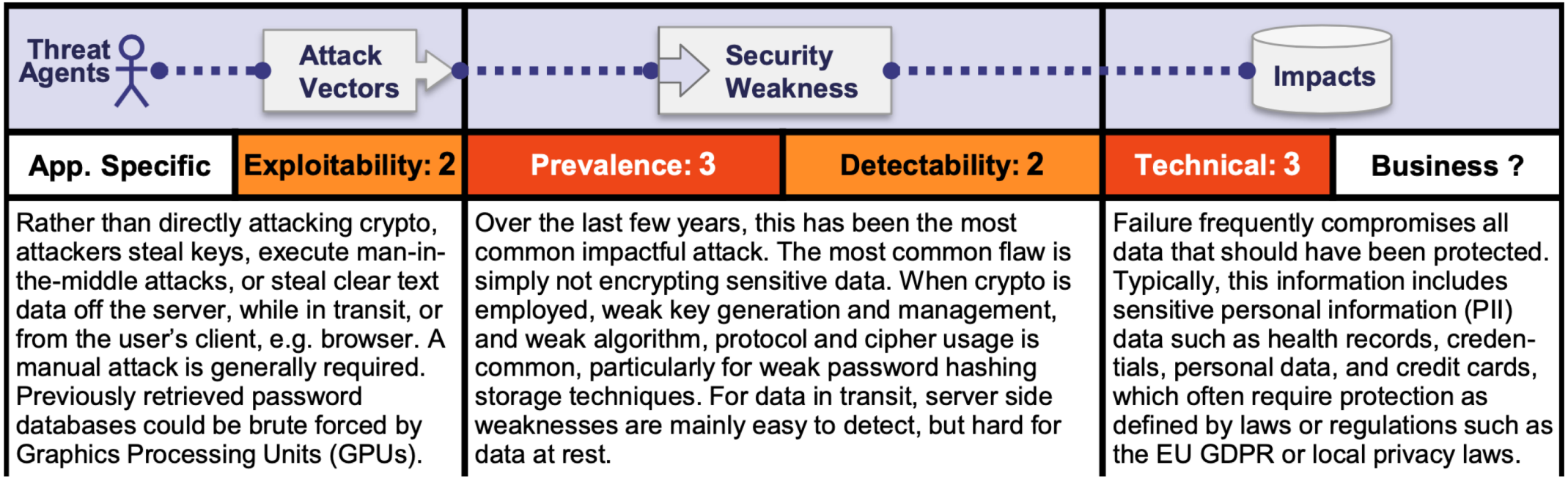
Mar 31, 2020 - In 2019, the Information Commissioner's Office (ICO), which is the U.K.'s privacy watchdog, hit **Marriott** with a $123 million (£99 million) penalty ...

# OWASP Top 10 : A3 Sensitive Data Exposure

**A3 :2017**

## Sensitive Data Exposure






Source: OWASP

# When is the application vulnerable?

- Is any data transmitted in clear text? This concerns protocols such as HTTP, SMTP, and FTP. External internet traffic is especially dangerous
- Is sensitive data stored in clear text, including backups?
- Are any old or weak cryptographic algorithms used either by default or in older code?
- Are default crypto keys in use, weak crypto keys generated or re-used, or is proper key management or rotation missing?

Source: OWASP

# When is the application vulnerable?

- Is encryption not enforced, e.g. are any user agent (browser) security directives or headers missing?
- Does the user agent (e.g. app, mail client) not verify if the received server certificate is valid?

Source: OWASP

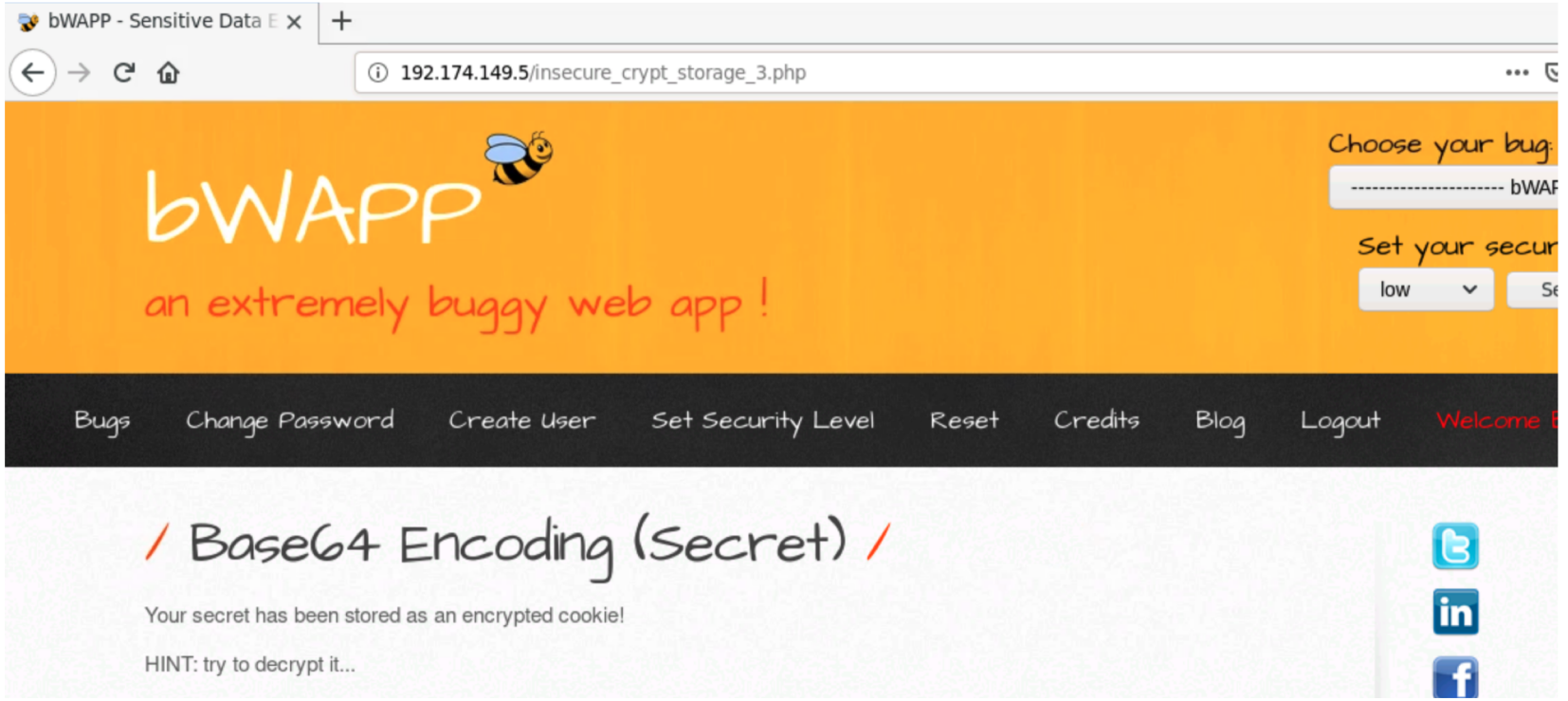
# Types of Storages

| | Cookies | Local Storage | Session Storage |
|---|---|---|---|
| **Capacity** | 4KB | 10MB | 5MB |
| **Browsers** | HTML4 / HTML 5 | HTML 5 | HTML 5 |
| **Accessible from** | Any window | Any window | Same tab |
| **Expires** | Manually set | Never | On tab close |
| **Storage Location** | Browser and server | Browser only | Browser only |
| **Sent with requests** | Yes | No | No |



# Encoded Cookie value

- bWAPP - Base64 Encoding (Secret)

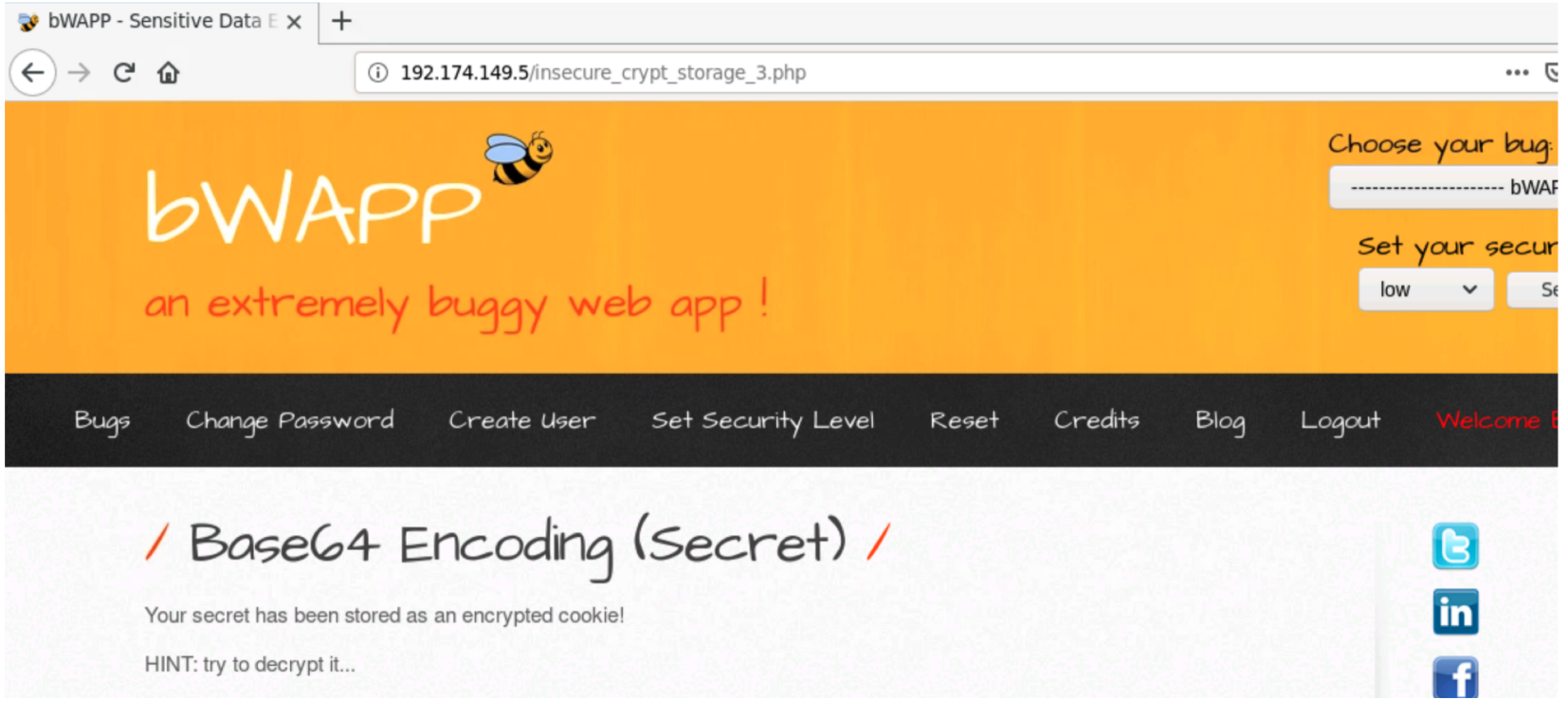


# Lab: Encoded Cookie Value

Lab URL: https://attackdefense.com/challengedetails?cid=1899
Video URL: https://youtu.be/NnweEYixzQg

# Data in Local Storage

- bWAPP - Base64 Encoding (Secret)



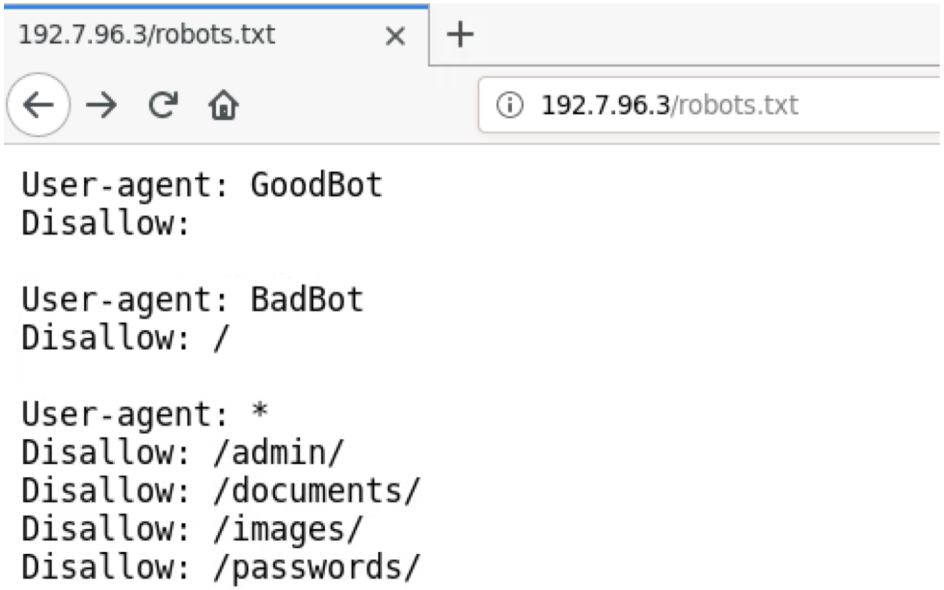
# Lab: Sensitive Data in Web Storage

Lab URL: https://attackdefense.com/challengedetails?cid=1899

Video URL: https://youtu.be/Y_6bbhD8x3o

# Robots.txt

- Tells Crawler allowed/disallowed pages/files/directories for crawling
- Used for Traffic Management
  - Prevents site from being overloaded by requests made from crawlers
- Preventing particular crawler from crawling pages.

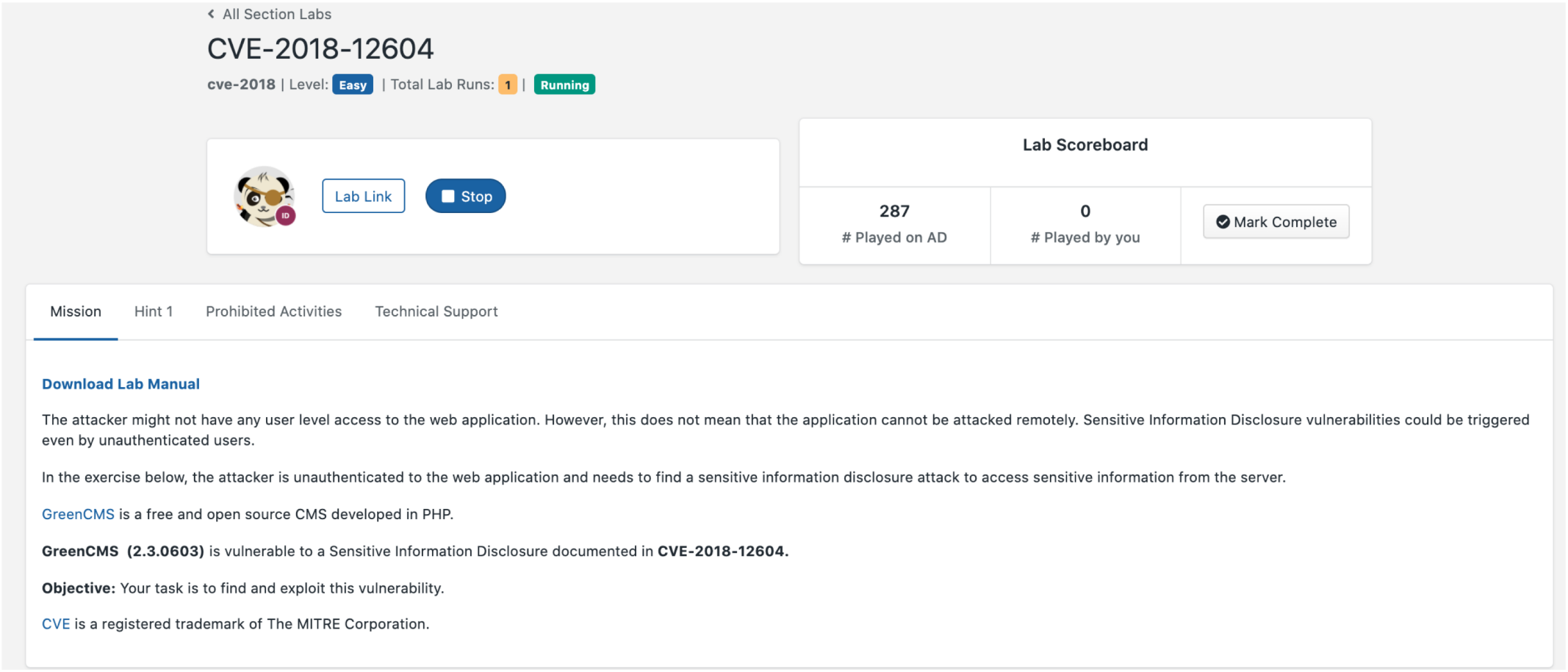


# Lab: Sensitive Directories in robots.txt

Lab URL: https://attackdefense.com/challengedetails?cid=1889

Video URL: https://youtu.be/7s5RD4OHD4w

# Real World Webapp




Lab Link: https://attackdefense.com/challengedetails?cid=11

# Lab: CVE-2018-12604

Lab URL: https://attackdefense.com/challengedetails?cid=11

# Prevention

- Classify data processed, stored, or transmitted by an application. Identify which data is sensitive according to privacy laws, regulatory requirements, or business needs
- Apply controls as per the classification
- Don’t store sensitive data unnecessarily. Discard it as soon as possible or use PCI DSS compliant tokenization or even truncation. Data that is not retained cannot be stolen.
- Make sure to encrypt all sensitive data at rest.

Source: OWASP