



SECURITY TESTING for QAs

Keep Your Company Safe and Have Fun Doing It

| AGENDA

What Will We See Today

01

MEET THE TEAM

Get To Know Us

02

SECURITY TESTING

What kind of security testing can you perform?

03

APPLICATION SECURITY

What is Application Security and why your applications should be secure?

04

WHY SHOULD YOU CARE?

Security Is cool and all, but why should you care?

05

HOW CAN YOU GET INVOLVED?

How can you start getting involved in security?

06

TOOLS

Open Source tools that make Security Testing easier.

07

OWASP TOP 10

What Is the OWASP TOP 10?

08

SECURITY MISCONFIGURATIONS AND HOW TO FIND THEM

Take a look at common security misconfigurations and learn how to find them.

09

CROSS SITE SCRIPTING AND TESTING FOR IT

What is Cross Site Scripting and how can test for it?

10

IDORS AND HOW TO SPOT THEM

What is an IDOR and learn an easy way to spot them.

11

ARBITRARY FILE UPLOADS AND EXPLOITING IT

Why file uploads matter and all the malicious things you can do with a bad file

12

RESOURCES

Helpful resources that can help you along your security journey

| MEET THE TEAM



AMELIO WRIGHT

Ethical Hacker

AWright2@Paylocity.com



KAYLA PATE

Penetration Tester

Kpate@Paylocity.com



MITHUN CHANDRA JAKKULA

Penetration Tester

MJakkula@Paylocity.com

| SECURITY TESTING

Black box testing: Performed without having knowledge of the software or programming. Internal workings of an application are not required to be known. Testing is based on external expectation, internal behavior of application is unknown. Least time consuming and exhaustive.

Gray box testing: Combination of white-box testing and black-box testing. Internal programming partially known. Some knowledge of internal working of application. Partly time consuming and exhaustive.

White box testing: Performed with full knowledge of software or programming. Tester has full knowledge of internal working of the application. Internal working are fully known and tester can design test data accordingly. Most exhaustive and time consuming.

| SECURITY TESTING

There are several different types of Security Testing that can be performed

Network: Network Security Testing involves looking for vulnerabilities in the network. Vulnerability Scans or penetration tests can help find vulnerabilities in a network.

Code: Testing for vulnerabilities in code can be done in many ways. If a penetration tester has insight into the code during a penetration test, they can try to exploit these vulnerabilities. Security code reviews can be done amongst team members. Static Scanning tools can help automate testing for vulnerabilities in the code.

Mobile: Since everyone does everything on a smartphone, it is important to test for vulnerabilities in a mobile application and making sure any confidential information is not being leaked

API: Of all the components that comprise an application, Application Programming Interfaces (APIs) provide the easiest access point for a hacker who wants your data.

Cloud: Cloud security is the protection of data, applications, and infrastructures involved in cloud computing

Web App: Web Apps contain a lot of important sensitive information such as credit card, health, social security, and other private information. There are many ways to look for vulnerabilities. Our talk will focus on Web Applications

APPLICATION SECURITY



“Application Security is the process of making apps more secure by finding, fixing, and enhancing the security of apps”
– David Strom, CSO Online

APPLICATION SECURITY: COMMON ATTACK GOALS AND MOTIVES

The following are common motives for Security attacks

Information: Bad actors look at application's to steal valuable information. Attacker's look for credentials so that they can gain access to user's information. They look to get information such as banking, credit card, health, social security, and other private information

Espionage: Attackers can disguise themselves in order to gain insider information for an outside competitor or enemy.

Sabotage: Disgruntled employees or angry fired employees may seek revenge on a company. With their insider information, they may be able to steal information or cause damage to a company. Another way to commit sabotage is through blackmailing a company such as a CEO.

| WHY SHOULD YOU CARE?



WHY SHOULD YOU CARE?

Your company might not experience that bad of a day if you don't practice good security, but here are some reasons you should care

Confidence: You can have confidence that you are building secure application's

Security Testing is Simple and Valuable for QAs to do: Since QAs are already used to testing, adding in some basic security tests should be fairly simple. QAs have an in depth knowledge of how application's work. This inside knowledge is very valuable when it comes to testing your own application and looking for vulnerabilities

Promote Security Awareness at your company: By incorporating security testing into your regular testing, you can help encourage other people and teams to do their own testing. By encouraging others you can spread awareness about application security and get others to help make your company secure as well.

Detect vulnerabilities: By testing for vulnerabilities in your application, you are ideally catching them before the bad guys do

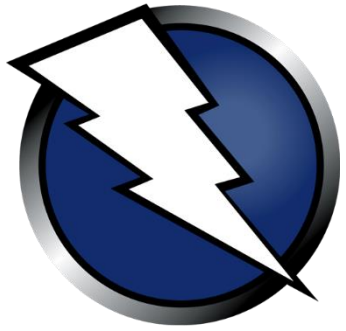
Vulnerability Prevention: Even better than detecting vulnerabilities, preventing them is even more valuable. By testing application's for security flaws before going to production, you can prevent releasing vulnerabilities a bad guy could find

Strengthen Organization: By making sure your application's are secure, you are strengthening the security of your organization. This helps keep your customer's safe and lower's your company's chance of being breached

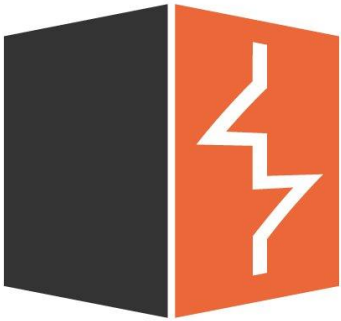
| HOW CAN YOU GET INVOLVED?

- Start small, but think big!
- Identify the top areas of security concerns in your application
- Locate Sensitive/ Confidential information in your application and make sure you are handling it properly
- Incorporate a couple basic security tests into every release
- Meet with your team regularly and discuss how you can make your application more secure
- Learn any chance you get!

| TOOLS



ZAP



BURP SUITE



FOXY PROXY



DEVELOPER
TOOLS



EDIT THIS
COOKIE



RETIRE JS

Links listed in slide notes

OWASP TOP 10

- 1) INJECTION
- 2) BROKEN AUTHENTICATION
- 3) SENSITIVE DATA EXPOSURE
- 4) XML EXTERNAL ENTITIES (XXE)
- 5) BROKEN ACCESS CONTROL
- 6) SECURITY MISCONFIGURATION
- 7) CROSS-SITE SCRIPTING (XSS)
- 8) INSECURE DESERIALIZATION
- 9) USING COMPONENTS WITH KNOWN VULNERABILITIES
- 10) INSUFFICIENT LOGGING AND MONITORING

https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project

OWASP TOP 10

#1 INJECTION

- **Vulnerability:** Injection flaws occur when untrusted data is sent as part of a command or query. SQL injection is most common, although other types do exist.
- **Risk:** 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.

#2 BROKEN AUTHENTICATION

- **Vulnerability:** The prevalence of broken authentication is widespread. Attackers can detect broken authentication using manual means and exploit them using automated tools with password lists and dictionary attacks.
- **Risk:** Attackers have to gain access to only a few accounts, or just one admin account to compromise the system. Depending on the domain of the application, this may allow money laundering, social security fraud, and identity theft, or disclose legally protected highly sensitive information.

OWASP TOP 10

#3 SENSITIVE DATA EXPOSURE

Vulnerability: Sensitive data is exposed by not properly encrypting sensitive data.

Risk: Sensitive data that's possible to be exposed includes, but is not limited to, sensitive personal information, health records, credentials, and credit cards.

#4 XML EXTERNAL ENTITIES (XXE)

Vulnerability: This is an attack against a web application that parses XML* input. This input can reference an external entity, attempting to exploit a vulnerability in the parser.

Risk: Attackers can also extract data, execute a remote request from the server, scan internal systems, perform a denial-of-service attack, as well as execute other attacks.

#5 BROKEN ACCESS CONTROL

Vulnerability: Broken access controls allow attackers to bypass authorization and perform tasks as though they were privileged users such as administrators

Risk: Attackers can act as administrators, access privileged functions, creating records, access records, update records, or deleting every record.

OWASP TOP 10

#6 SECURITY MISCONFIG

Vulnerability: Most common vulnerability on the list Caused by bad/insecure default configurations.

Risk: This vulnerability can give out information that user's should not have access to. It can also give unauthorized access to some system data or functionality. Occasionally, such flaws result in a complete system compromise.

#7 CROSS-SITE SCRIPTING (XSS)

Vulnerability: Cross-Site Scripting (XSS) is an injection attack where a malicious script is injected into a trusted websites.

Risk: Can have minor or major consequences which range from redirecting a user to a malicious site to taking over an account

| OWASP TOP 10

#8 INSECURE DESERIALIZATION

Vulnerability: An insecure deserialization exploit is the result of deserializing data from untrusted sources

Risk: The consequences range from harmless information disclosure to critical remote code execution.

#9 USING COMPONENTS WITH KNOWN VULNS

Vulnerability: Attacker's exploit vulnerabilities in outdated libraries

Risk: While some known vulnerabilities lead to only minor impacts, some of the largest breaches to date have relied on exploiting known vulnerabilities in components.

#10 INSUFFICIENT LOGGING AND MONITORING

Vulnerability: Many web applications are not taking enough steps to detect data breaches.

Risk: The average discovery time for a breach is around 200 days after it has happened. This gives attackers a lot of time to cause damage before there is any response.

| DISCLAIMER

PLEASE ALWAYS GET PERMISSION BEFORE STARTING A SECURITY TESTING ENGAGEMENT

ALL SECURITY TESTING SHOULD BE PERFORMED IN A TESTING ENVIRONMENT

SECURITY TESTING OF ANY KIND SHOULD NOT BE PERFORMED AGAINST A PRODUCTION OR LIVE ENVIRONMENT

| DEMO TIME

TIME TO DEMO!

| SECURITY MISCONFIGURATIONS

- Most commonly seen issue
- # 6 on the OWASP top 10 - https://www.owasp.org/index.php/Top_10-2017_A6-Security_Misconfiguration
- Caused by bad/insecure default configurations, incomplete or ad hoc configurations, misconfigured HTTP headers, or error messages containing sensitive information
- Make sure your applications are configured properly and components are regularly patched or upgraded!

Example Attack: The application server comes with sample applications that are not removed from the production server. These sample applications have known security flaws attackers use to compromise the server. If one of these applications is the admin console, and default accounts weren't changed the attacker logs in with default passwords and takes over

| SECURITY MISCONFIGURATIONS: INFORMATION DISCLOSURE

- This occurs when an application does not properly protect information from those that should not see that information.
- This issues is usually more informational and not exploitable in most cases.
- However, this could allow an attacker to find out information about an application that could later be used to carry out an attack

SECURITY MISCONFIGURATIONS: INFORMATION DISCLOSURE

Demo: How to spot information disclosure in the header of an application

Request	Response
Raw	HeadersHexHTMLRender
HTTP/1.1 200 OK	
Cache-Control: private	
Content-Type: text/html; charset=utf-8	
Server: Microsoft-IIS/8.5	
X-AspNetMvc-Version: 5.2	
X-AspNet-Version: 4.0.30319	
X-Powered-By: ASP.NET	
Date: Tue, 17 Apr 2018 20:06:02 GMT	
Connection: close	
Content-Length: 5662	

SECURITY MISCONFIGURATIONS: UNNECESSARY COMMENTS

- Unnecessary Comments in code can give a user or bad actor information that they should not have.
- These comments could be personal information, code secrets, bad coding practices, etc.
- Make sure developers are not leaving comments in their code that they should not be

SECURITY MISCONFIGURATIONS: UNNECESSARY COMMENTS

Demo: Easy ways to spot unnecessary comments in the code of an application

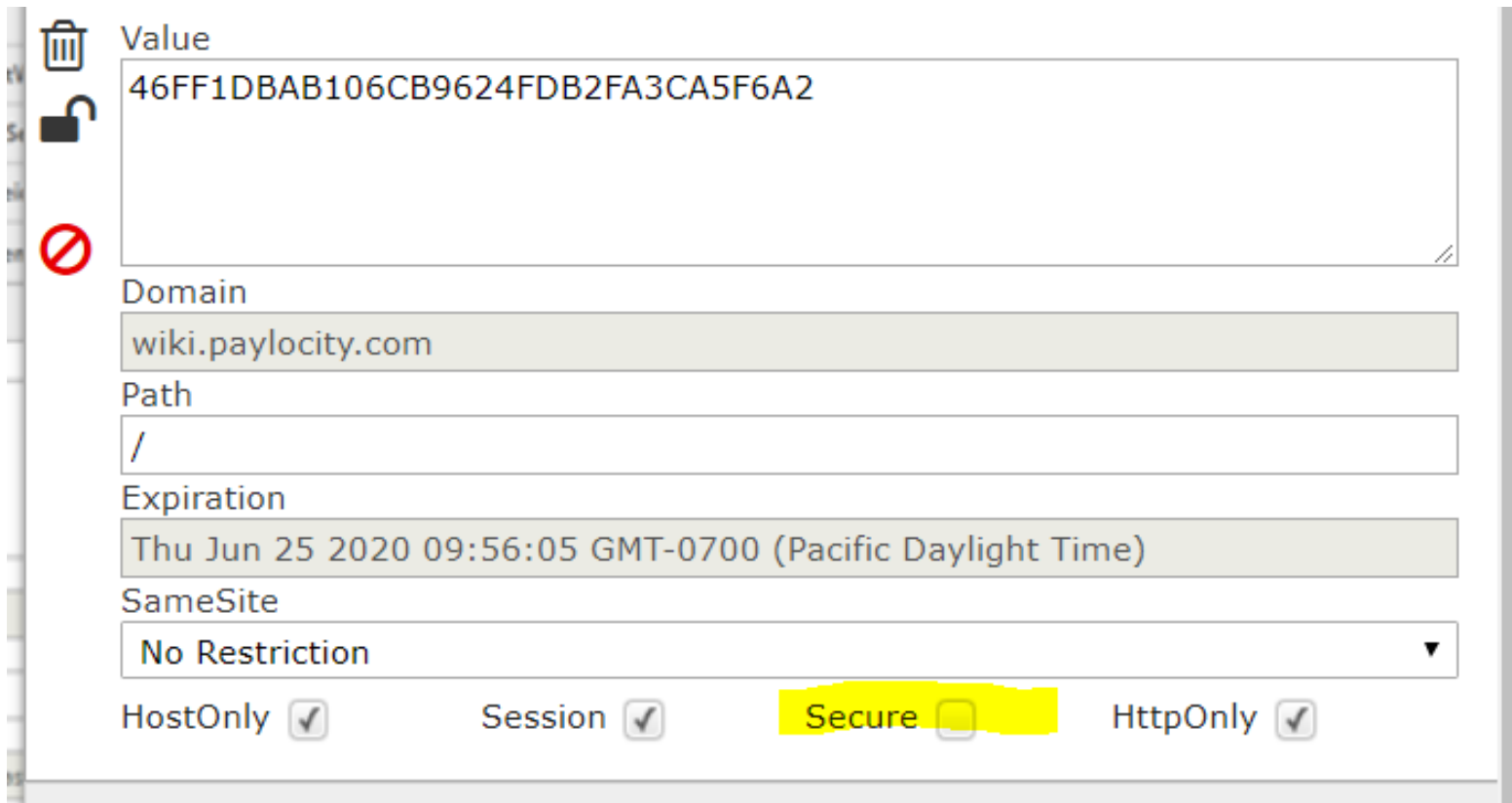
```
<h2>Demographics</h2>
<ul class="nav nav-tabs success">
  <li class="active"><a href="/NinjaChallenge/Employee/DemographicsRoot">Demographics</a></li>
  <li><a href="/NinjaChallenge/Employee/Edit">Edit</a></li>
  <li><a href="/NinjaChallenge/Employee/Paychecks">Paychecks</a></li>
  <li><a href="/NinjaChallenge/Employee/Media">Media</a></li>
  <li><a href="/NinjaChallenge/Employee/UploadPhoto">Upload Photo</a></li>
</ul>
<div>
  <hr />
  <!-- SSN 319-78-2538 -->
  <dl class="dl-horizontal">
    <dt>
```

| SECURITY MISCONFIGURATIONS: SECURITY FLAGS NOT SET

- **Secure flag** is to prevent cookies from being observed by unauthorized parties due to the transmission of a the cookie in clear text
- **HttpOnly flag** is used when generating a cookie in order to help mitigate the risk of cross site scripting attack to access the protected cookie
- Make sure your security flags are set

SECURITY MISCONFIGURATIONS: SECURITY FLAGS NOT SET

Demo: Use Edit This Cookie to make sure security flags are set in an application



The screenshot displays the 'Edit This Cookie' interface with the following fields and values:

- Value:** 46FF1DBAB106CB9624FDB2FA3CA5F6A2
- Domain:** wiki.paylocity.com
- Path:** /
- Expiration:** Thu Jun 25 2020 09:56:05 GMT-0700 (Pacific Daylight Time)
- SameSite:** No Restriction
- HostOnly:** ☒
- Session:** ☒
- Secure:** ☐ (highlighted in yellow)
- HttpOnly:** ☒

| SECURITY MISCONFIGURATIONS: SESSION FIXATION

- This allows an attacker to hijack a valid user session.
- This attack is executed by getting a user to authenticate themselves, and then hijacking the user-validated session by the knowledge of the used session ID.
- Make sure your applications are giving out new session ids to users and that the session ids are random

SECURITY MISCONFIGURATIONS: SESSION FIXATION

Demo: Using Edit This Cookie to see if the application is using a new and random Session ID

Value
46FF1DBAB106CB9624FDB2FA3CA5F6A2

Domain
wiki.paylocity.com

Path
/

Expiration
Thu Jun 25 2020 09:56:05 GMT-0700 (Pacific Daylight Time)

SameSite
No Restriction

HostOnly ☒ Session ☒ Secure ☐ HttpOnly ☒

CROSS SITE SCRIPTING (XSS)

- # 7 on OWASP top 10 -
[https://www.owasp.org/index.php/Top_10-2017_A7-Cross-Site_Scripting_\(XSS\)](https://www.owasp.org/index.php/Top_10-2017_A7-Cross-Site_Scripting_(XSS))
- Cross-Site Scripting (XSS) is an injection attack where a malicious script is injected into a trusted websites.
- These attacks are common and is caused by application not validating input and/or output
- Make sure your application's are validating input and output, encoding input and output

CROSS SITE SCRIPTING (XSS): Examples

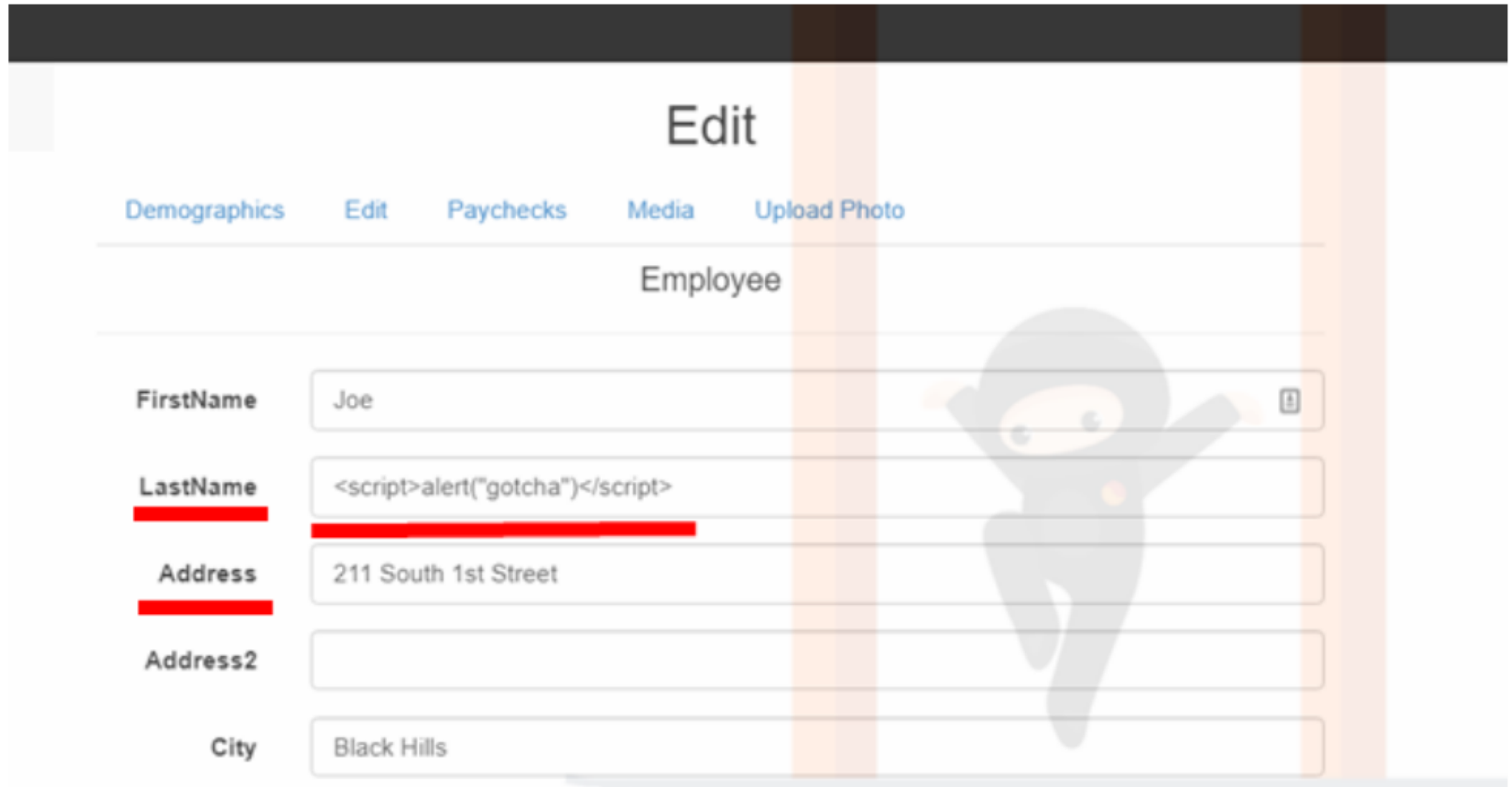
- XSS vulnerability allows an attacker to modify a press release or news item could affect a company's stock price or lessen consumer confidence.
- XSS vulnerability on a pharmaceutical site could allow an attacker to modify dosage information resulting in a patient taking the wrong or incorrect amounts of medicine

| CROSS SITE SCRIPTING (XSS): TYPES

- **Reflected/ Non Persistent XSS** – The most common type of XSS. The malicious script has to be a part of the request that is sent to the web server. It is then reflected back so that the HTTP response includes the payload from the HTTP. Attacker's use malicious links, phishing attacks, and other social engineering attacks to execute this
- **Stored/ Persistent XSS** - The most damaging type of XSS. An attacker stores a script into an application. A common place to store a script is in a blog, so that when a user opens the page, the script executes
- **DOM** – Least common and more advanced XSS attack

CROSS SITE SCRIPTING (XSS): TYPES

Demo: Simple Tests to Check for XSS in your application

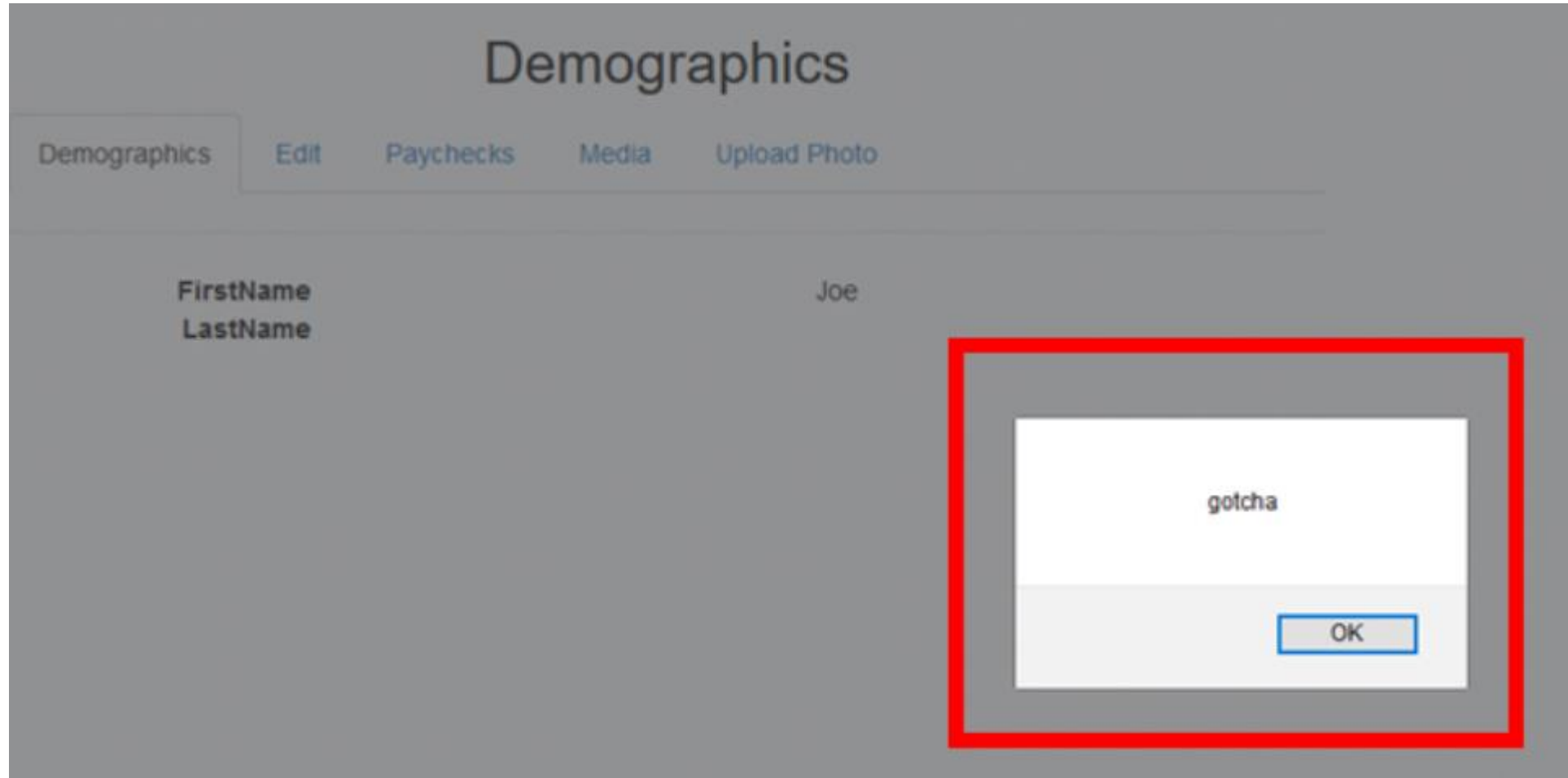


The screenshot shows a web application interface for editing an employee record. The title 'Edit' is centered at the top. Below it are navigation tabs: 'Demographics', 'Edit', 'Paychecks', 'Media', and 'Upload Photo'. The 'Edit' tab is selected. The form is titled 'Employee' and contains several input fields. The 'LastName' field is highlighted with a red underline and contains the malicious payload `<script>alert("gotcha")</script>`. The 'Address' field is also highlighted with a red underline and contains the text '211 South 1st Street'. The 'City' field contains 'Black Hills'. A faint watermark of a cartoon ninja is visible in the background of the form.

Edit	
Demographics Edit Paychecks Media Upload Photo	
Employee	
FirstName	Joe
<u>LastName</u>	<code><script>alert("gotcha")</script></code>
<u>Address</u>	211 South 1st Street
Address2	
City	Black Hills

CROSS SITE SCRIPTING (XSS): TYPES

Demo: Simple Tests to Check for XSS in your application



| INSECURE DIRECT OBJECT REFERENCE (IDOR)

- #5 on OWASP Top 10 - https://www.owasp.org/index.php/Top_10-2017_A5-Broken_Access_Control
- Applications frequently use the actual name or key of an object when generating web pages and applications don't always verify the user is authorized for the target object. Testers can easily manipulate parameter values to detect this vulnerability
- Example: I have <https://website.com/userprofile=123> and I want to access another user's profile. I can change <https://website.com/userprofile=123> to <https://website.com/userprofile=124> and access another's user's information
- Make sure user's are given correct permissions and use a guid or encrypt URL parameters instead of using easily guessable parameters

| UNRESTRICTED FILE UPLOAD

- This vulnerability takes advantage of application's that do not perform proper checks on file uploads. An attacker's goal with this vulnerability is to upload malicious files
- The consequences of unrestricted file upload can vary ranging from site defacement to complete site take down. It depends on what the application does with the uploaded file and especially where it is stored.
- Make sure you are analyzing uploaded files, only allowing required file formats to be uploaded, and perform client and server side checks on the upload.

RESOURCES: VULNERABLE WEB APPS



ATTACK DEFENSE



BWAPP



JUICE SHOP



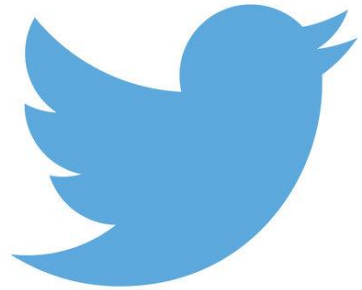
HACK THE BOX



CLOUD GOAT

LINKS IN SLIDE NOTES

RESOURCES: SECURITY NEWS



WIRED

SANS
INSTITUTE

DARKReading



OWASP

Open Web Application
Security Project

RESOURCES: CHICAGO SECURITY CONFERENCES

