Hands-On Ethical Hacking and Network Defense, Edition 4

Chapter 10: Hacking Web Servers

Module Objectives

- By the end of this module, you should be able to:
  - Describe web applications
  - Explain web application vulnerabilities
  - Describe the tools used to attack web servers

Understanding Web Applications

- Writing a program without bugs is difficult
  - Some defects create security vulnerabilities
  - The bigger the program, the more bugs or defects are possible
  - The more people who have access to a program, the bigger the risk of security vulnerabilities

Web Applications Components

- **Static webpages**
  - Created using HTML
  - Display the same information regardless of the time of day or the user who accesses the page
- **Dynamic webpages**
  - Information that is displayed varies
  - Need special components for displaying information that changes depending on user input or information from a back-end server
  - Use a variety of tools
    - `<form>` element, Asynchronous JavaScript and XML (AJAX), Common Gateway Interface (CGI), Active Server Pages (ASP.NET), Java Server Pages (JSP), Hypertext Preprocessor (PHP), ColdFusion (CF), JavaScript (JS), and database connector strings

Web Forms (1 of 3)

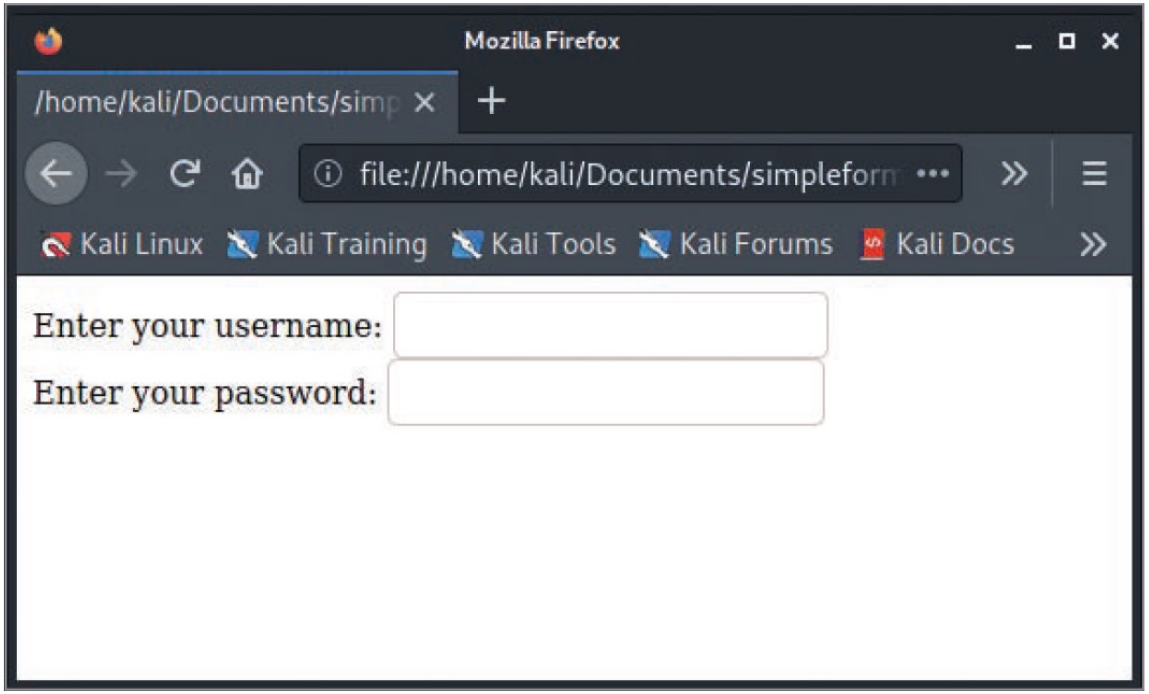
- Use the `<form>` element in an HTML document
  - To allow customers to submit information to the web server
- Some forms can be quite long and ask for a lot of information
  - Some have only a couple of input fields
- Web servers
  - Use a web application to process information from a form

Web Forms (2 of 3)

- Web form example:

```
<html>
<body>
<form>
Enter your username:
<input type="text" name="username">
<br>
Enter your password:
<input name="password" type="password">
</form></body></html>
```

Web Forms (3 of 3)



Source: Mozilla Firefox

Figure 10-1 HTML webpage with a form

Common Gateway Interface (CGI) (1 of 2)

- Handles moving data from a web server to a web browser
- Enables web designers to create dynamic HTML web applications
- Many dynamic webpages are created with CGI and scripting languages
- Interface that determines how a web server passes data to a web browser
  - Relies on Perl or another scripting or programming language to create dynamic webpages
- Main role
  - Passing data between a web server and web browser

Common Gateway Interface (CGI) (2 of 2)

- CGI programs can be written in many programming and scripting languages
  - C/C++, Perl, UNIX shells, Visual Basic, and Java
- CGI example displaying “Hello Security Testers” written in Perl:

```
#!/usr/bin/perl
print "Content-type: text/html\n\n";
print "Hello Security Testers!";
```

Third-Party Frameworks and Libraries

- A few of the hundreds of frameworks designed to make programming easier:
  - Spring
  - JSF
  - AngularJS
  - Yeoman
  - Sass
  - Vaadin
- As third-party libraries grow in popularity, keeping them current and secure becomes more important

Active Server Pages (1 of 2)

- **ASP and ASP.NET**
  - Two other technologies that developers can use to display HTML documents to users on the fly
- Has been succeeded by ASP.NET
  - Uses a compiled server-side language (such as C#) and the .NET framework
- Enabled developers to build dynamic, interactive webpages using scripting languages
  - JScript
  - VBScript
- Not all web servers support ASP or ASP.NET

Active Server Pages (2 of 2)

- ASP example:

```
<html>
<head><title>My First ASP.NET Webpage</title></head>
<body>
<h1>Hello, security professionals</h1>
The date and time is <%=DateTime.Now %>.
</body>
</html>
```

- Keep in mind that the web server, not the web browser, must support ASP

Apache Web Server

- Apache
  - Another web server program
  - 2021: Apache Web Server has 31.7% of the web server market share compared to 6.7% for IIS
- Advantages
  - Works in just about any *nix and Windows platform
  - Free
  - Apache Web Server daemon (httpd) is included by default in Kali Linux

Using Scripting Languages

- Web pages
  - Developed using several scripting languages
    - VBScript
    - JavaScript
- Many security-testing tools are written with scripting languages
- Most macro viruses and all worms that take advantage of cross-site scripting vulnerabilities are based on a scripting language

PHP Hypertext Processor (PHP) (1 of 2)

- Enables creation of dynamic webpages
  - Similar to ASP and ASP.NET
- An open-source server-side scripting language
  - Embedded in an HTML webpage by using the PHP tags `<?php` and `?>`
- Users cannot see PHP code on their web browser
  - Because PHP webpages run on the server
- Originally used mainly on UNIX systems
  - More widely used now on many platforms, including Macintosh and Windows

PHP Hypertext Processor (PHP) (2 of 2)

- Code example for a static PHP webpage showing the use of PHP tags:

```
<html>
<head>
<title>My First PHP Program </title>
</head>
<body>
<?php echo '<h1>Hello, Security Testers!</h1>"; ?>
</body>
</html>
```

Cold Fusion (1 of 2)

- Server-side scripting language
  - Used to develop dynamic webpages
  - Created by Allaire Corporation
    - Now owned by Adobe Systems, Inc.
- Uses proprietary tags
  - Written in ColdFusion Markup Language (CFML)
- CFML web applications
  - Can contain other client-side technologies, such as HTML and JavaScript

Cold Fusion (2 of 2)

- CFML example:

```
<html>
<head>
<title>Using CFML</title>
</head>
<body>
<CFLOCATION URL="www.isecom.org" ADDTOKEN="NO">
</body>
</html>
```

JavaScript (1 of 4)

- Popular scripting language used for creating dynamic webpages
- Has the power of a programming language
  - Branching
  - Looping
  - Testing
- Creates functions and procedures in HTML webpages
- Widely used
- Variety of vulnerabilities have been exploited in older web browsers
  - Security testers and administrators should inspect every computer for unpatched or outdated browser versions
    - Keep up with vulnerabilities

JavaScript (2 of 4)

- HTML snippet with JavaScript code:

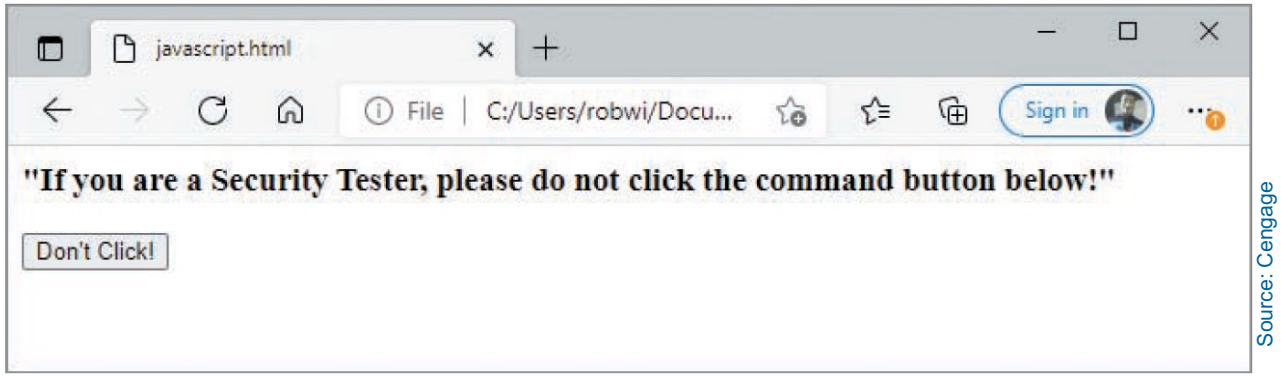
```
<html>
<head>
<script type="text/javascript">
function chastise_user()
{
alert("So, you like breaking rules?")
document.getElementByld("cmdButton").focus()
}
</script>
</head>
<body>
```

JavaScript (3 of 4)

- JavaScript example (continued):

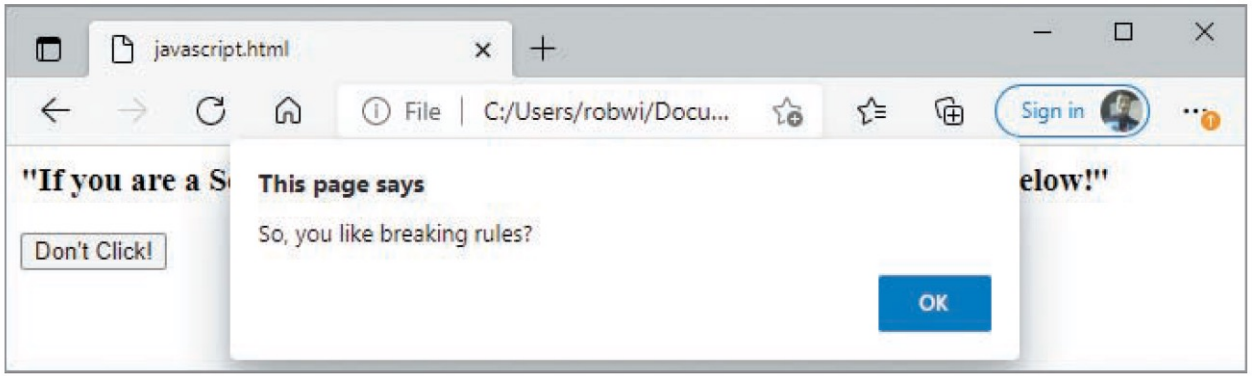
```
<h3>"If you are a Security Tester, please do not click the command
  button below!"</h3>
<form>
<input type="button" value="Don't Click!" name="cmdButton"
onClick="chastise_user()" />
</form>
</body>
</html>
```

JavaScript (4 of 4)



Source: Cengage

Figure 10-6 A command button created with JavaScript



Source: Cengage

Figure 10-7 An alert message created with JavaScript

Connecting to Databases

- Most webpages that display company information to users are stored on a database server
- The technology used to connect a web application to a database server might vary depending on the OS
  - But the theory is the same

Open Database Connectivity

- **Open Database Connectivity (ODBC)**
  - A standard database access method developed by the SQL Access Group
- ODBC interface
  - Allows an application to access data stored in a database management system (DBMS), or any system that can recognize and issue ODBC commands
- Interoperability is accomplished by defining the following:
  - Standardized representation for data types
  - Library of ODBC function calls that allow an application to connect to a DBMS, run SQL statements, and retrieve the results
  - Standard method of connecting to and logging onto a DBMS

Object Linking and Embedding Database (OLE DB)

- **OLE DB**
  - Set of interfaces that enable applications to access data stored in a DBMS
- Designed by Microsoft
  - Faster, more efficient, and more stable than ODBC
- Relies on connection strings that allow the application to access data stored on an external device
- Different providers can be used
  - Depends on the data source

OLE DB Providers (1 of 2)

OLE DB provider	Description in connection string
Microsoft Active Directory Service	`Provider=ADSDSOOBJECT`
Advantage	`Provider=Advantage OLE DB Provider`
AS/400 (from IBM)	`Provider=IBMDA400`
AS/400 and VSAM (from Microsoft)	`Provider=SNAOLEDB`
MS Commerce Server	`Provider=Commerce.DSO.1`
DB2	`Provider=DB2OLEDB`
Microsoft Jet	`Provider=Microsoft.Jet.OLEDB.4.0`
Microsoft.ACE	`Provider=Microsoft.ACE.OLEDB.12.0 MS Exchange`

OLE DB Providers (2 of 2)

OLE DB provider	Description in connection string
MySQL	`Provider=MySQLProv`
Oracle (from Microsoft)	`Provider=msdaora`
Oracle (from Oracle)	`Provider=OraOLEDB.Oracle MS SQL`
MS SQL Server	`Provider=SQLOLEDB`

ActiveX Data Objects

- **ActiveX Data Objects (A D O)**
  - A programming interface for connecting a web application to a database
  - Defines a set of technologies that allow applications to interact with the web
- Steps for accessing a database from an ASP webpage:
  - Create an A D O connection to the database you want to access
  - Open the database connection you created in Step 1
  - Create an A D O recordset
  - Open the recordset
  - Select the data you need from the recordset, based on particular criteria
  - Close recordset and database connection

Knowledge Check Activity 10-1

Which of the following can be used to create dynamic webpages? (Choose all that apply.)

a. ColdFusion
b. PHP
c. ASP
d. MySQL

Knowledge Check Activity 10-1: Answer

Which of the following can be used to create dynamic webpages?

Answer: a., b., and c. ColdFusion, PHP, and ASP

Dynamic webpages can be created with a variety of techniques, including CGI, ASP.NET, ASP, PHP, ColdFusion, and JavaScript.

Polling Activity 10-1

Which of the following is an open source technology for creating dynamic HTML webpages?

a. ASP
b. PHP
c. Java
d. Oracle

Polling Activity 10-1: Answer

Which of the following is an open source technology for creating dynamic HTML webpages?

Answer: b. PHP

PHP Hypertext Processor (PHP) enables web developers to create dynamic webpages.

Discussion Activity 10-1

CGI is used in Microsoft ASP pages. True or false?

Also, discuss various web application components with your classmates.

Discussion Activity 10-1: Answer

CGI is used in Microsoft ASP pages. True or false?

Answer: False

Explanation: Microsoft uses ASP, which enabled developers to build dynamic, interactive webpages using scripting languages, such as JScript (Microsoft's version of JavaScript) or VBScript. It did not use CGI.

Understanding Web Application Vulnerabilities

- Many platforms and programming languages can be used to design a website
- Regardless of the platform, security professionals need to assess the system and examine potential methods for attacking it
- Application security
  - As important as network security
  - Referred to as AppSec
  - Was once overlooked by professionals because it is specialized practice
- Attackers controlling a web server can:
  - Deface the website
  - Destroy the application's database or sell its contents
  - Gain control of user accounts
  - Perform secondary attacks from the web server
  - Gain access to other servers that are part of the network infrastructure

Application Vulnerabilities and Countermeasures (1 of 6)

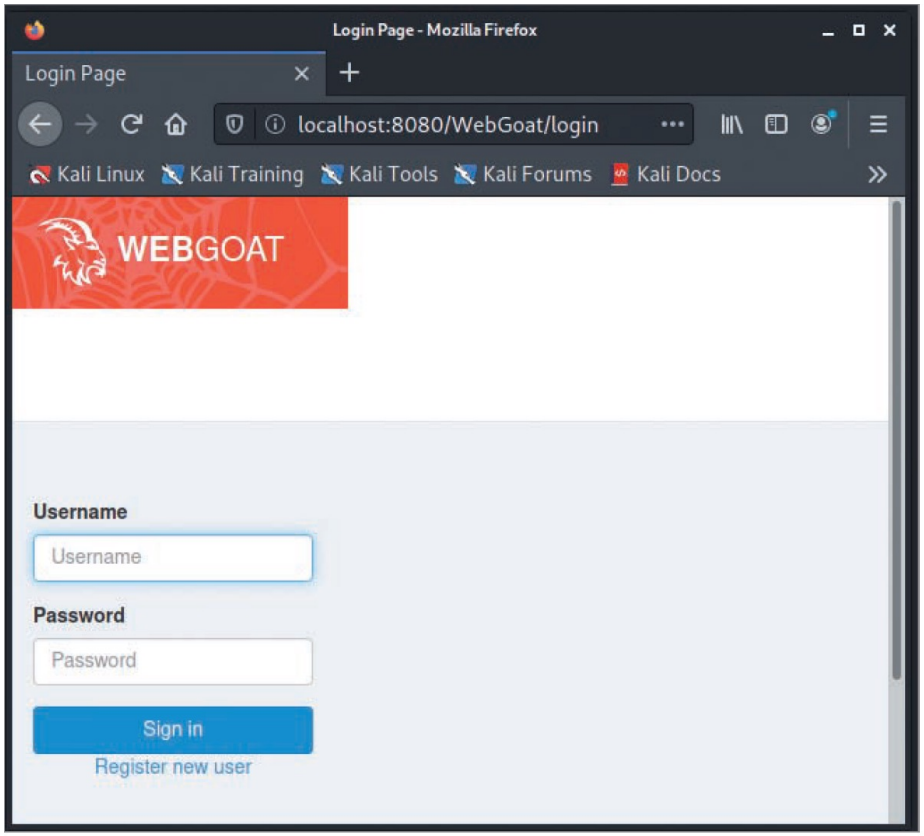
- **Open Web Application Security Project (OWASP)**
  - Not-for-profit organization
  - Finds and fights the causes of web application vulnerabilities
  - Publishes the “Ten Most Critical Web Application Security Risks” paper
    - Built into the Payment Card Industry (PCI) Data Security Standard (DSS)
- A security tester might need to analyze vulnerabilities such as the following in the OWASP Top 10 list:
  - Injection vulnerabilities
  - Authentication flaws and weaknesses
  - Sensitive data exposure
  - XML external entities (XXE)
  - Broken access control
  - Security misconfigurations

Application Vulnerabilities and Countermeasures (2 of 6)

- Cross-site scripting (XSS)
- Insecure deserialization
- Using components with known vulnerabilities
- Insufficient logging and monitoring

- OWASP
  - Offers Broken Web Apps and **WebGoat**
    - Webgoat: An online utility that helps beginning security testers understand the web application vulnerabilities covered in this list
- WebGoat project
  - Helps security testers learn how to conduct vulnerability testing on web applications
  - Experts from all over the world use WebGoat and offer their input
  - The following slides contain images of WebGoat

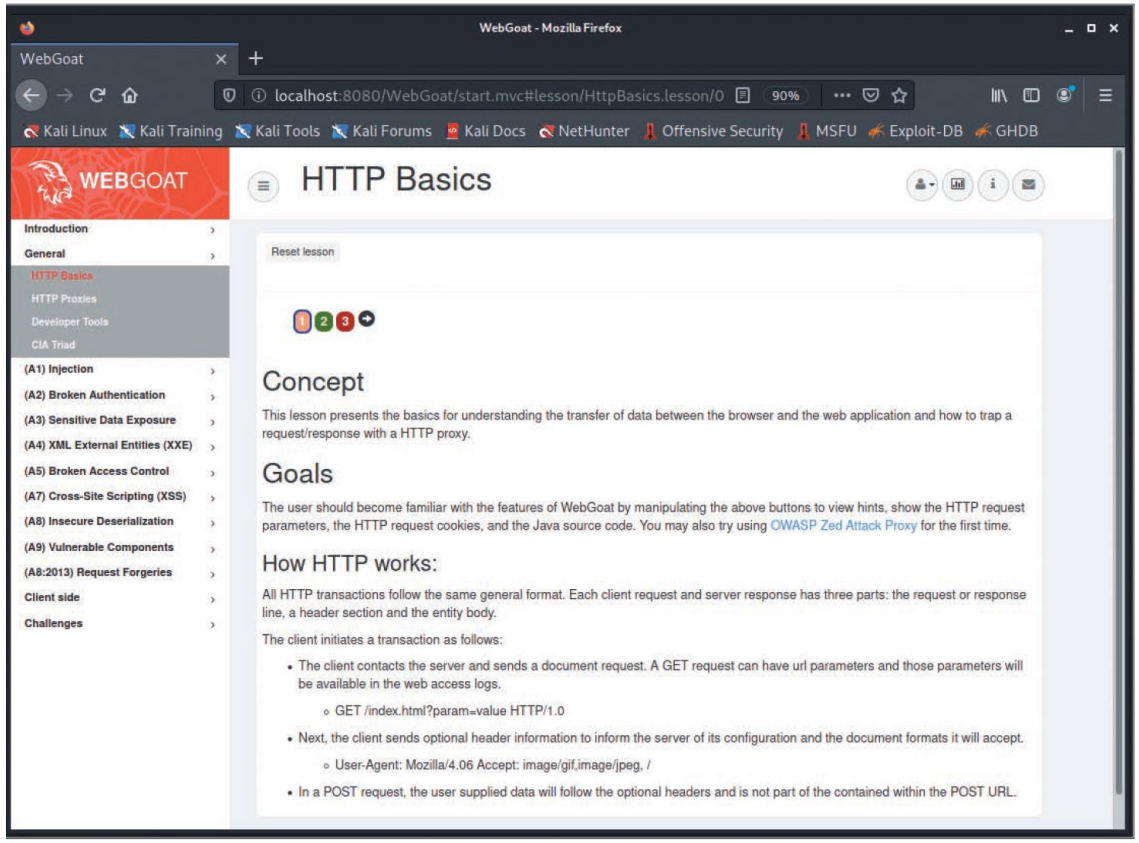
Application Vulnerabilities and Countermeasures (3 of 6)



Source: OWASP Foundation

Figure 10-8 WebGoat start page

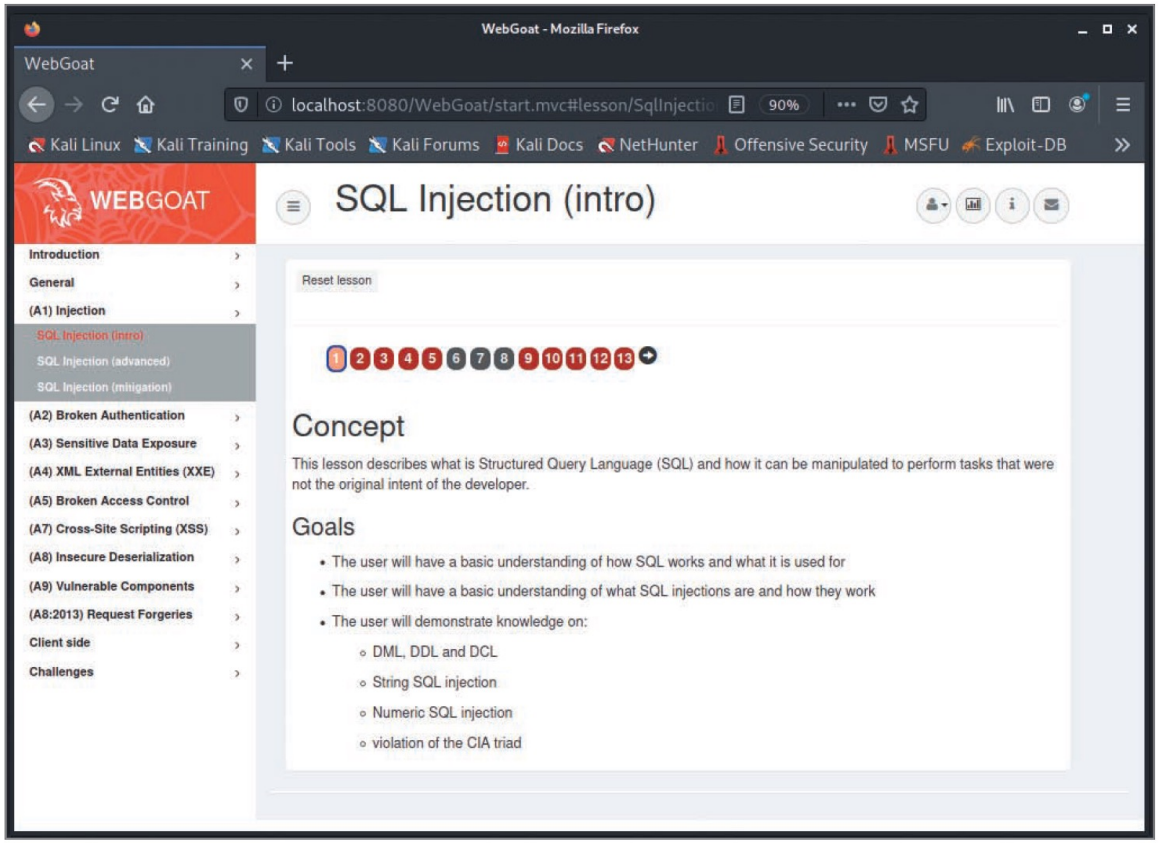
Application Vulnerabilities and Countermeasures (4 of 6)



Source: OWASP Foundation

Figure 10-10 WebGoat HTTP Basics exercise

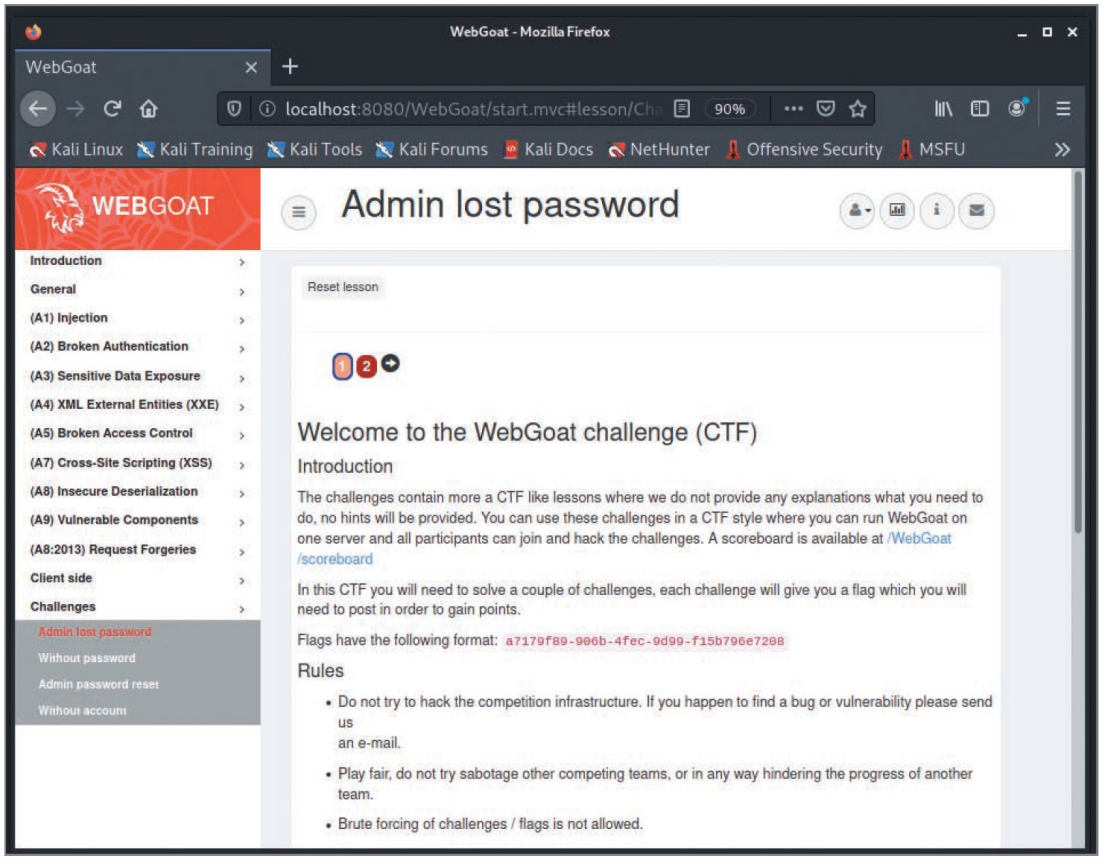
Application Vulnerabilities and Countermeasures (5 of 6)



Source: OWASP Foundation

Figure 10-11 WebGoat SQL Injection (intro) exercise

Application Vulnerabilities and Countermeasures (6 of 6)



Source: OWASP Org

Figure 10-12 WebGoat Challenges page

Web Application Test Execution

- Two techniques by which an application can be tested:
  - **Static Application Security Testing (SAST)**
    - Analyzing an application's source code for vulnerabilities
      - Only possible when the source code is available
    - A reliable way to enumerate most application vulnerabilities
  - **Dynamic Application Security Testing (D A S T)**
    - Analysis of a running application for vulnerabilities
    - Can be used alongside SAST to prioritize SAST findings
- Other technique
  - **Interactive Application Security Testing (IAST)**
    - Combines elements of both SAST and D A S T
    - Uses an agent inside the application to perform its analysis in real-time at any point in the development process

Information Gathering and Architecture Mapping

- Security testers should look for answers to some important questions:
  - Does the application have a database?
  - Does the application require authentication?
  - Does the application have static or dynamic pages?
  - What languages and platform does the application use?
  - Are there devices between your web browser and the application designed to stop attacks from occurring?
  - How does data flow in the application?

Platform Security and Configuration

- Several different platforms and technologies can be used to develop web applications
  - Attacks differ depending on platform and technology
    - Footprinting is used to discover the OS and DBMS that the attacked system is using
  - The more you know about a system, the easier it is to gather information about vulnerabilities and common misconfigurations
- Questions to consider during this phase:
  - Do the underlying platforms and components contain known vulnerabilities?
  - Is the web server configured to protect the confidentiality of users who connect to it?
  - Are there administrative interfaces to the infrastructure components and the applications being tested?

Authentication and Session Testing

- Many web applications require that a server other than the web server authenticate users
  - Examine how information is passed between the two servers
  - Is an encrypted channel used or is data passed in cleartext?
  - Is the server used for authentication properly configured and patched?
  - Are logon and password information stored in a secured location?

Authorization Testing

- Authorization
  - The act of checking a user's privileges to allow or deny access to a page, field, resource, or action in an application
- Application developers
  - Commonly use hidden fields in tables and obscured URLs to enforce their access control instead of checking users' privileges
- Authorization testing can reveal major areas of concern
  - An important part of any application test

Input Validation (1 of 3)

- Input validation
  - The act of filtering, rejecting, or sanitizing a user's untrusted input before the application processes it
- Input validation problems can lead to
  - Data disclosure
  - Alteration
  - Destruction
- Security testers should check for possibility of SQL injection used to attack the system
  - **SQL injection (SQLi):** Attacker supplies SQL commands when prompted to fill in a web application field

Input Validation (2 of 3)

- SQL injection example:

```
SELECT * FROM customer
WHERE tblusername = ' OR 1=1 - - AND tblpassword = ' '
```

- Because 1 equals 1 is always true, the query is carried out successfully
  - Double hyphens (--) are used in SQL to indicate a comment

Input Validation (3 of 3)

- Security testers should test any web applications when performing a security test and are authorized in writing to do so
- Basic testing should look for:
  - Whether you can enter text containing punctuation marks of any kind
  - Whether you can enter a single quotation mark followed by any SQL keywords
  - Whether you can get any sort of database error when attempting to inject SQL statements
- Sometimes, a web application will give a tester no indication that a SQL statement was run
  - OWASP calls this “Blind SQL injection” and it has its own set of tests that are required for detection

Error Handling

- A web application can be configured or written to handle errors in a variety of ways
  - Developers can enable debugging
    - Provides rich logging information helpful to diagnose issues
    - If the debugging mode is left on, it can provide a rich source of information for attackers
- Developers should minimize the amount of information shared with users when an application encounters an error
  - No information or only a generic message should be displayed to users in these error cases

Cryptography Testing

- Many problems in cryptography are due to simple things:
  - Bad random number generators
  - A known weak method of encryption
  - An encryption algorithm with known flaws that allow it to be cracked
  - An application that doesn’t actually enforce the use of secure channels
  - A self-signed certificate instead of a purchased certificate

Business Logic Testing

- Business logic
  - Refers to the procedure a user is expected to follow in an application to accomplish a goal
- Example:
  - Before a wire transfer, a user must first satisfy the requirement of having at least that amount of money in the transferring account
  - If the user doesn't have adequate funds, the transfer should be halted
- Business logic testing
  - Involves using creative ways to bypass these types of checks

Client-Side Testing

- Client-side issues
  - Arise from code executing on the user's machine, typically within the web browser
- Client-side controls
  - Insufficient on their own and should be paired with server-side controls that cannot be bypassed
- Key questions to ask with a client-side test are as follows:
  - Does the application store sensitive information on the client's machine in an insecure manner?
  - Does the application allow for client browser redirection if the server is fed a specially crafted request?

Polling Activity 10-2

What is D A S T?

a. Dynamic Application Static Testing
b. Dynamic Application Server Takeover
c. Delivery Application Server Testing
d. Dynamic Application Security Testing

Polling Activity 10-2: Answer

What is D A S T?

Answer: d. Dynamic Application Security Testing

D A S T stands for Dynamic Application Security Testing.

Knowledge Check Activity 10-2

Entering the value `' OR 1 = 1` in a web application that has an “Enter Your PIN” field is most likely an example of which attack?

a. SQL injection
b. Code injection
c. Buffer overflow
d. Ethernet flaw

Knowledge Check Activity 10-2: Answer

Entering the value `' OR 1 = 1` in a web application that has an “Enter Your PIN” field is most likely an example of which attack?

Answer: a. SQL injection

In SQL injection (SQLi), the attackers insert (“inject”) their own SQL statements within the “Enter Your PIN” field in a web application. Because 1 = 1 is always true, the query is carried out successfully.

Tools for Web Attackers and Security Testers

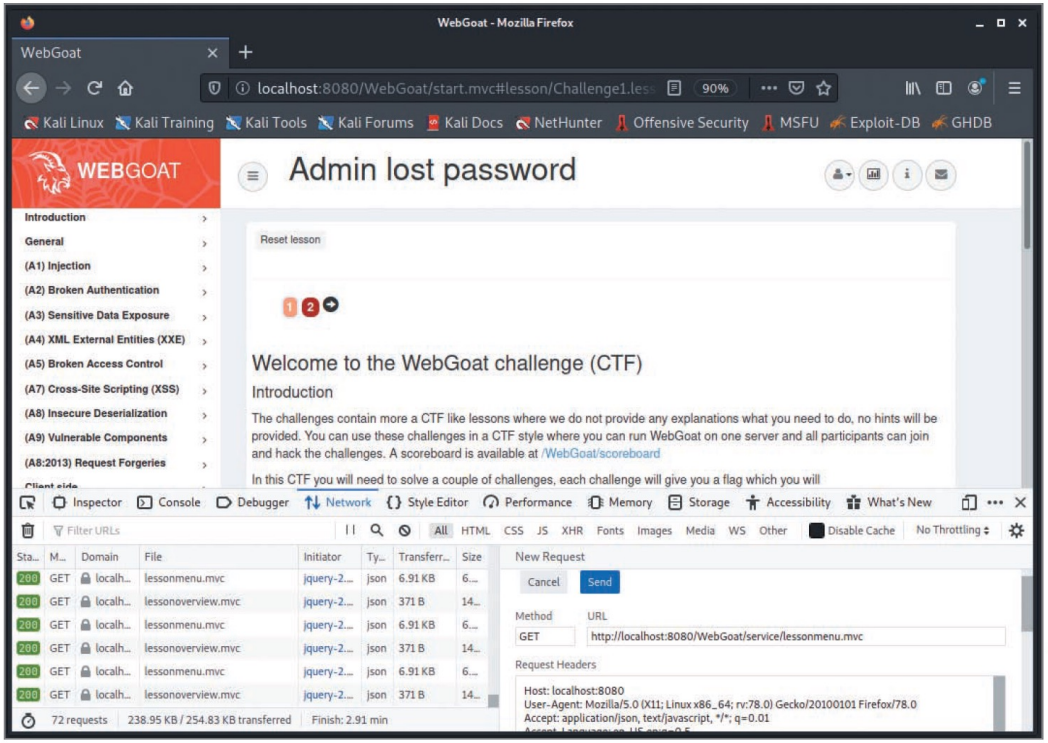
- After vulnerabilities of a web application or an OS platform are discovered, security testers or attackers look for tools to test or attack the system
  - All platforms and web application components have vulnerabilities
  - No matter which platform is used to develop a web application, there is a security hole that attackers can exploit to break into the system

Web Tools

- Most tools for performing a security test or attacking a network can be found on the Internet and are usually free
- Kali Linux
  - Is packed with free tools for hacking web applications
- You can install new tools with a simple `apt-get install packagename` command
- Other tools might be more suitable for a specific task

Firefox and Chrome Built-In Developer Tools

- Both come with similar set of developer tools
  - Useful for an application security tester
- Allow an attacker to:
  - View parameters in requests
  - Examine cookies
  - Tamper with and resend requests

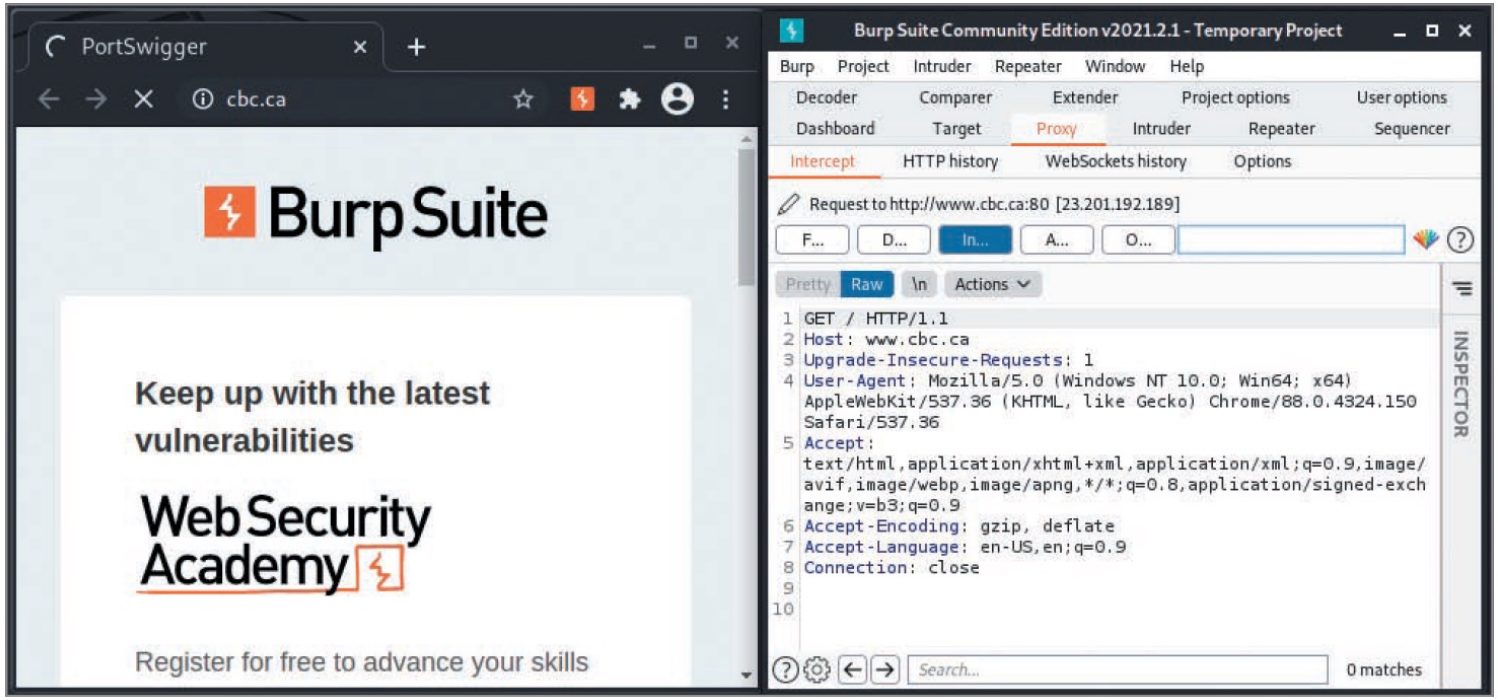


Source: Mozilla

Burp Suite and Zed Attack Proxy (1 of 2)

- Burp Suite
  - Included in Kali Linux
  - Offers the tester a number of features for testing web applications and web services
  - Allows you to intercept traffic between the web browser and the server so that you can inspect and manipulate requests before sending it to the server
  - Crawl, scan, and use brute force on applications
- Zed Attack Proxy
  - Can be used interchangeably with Burp Suite

Burp Suite and Zed Attack Proxy (2 of 2)



Source: Burp Suite

Figure 10-14 Burp Suite intercepting proxy

Wapiti

- Web application vulnerability scanner
- Uses a black box approach
  - Doesn’t inspect code
  - Inspects a website by searching from the outside
    - Ways to take advantage of XSS, SQL, PHP, JSP, and file-handling vulnerabilities
  - Uses “fuzzing”
    - Trying to inject data into whatever will accept it

Self-Assessment

Recall the differences between Common Gateway Interface (CGI) and Active Server Pages (ASP).

Describe the different types of scripting languages used to develop webpages.

Summary

- Now that the lesson has ended, you should be able to:
  - Describe web applications
  - Explain web application vulnerabilities
  - Describe the tools used to attack web servers