

Cybersecurity Audit Report

1. Scope and Purpose of the Audit

Objective:

The purpose of this audit is to identify security vulnerabilities in the WebGoat 8.1.0 web application and assess its overall security. WebGoat is a deliberately vulnerable platform designed for cybersecurity training.

Scope:

- Application Address: <http://127.0.0.1:8080/WebGoat>
- Targeted vulnerabilities:
 - A3: SQL Injection (Section 11)
 - A3: Cross-Site Scripting (Section 7)
 - A5: Security Misconfiguration (Section 4)
 - A6: Vulnerable and Outdated Components (Section 5)
 - A7: Identity and Authentication Failures (Secure Passwords, Section 4)

Tools Used:

- Docker (for deploying the environment)
- SQLMap
- Nmap
- Burp Suite

2. Executive Summary

2.1. Overview of the Process

- Deployed the WebGoat application using Docker.
- Conducted reconnaissance to gather critical information about the application's infrastructure and components.
- Exploited identified vulnerabilities aligned with OWASP Top 10 standards.

- Documented findings and proposed actionable recommendations to mitigate the risks.

2.2. Key Findings

- **SQL Injection:** Exploited error-based vulnerabilities to extract database details.
- **Cross-Site Scripting (XSS):** Injected malicious scripts to demonstrate user session compromise.
- **Security Misconfiguration:** Detected server misconfigurations revealing sensitive details.
- **Outdated Components:** Identified outdated libraries within the application stack.
- **Authentication Weaknesses:** Found weak password enforcement policies that enabled brute-force attacks.

```
alain@alain: ~  
File Actions Edit View Help  
(alain@alain)-[~]  
$ nmap webgoat.com  
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-12-01 14:51 CET  
Nmap scan report for webgoat.com (15.197.148.33)  
Host is up (0.0047s latency).  
Other addresses for webgoat.com (not scanned): 3.33.130.190  
rDNS record for 15.197.148.33: a2aa9ff50de748dbe.awsglobalaccelerator.com  
Not shown: 998 filtered tcp ports (no-response)  
PORT      STATE SERVICE  
80/tcp    open  http  
443/tcp    open  https  
Nmap done: 1 IP address (1 host up) scanned in 4.77 seconds
```

Goat without using WebWolf. Lessons where you can use WebWolf, are marked with the t

```
(alain@alain)-[~]  
$
```

s present, you are not obliged to use WebWolf. You can also use any intercepting tool you like. (netcat, etc.)

open WebWolf by clicking the icon in the top right corner.

n a new browser tab and is a separate web application that simulates an attacker's machine. It makes it possible for us to distinguish between what takes place on the at

as for WebWolf came about after a couple of workshops where we received feedback that there was no clear distinction between what was part of the "attackers" role and

wing functionality:

lie

email

age for incoming requests

```
(alain@alain)-[~]  
$ nmap -O webgoat.com  
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-12-01 14:52 CET  
Nmap scan report for webgoat.com (15.197.148.33)  
Host is up (0.0078s latency).  
Other addresses for webgoat.com (not scanned): 3.33.130.190  
rDNS record for 15.197.148.33: a2aa9ff50de748dbe.awsglobalaccelerator.com  
Not shown: 998 filtered tcp ports (no-response)  
PORT      STATE SERVICE  
80/tcp    open  http  
443/tcp    open  https  
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port  
Device type: bridge|general purpose You can also use any intercepting tool you like. ( netcat, etc.)  
Running (JUST GUESSING): Oracle Virtualbox (97%), QEMU (93%)  
OS CPE: cpe:/o:oracle:virtualbox cpe:/a:qemu:qemu  
Aggressive OS guesses: Oracle Virtualbox (97%), QEMU user mode network gateway (93%)  
No exact OS matches for host (test conditions non-ideal).  
a for WebWolf came about after a couple of workshops where we received feedback that there was no clear distinction between what  
OS detection performed. Please report any incorrect results at https://nmap.org/submit/ .  
Nmap done: 1 IP address (1 host up) scanned in 8.88 seconds
```



webgoat.com

Technology stack

Hosting



GoDaddy

```
(alain@alain)-[~] localhost:~$ curl -I webgoat.com
HTTP/1.1 405 Method Not Allowed
Server: openresty
Date: Sun, 01 Dec 2024 13:58:09 GMT
Connection: keep-alive
```

```
(alain@alain)-[~] $ nmap --script=http-enum -p 80 webgoat.com
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-12-01 14:59 CET
Nmap scan report for webgoat.com (3.33.130.190)
Host is up (0.0034s latency).
Other addresses for webgoat.com (not scanned): 15.197.148.33
rDNS record for 3.33.130.190: a2aa9ff50de748dbe.awsglobalaccelerator.com
PORT      STATE SERVICE
80/tcp    open  http
Nmap done: 1 IP address (1 host up) scanned in 8.13 seconds
```

Home > Whois Lookup > WebGoat.com

Whois Record for WebGoat.com

Domain Profile

Registrar	GoDaddy.com, LLC IANA ID: 146 URL: https://www.godaddy.com,http://www.godaddy.com Whois Server: whois.godaddy.com abuse@godaddy.com (p) +1.480.624.2505
Registrar Status	clientDeleteProhibited, clientRenewProhibited, clientTransferProhibited, clientUpdateProhibited
Dates	8,392 days old Created on 2001-12-10 Expires on 2024-12-10 Updated on 2023-12-11
Name Servers	NS41.DOMAINCONTROL.COM (has 58,992,386 domains) NS42.DOMAINCONTROL.COM (has 58,992,386 domains)
IP Address	3.33.130.190 - 27,611,318 other sites hosted on this server
IP Location	 - New Jersey - Princeton - Amazon Technologies Inc.
ASN	 AS16509 AMAZON-02, US (registered May 04, 2000)
Domain Status	Registered And No Website
IP History	35 changes on 35 unique IP addresses over 20 years
Hosting History	16 changes on 11 unique name servers over 21 years

Whois Record (last updated on 2024-12-01)

Domain Name:	WEBGOAT.COM
Registry Domain ID:	80986420 DOMAIN.COM-VRSN
Registrar WHOIS Server:	whois.godaddy.com
Registrar URL:	https://www.godaddy.com
Updated Date:	2023-12-11T12:33:36Z
Creation Date:	2001-12-10T18:17:55Z
Registrar Registration Expiration Date:	2024-12-10T18:17:55Z

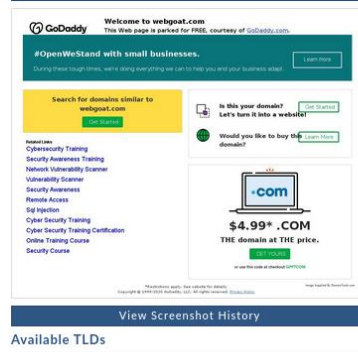
How does this work?



Preview the Full Domain Report

Tools

- Hosting History
- Monitor Domain Properties
- Reverse IP Address Lookup
- Network Tools
- Visit Website



Available TLDs

- General TLDs
- Country TLDs

Registrar Registration Expiration Date:	2024-12-10T18:17:55Z
Registrar:	GoDaddy.com, LLC
Registrar IANA ID:	146
Registrar Abuse Contact Email:	abuse@godaddy.com
Registrar Abuse Contact Phone:	+1.480.624.2505
Domain Status:	clientTransferProhibited https://icann.org/epp#clientTransferProhibited
Domain Status:	clientUpdateProhibited https://icann.org/epp#clientUpdateProhibited
Domain Status:	clientRenewProhibited https://icann.org/epp#clientRenewProhibited
Domain Status:	clientDeleteProhibited https://icann.org/epp#clientDeleteProhibited
Registry Registrant ID:	Not Available From Registry
Registrant Name:	Registration Private
Registrant Organization:	Domains By Proxy, LLC
Registrant Street:	DomainsByProxy.com
Registrant Street:	100 S. Mill Ave, Suite 1600
Registrant City:	Tempe
Registrant State/Province:	Arizona
Registrant Postal Code:	85281
Registrant Country:	US
Registrant Phone:	+1.480.624.2599
Registrant Phone Ext:	
Registrant Fax:	
Registrant Fax Ext:	
Registrant Email:	Select Contact Domain Holder link at https://www.godaddy.com/whois/results.aspx?domain=WEBGOAT.COM
Registry Tech ID:	Not Available From Registry
Tech Name:	Registration Private
Tech Organization:	Domains By Proxy, LLC
Tech Street:	DomainsByProxy.com
Tech Street:	100 S. Mill Ave, Suite 1600
Tech City:	Tempe
Tech State/Province:	Arizona
Tech Postal Code:	85281
Tech Country:	US
Tech Phone:	+1.480.624.2599
Tech Phone Ext:	
Tech Fax:	
Tech Fax Ext:	
Tech Email:	Select Contact Domain Holder link at https://www.godaddy.com/whois/results.aspx?domain=WEBGOAT.COM
Name Server:	NS41.DOMAINCONTROL.COM
Name Server:	NS42.DOMAINCONTROL.COM
DNSSEC:	unsigned
URL of the ICANN WHOIS Data Problem Reporting System:	http://wdprs.internic.net/

General TLDs Country TLDs

The following domains are available through our preferred partners. Select domains below for more information. (3rd party site)

- Taken domain.
- Available domain.
- Deleted previously owned domain.

WebGoat.com	View Whois
WebGoat.net	View Whois
WebGoat.org	View Whois
WebGoat.info	Buy Domain
WebGoat.biz	Buy Domain
WebGoat.us	Buy Domain

2.3. Conclusions

The WebGoat application contains several vulnerabilities that align with OWASP Top 10 risks. These findings emphasize the importance of secure development practices and periodic security testing.

3. Audit Process Description

3.1. Reconnaissance and Information Gathering

Used Nmap and Burp Suite to scan the application's open ports and obtain details about its server settings and backend technologies.

3.2. Exploitation of Vulnerabilities

- **SQL Injection:** Leveraged SQLMap to manipulate database queries and retrieve sensitive information.
- **Cross-Site Scripting (XSS):** Injected harmful payloads in input fields to observe execution on other user sessions.
- **Security Misconfiguration:** Identified open admin interfaces and improperly configured security headers.
- **Outdated Components:** Verified application dependencies and identified versions with known vulnerabilities.
- **Authentication Failures:** Tested password strength and policies, revealing susceptibility to brute-force attacks.

It is your turn!

You are an employee named John **Smith** working for a big company. The company has an internal system that allows all employees to see their own internal data such as the department they work in and their salary.

The system requires the employees to use a unique *authentication TAN* to view their data.

Your current TAN is **3SL99A**.

Since you always have the urge to be the most highly paid employee, you want to exploit the system so that instead of viewing your own internal data, *you want to take a look at the data of all your colleagues* to check their current

Use the form below and try to retrieve all employee data from the **employees** table. You should not need to know any specific names or TANs to get the information you need.

You already found out that the query performing your request looks like this:

```
"SELECT * FROM employees WHERE last_name = ' " + name + "' AND auth_tan = ' " + auth_tan + "'";
```



Employee Name:

Authentication TAN:

You have succeeded! You successfully compromised the confidentiality of data by viewing internal information that you should not have access to. Well done!

USERID	FIRST_NAME	LAST_NAME	DEPARTMENT	SALARY	AUTH_TAN
--------	------------	-----------	------------	--------	----------

32147	Paulina	Travers	Accounting	46000	P45JSI
-------	---------	---------	------------	-------	--------

34477	Abraham	Holman	Development	50000	UU2ALK
-------	---------	--------	-------------	-------	--------

37648	John	Smith	Marketing	64350	3SL99A
-------	------	-------	-----------	-------	--------

89762	Tobi	Barnett	Development	77000	TA9LL1
-------	------	---------	-------------	-------	--------

96134	Bob	Franco	Marketing	83700	LO9S2V
-------	-----	--------	-----------	-------	--------

Try It! Reflected XSS

The assignment's goal is to identify which field is susceptible to XSS.

It is always a good practice to validate all input on the server side. XSS can occur when unvalidated user input gets used in an HTTP response. In a reflected XSS attack, an attacker can craft a URL with the attack script and post it to the website, email it, or otherwise get a victim to click on it.

An easy way to find out if a field is vulnerable to an XSS attack is to use the `alert()` or `console.log()` methods. Use one of them to find out which field is vulnerable.

Shopping Cart

Shopping Cart Items -- To Buy Now	Price	Quantity	Total
Studio RTA - Laptop/Reading Cart with Tilting Surface - Cherry	69.99	<input type="text" value="1"/>	\$0.00
Dynex - Traditional Notebook Case	27.99	<input type="text" value="1"/>	\$0.00
Hewlett-Packard - Pavilion Notebook with Intel Centrino	1599.99	<input type="text" value="1"/>	\$0.00
3 - Year Performance Service Plan \$1000 and Over	299.99	<input type="text" value="1"/>	\$0.00

Enter your credit card number:

Enter your three digit access code:

Seems like you tried to compromise our shop with an reflected XSS attack.
We do our... "best"... to prevent such attacks. Try again!

Exploit the Comment Field

1. If the form uses XML to process the submitted data, inject a malicious payload into the comment field. Using tools like Burp Suite or Zaproxy, the XML file can be added, and the response can be modified to achieve the desired result.
2. Example of XML:

xml

Copy code

```
<?xml version="1.0"?>
```

```
<!DOCTYPE foo [<!ENTITY xxe SYSTEM "file:///etc/passwd">]>
```

```
<comment>
```

```
&xxe;
```

```
</comment>
```

- `<!DOCTYPE foo>`: Declares the document type.
- `<!ENTITY xxe SYSTEM "file:///etc/passwd">`: Creates an external entity pointing to a system file.
- `xxe;`: Invokes the xxe entity, attempting to read the file.

If the system is vulnerable, the content of files like `/etc/passwd` (on Linux systems) or the system's root directory (`/`) could be listed, and the requested file's content (such as system users in `/etc/passwd`) will be visible in the comments.

If not, the system might have implemented security measures, such as disabling DTD or validating input.

Mitigating XXE

Developers can prevent XXE by ensuring DTDs are disabled in XML parsers, validating and sanitizing user input, and using modern and secure libraries to handle XML.

3.4. Mitigation Strategies

- Implement input validation and parameterized queries to prevent SQL Injection attacks.
- Sanitize all user inputs to reduce Cross-Site Scripting risks.
- Harden server configurations to minimize information disclosure.
- Regularly update and patch application libraries and components.
- Strengthen password policies and adopt multi-factor authentication.

3.5. Tools Utilized

- **Docker:** For deploying the WebGoat environment.
- **SQLMap:** For exploiting SQL Injection vulnerabilities.
- **Nmap:** For network and port scanning.
- **Burp Suite:** For identifying and testing application vulnerabilities.