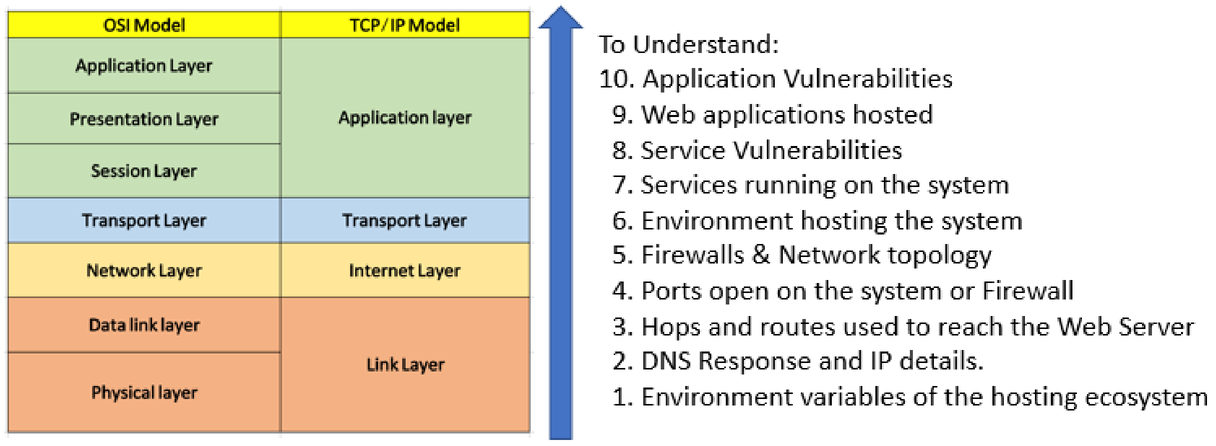
Network and Information Security Management Assessment for e-commerce website

Introduction

This report investigates an e-commerce website used to provide payment services and advice for commercial website operators. Attacks to this type of websites utilise vulnerabilities specific to the e-commerce websites, such as shopping cart, or vulnerabilities common to any web application, such as SQL injection or cross-site scripting. One of the most critical guides that we will consider through this process is the OWASP (OWASP, 2020), which constitutes an essential and globally approved framework to use as a starting point. The methodology, scanning tools, and limitations will be analysed through the first part of the reconnaissance and evidence evaluation. At a later stage, a new report will be produced explaining the vulnerabilities aligned with the industry's standards. The report will also include suggestions to mitigate the potential risks and non-conformities.

Methodology

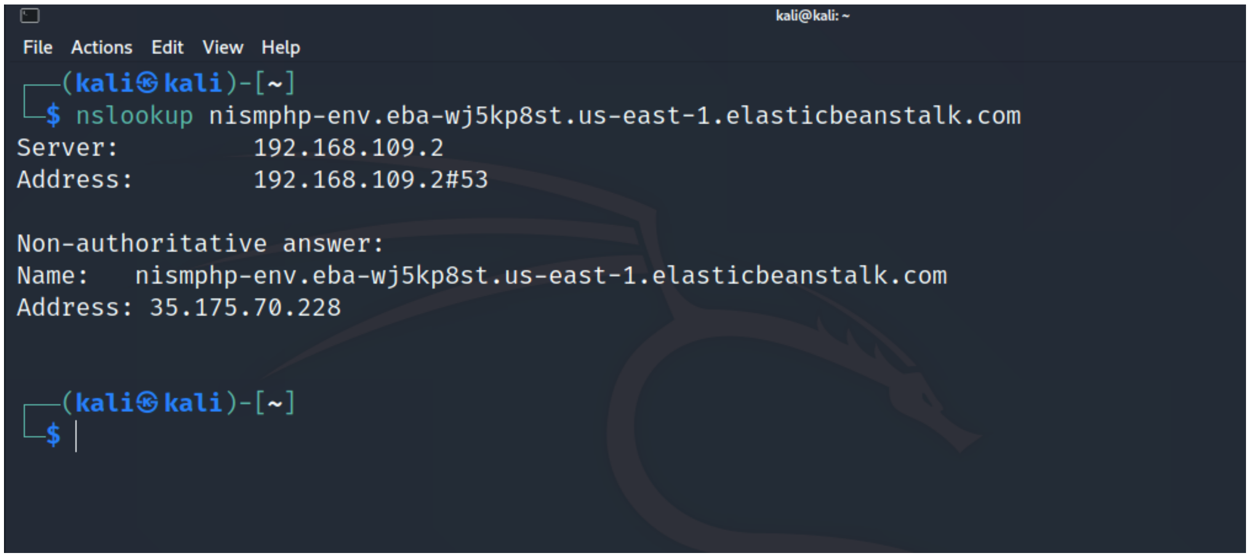
In his article "OSI: The Internet That Wasn't" (2013), Russell shares a brief history and the surrounding thought process that helped the industry conclude TCP/IP stack as the de facto standard. Inspired by the idea of layered modularity, we try to bolster the test procedures and develop a structured approach while examining the website in the bottom-up fashion through the TCP/IP stack (*Figure 1*). We use manual testing to understand the modus operandi rather than blindly relying on automated scripts to understand the testing process better.



OSI Model
TCP/IP Model
Application Layer
Presentation Layer
Session Layer
Application layer
Transport Layer
Transport Layer
Network Layer
Internet Layer
Data link layer
Physical layer
Link Layer
To Understand:
10. Application Vulnerabilities
9. Web applications hosted
8. Service Vulnerabilities
7. Services running on the system
6. Environment hosting the system
5. Firewalls & Network topology
4. Ports open on the system or Firewall
3. Hops and routes used to reach the Web Server
2. DNS Response and IP details.
1. Environment variables of the hosting ecosystem

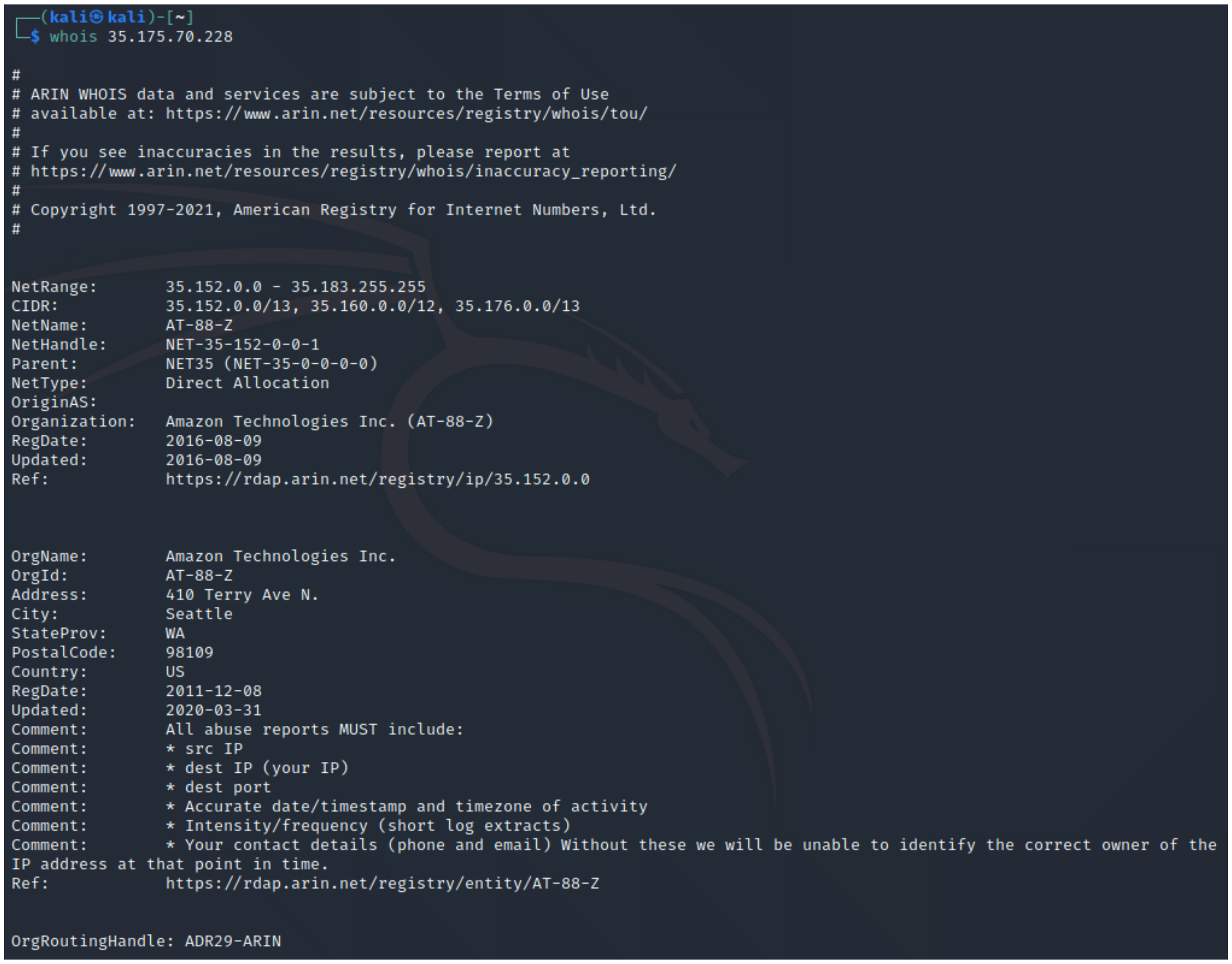
Figure 1: Bottom-Up approach

First, we try to recognise the neighbouring elements in the architecture enveloping the target web server. We found details like the environment hosting the web server, time to live (TTL), hops needed to reach the web server, and the DNS response. Examining the domain name helped us confirm that the web server is hosted on Amazon Web Services. The DNS resolution yielded only one record for an IPv4 address (Figure 2). The IP is registered under American Registry for Internet Numbers (ARIN) in the name of Amazon Technologies .Inc (Figure 3). Traceroute relies on ICMP protocol, so we checked if the server responds to ICMP to find the hops' number until the web server IP. We witnessed that some intermediary hops timed out (with request timed out) possibly due to the firewall configurations on the internet service providers (ISP). Many ISPs disable the ICMP packets internally to avoid flooding and enforce what internet service providers refer to as topology hiding (Figure 5).



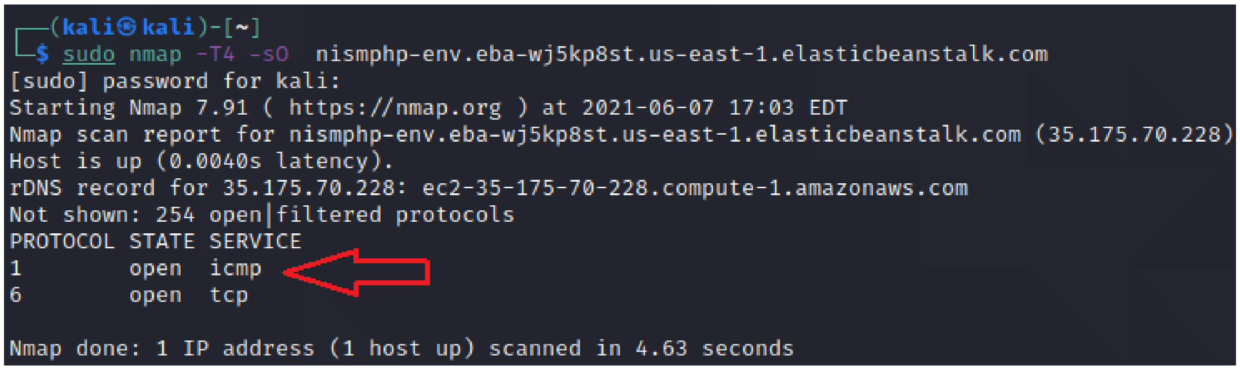
```
┌──(kali㉿kali)-[~]
└─$ nslookup nismphp-env.eba-wj5kp8st.us-east-1.elasticbeanstalk.com
Server:         192.168.109.2
Address:        192.168.109.2#53

Non-authoritative answer:
Name:   nismphp-env.eba-wj5kp8st.us-east-1.elasticbeanstalk.com
Address: 35.175.70.228


┌──(kali㉿kali)-[~]
└─$
```

Figure 2: DNS resolution using NSLOOKUP



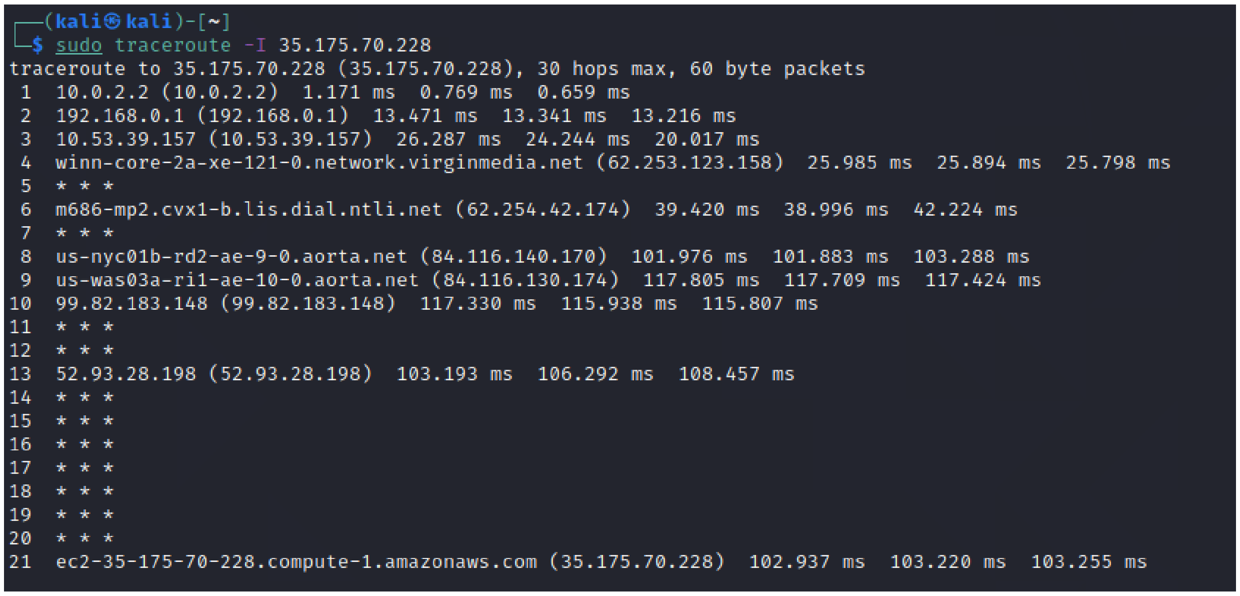
```
┌──(kali㉿kali)-[~]
└─$ whois 35.175.70.228

#
# ARIN WHOIS data and services are subject to the Terms of Use
# available at: https://www.arin.net/resources/registry/whois/tou/
#
# If you see inaccuracies in the results, please report at
# https://www.arin.net/resources/registry/whois/inaccuracy_reporting/
#
# Copyright 1997-2021, American Registry for Internet Numbers, Ltd.
#


NetRange:       35.152.0.0 - 35.183.255.255
CIDR:           35.152.0.0/13, 35.160.0.0/12, 35.176.0.0/13
NetName:        AT-88-Z
NetHandle:      NET-35-152-0-0-1
Parent:         NET35 (NET-35-0-0-0-0)
NetType:        Direct Allocation
OriginAS:
Organization:   Amazon Technologies Inc. (AT-88-Z)
RegDate:        2016-08-09
Updated:        2016-08-09
Ref:            https://rdap.arin.net/registry/ip/35.152.0.0



OrgName:        Amazon Technologies Inc.
OrgId:          AT-88-Z
Address:        410 Terry Ave N.
City:           Seattle
StateProv:      WA
PostalCode:     98109
Country:        US
RegDate:        2011-12-08
Updated:        2020-03-31
Comment:        All abuse reports MUST include:
Comment:        * src IP
Comment:        * dest IP (your IP)
Comment:        * dest port
Comment:        * Accurate date/timestamp and timezone of activity
Comment:        * Intensity/frequency (short log extracts)
Comment:        * Your contact details (phone and email) Without these we will be unable to identify the correct owner of the
IP address at that point in time.
Ref:            https://rdap.arin.net/registry/entity/AT-88-Z


OrgRoutingHandle: ADR29-ARIN
```

Figure 3: IP ownership and registration using WHOIS



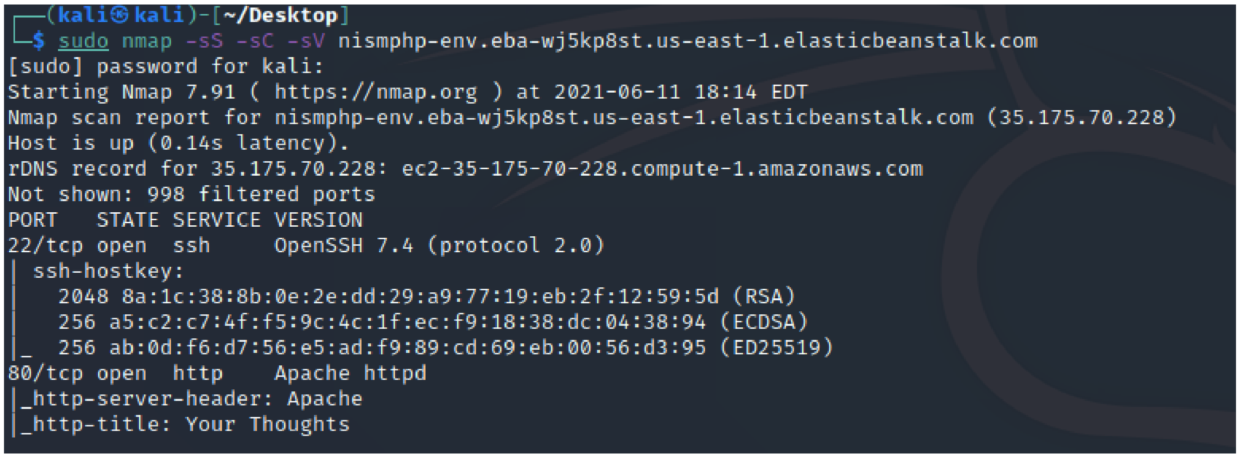
```
┌──(kali㉿kali)-[~]
└─$ sudo nmap -T4 -sO  nismphp-env.eba-wj5kp8st.us-east-1.elasticbeanstalk.com
[sudo] password for kali:
Starting Nmap 7.91 ( https://nmap.org ) at 2021-06-07 17:03 EDT
Nmap scan report for nismphp-env.eba-wj5kp8st.us-east-1.elasticbeanstalk.com (35.175.70.228)
Host is up (0.0040s latency).
rDNS record for 35.175.70.228: ec2-35-175-70-228.compute-1.amazonaws.com
Not shown: 254 open|filtered protocols
PROTOCOL STATE SERVICE
1        open  icmp
6        open  tcp

Nmap done: 1 IP address (1 host up) scanned in 4.63 seconds
```

Figure 4: Checking Server for ICMP protocol



```
┌──(kali㉿kali)-[~]
└─$ sudo traceroute -I 35.175.70.228
traceroute to 35.175.70.228 (35.175.70.228), 30 hops max, 60 byte packets
 1  10.0.2.2 (10.0.2.2)  1.171 ms  0.769 ms  0.659 ms
 2  192.168.0.1 (192.168.0.1)  13.471 ms  13.341 ms  13.216 ms
 3  10.53.39.157 (10.53.39.157)  26.287 ms  24.244 ms  20.017 ms
 4  winn-core-2a-xe-121-0.network.virginmedia.net (62.253.123.158)  25.985 ms  25.894 ms  25.798 ms
 5  * * *
 6  m686-mp2.cvx1-b.lis.dial.ntli.net (62.254.42.174)  39.420 ms  38.996 ms  42.224 ms
 7  * * *
 8  us-nyc01b-rd2-ae-9-0.aorta.net (84.116.140.170)  101.976 ms  101.883 ms  103.288 ms
 9  us-was03a-ri1-ae-10-0.aorta.net (84.116.130.174)  117.805 ms  117.709 ms  117.424 ms
10  99.82.183.148 (99.82.183.148)  117.330 ms  115.938 ms  115.807 ms
11  * * *
12  * * *
13  52.93.28.198 (52.93.28.198)  103.193 ms  106.292 ms  108.457 ms
14  * * *
15  * * *
16  * * *
17  * * *
18  * * *
19  * * *
20  * * *
21  ec2-35-175-70-228.compute-1.amazonaws.com (35.175.70.228)  102.937 ms  103.220 ms  103.255 ms
```

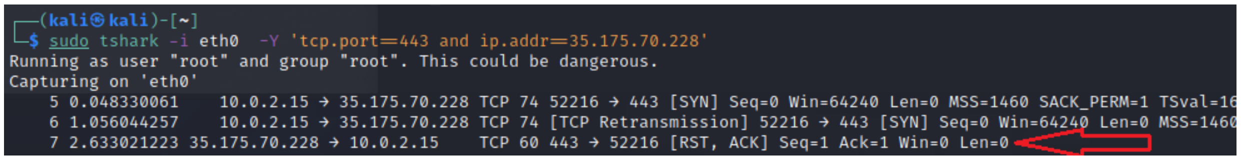
Figure 5: Reckoning the total path to the final destination through the hops



```
┌──(kali㉿kali)-[~/Desktop]
└─$ sudo nmap -sS -sC -sV nismphp-env.eba-wj5kp8st.us-east-1.elasticbeanstalk.com
[sudo] password for kali:
Starting Nmap 7.91 ( https://nmap.org ) at 2021-06-11 18:14 EDT
Nmap scan report for nismphp-env.eba-wj5kp8st.us-east-1.elasticbeanstalk.com (35.175.70.228)
Host is up (0.14s latency).
rDNS record for 35.175.70.228: ec2-35-175-70-228.compute-1.amazonaws.com
Not shown: 998 filtered ports
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 7.4 (protocol 2.0)
| ssh-hostkey:
|   2048 8a:1c:38:8b:0e:2e:dd:29:a9:77:19:eb:2f:12:59:5d (RSA)
|   256 a5:c2:c7:4f:f5:9c:4c:1f:ec:f9:18:38:dc:04:38:94 (ECDSA)
|_  256 ab:0d:f6:d7:56:e5:ad:f9:89:cd:69:eb:00:56:d3:95 (ED25519)
80/tcp open  http    Apache httpd
|_http-server-header: Apache
|_http-title: Your Thoughts
```

Figure 6: Using nmap to scan for open TCP ports and the services running on them

We use NMAP to scan for the Transmission Control Protocol (TCP) ports, revealing that two services are open and listening services on port 22 and port 80. Since TCP protocol works on a 3-way handshake; we try to see beyond the outcome from the scan above and initiate handshake by sending TCP SYN for all ports (1 - 65535) towards the destination server using nping. An intriguing find worth mentioning is that port 443 was also open on the firewall, and the TCP SYN were rejected with an RST by the destination server (Figure 7 & 8).



```
┌──(kali㉿kali)-[~]
└─$ sudo tshark -i eth0  -Y 'tcp.port==443 and ip.addr==35.175.70.228'
Running as user "root" and group "root". This could be dangerous.
Capturing on 'eth0'
    5 0.048330061    10.0.2.15 → 35.175.70.228 TCP 74 52216 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=16
    6 1.056044257    10.0.2.15 → 35.175.70.228 TCP 74 [TCP Retransmission] 52216 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460
    7 2.633021223 35.175.70.228 → 10.0.2.15    TCP 60 443 → 52216 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
```

Figure 7: RST received from port 443

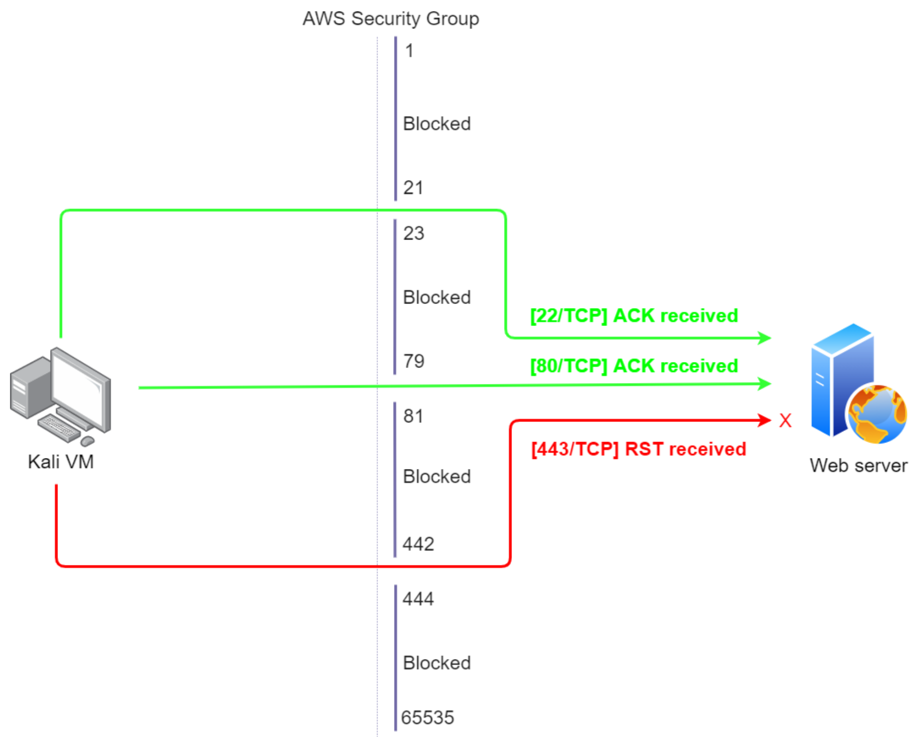
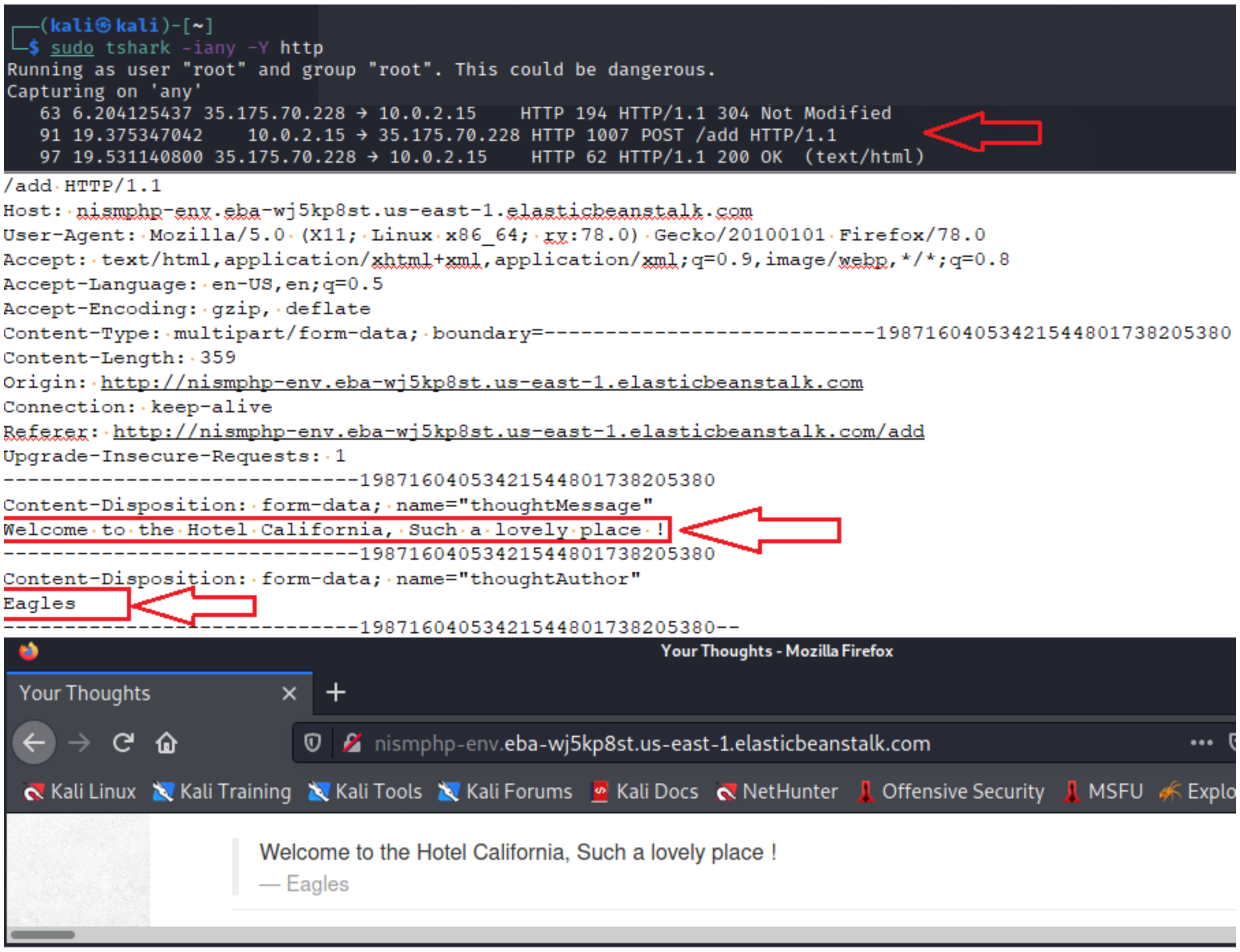


Figure 8: Open TCP Ports on the firewall

The internet browser helped us confirm that the web server was running on port 80, making it insecure. This fact makes the HTTP requests and responses human-readable when eavesdropped using a packet capturing tool like tshark without any need to decrypt it (*Figure 9*). We use HTTP add to publish a comment on the website and confirm that the transactions were readable in the capture on the local interface using tshark and investigated with Wireshark.



```
┌──(kali㉿kali)-[~]
└─$ sudo tshark -iany -Y http
Running as user "root" and group "root". This could be dangerous.
Capturing on 'any'
   63 6.204125437 35.175.70.228 → 10.0.2.15    HTTP 194 HTTP/1.1 304 Not Modified
   91 19.375347042    10.0.2.15 → 35.175.70.228 HTTP 1007 POST /add HTTP/1.1
   97 19.531140800 35.175.70.228 → 10.0.2.15    HTTP 62 HTTP/1.1 200 OK  (text/html)
```

```
/add HTTP/1.1
Host: nismphp-env.eba-wj5kp8st.us-east-1.elasticbeanstalk.com
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: multipart/form-data; boundary=---------------------------19871604053421544801738205380
Content-Length: 359
Origin: http://nismphp-env.eba-wj5kp8st.us-east-1.elasticbeanstalk.com
Connection: keep-alive
Referer: http://nismphp-env.eba-wj5kp8st.us-east-1.elasticbeanstalk.com/add
Upgrade-Insecure-Requests: 1
-----------------------------19871604053421544801738205380
Content-Disposition: form-data; name="thoughtMessage"
Welcome to the Hotel California, Such a lovely place !
-----------------------------19871604053421544801738205380
Content-Disposition: form-data; name="thoughtAuthor"
Eagles
-----------------------------19871604053421544801738205380--
```

Welcome to the Hotel California, Such a lovely place !
— Eagles

Figure 9: Human-readable HTTP POST request

We ran a bash script (Figure 10) with the above understanding to see if the system has any defensive mechanism to avoid flooding and found no security measures like CAPTCHA or timeout to avoid automated flooding.

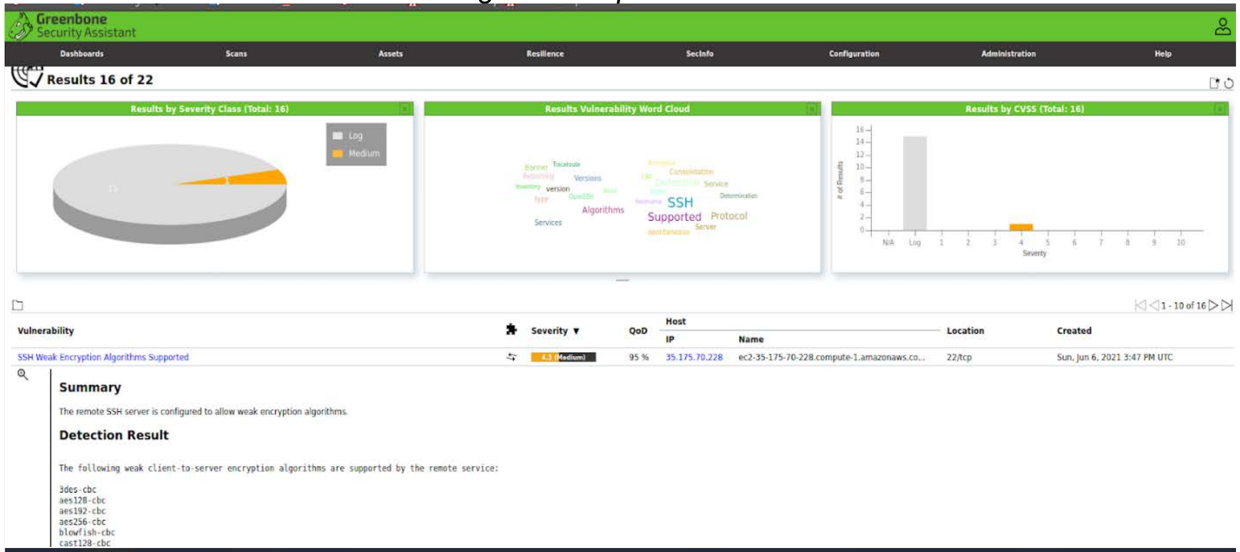
```bash
#!/usr/bin/bash
timestamp=$(date +%d-%m-%Y_%H-%M-%S);
for i in {1..10000};
do
        curl -d 'thoughtMessage=10th process. Will I kill the server???? test'$i' bash automate date_and_time:'$timestamp'&thoughtAuthor=Lukasz '$i'' 'http://nismphp-env.eba-wj5kp8st.us-east-1.elasticbeanstalk.com/add'
done
```

Figure 10: Bash script

Subsequently, we switched to scan the available SSH and HTTP server using penetration testing suites like OpenVAS (Figure 11), Nikto (Figure 12), OWASP ZAP (Figure 13) and Metasploit (Figure 14).

OpenVAS and Nikto, provided extensive reports with explanations, matching Common Vulnerabilities and Exposures (CVE) whilst suggesting mitigations. The scan reports also yielded some false positives, which were filtered in the process. OWASP ZAP proved to be instrumental and more efficient than the aforementioned suites. The report had an impressive depth of information. After that, we used Metasploit but were not able to exploit/find any additional SSH vulnerability.

Figure 11: OpenVAS GUI



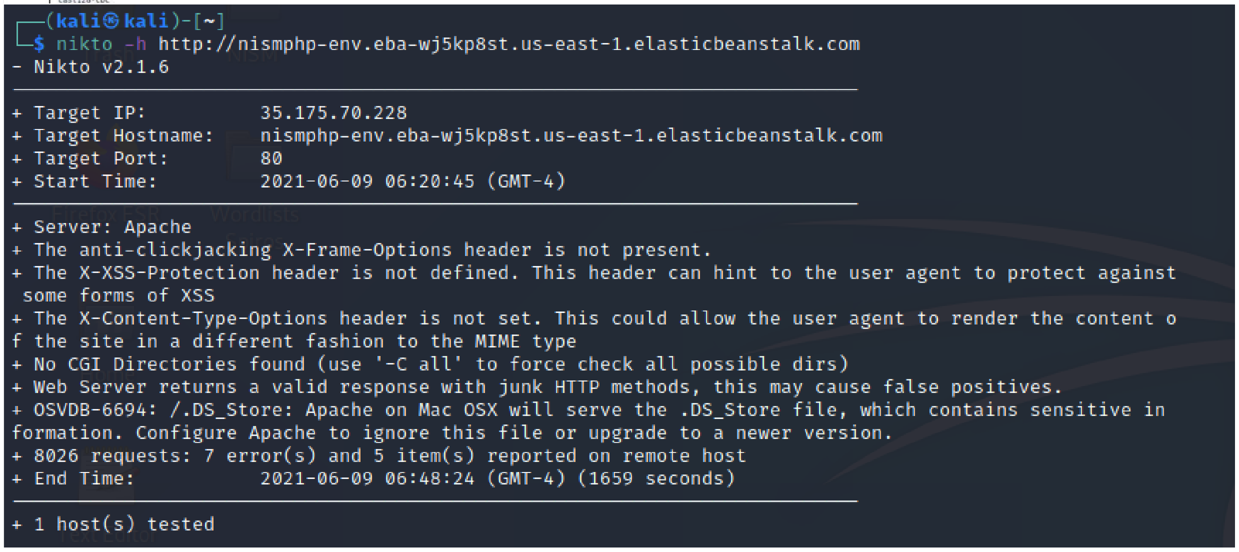
Vulnerability		Severity ▼	QoD	Host IP	Host Name	Location	Created
SSH Weak Encryption Algorithms Supported		4.3 (Medium)	95 %	35.175.70.228	ec2-35-175-70-228.compute-1.amazonaws.co...	22/tcp	Sun, Jun 6, 2021 3:47 PM UTC

Summary

The remote SSH server is configured to allow weak encryption algorithms.

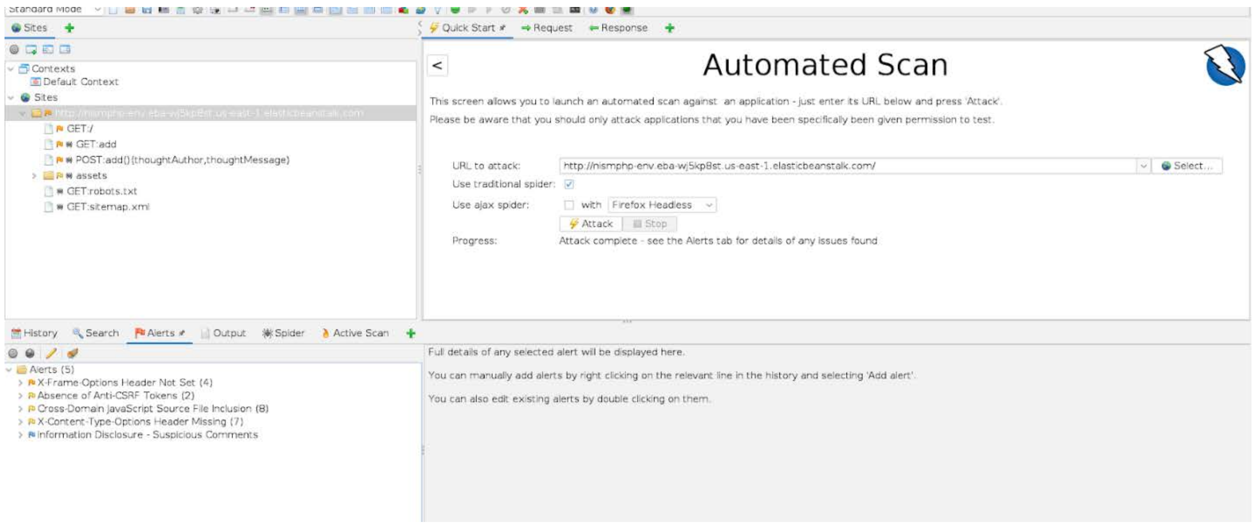
Detection Result

```
The following weak client-to-server encryption algorithms are supported by the remote service:

3des-cbc
aes128-cbc
aes192-cbc
aes256-cbc
blowfish-cbc
cast128-cbc
```



```
┌──(kali㉿kali)-[~]
└─$ nikto -h http://nismphp-env.eba-wj5kp8st.us-east-1.elasticbeanstalk.com
- Nikto v2.1.6
---------------------------------------------------------------------------
+ Target IP:          35.175.70.228
+ Target Hostname:    nismphp-env.eba-wj5kp8st.us-east-1.elasticbeanstalk.com
+ Target Port:        80
+ Start Time:         2021-06-09 06:20:45 (GMT-4)
---------------------------------------------------------------------------
+ Server: Apache
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against
 some forms of XSS
+ The X-Content-Type-Options header is not set. This could allow the user agent to render the content o
f the site in a different fashion to the MIME type
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ Web Server returns a valid response with junk HTTP methods, this may cause false positives.
+ OSVDB-6694: /.DS_Store: Apache on Mac OSX will serve the .DS_Store file, which contains sensitive in
formation. Configure Apache to ignore this file or upgrade to a newer version.
+ 8026 requests: 7 error(s) and 5 item(s) reported on remote host
+ End Time:           2021-06-09 06:48:24 (GMT-4) (1659 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested
```

Figure 12: NIKTO CLI



Automated Scan

This screen allows you to launch an automated scan against an application - just enter its URL below and press 'Attack'.
Please be aware that you should only attack applications that you have been specifically been given permission to test.

URL to attack: http://nismphp-env.eba-wj5kp8st.us-east-1.elasticbeanstalk.com/
Use traditional spider: ✓
Use ajax spider: ☐ with Firefox Headless
Progress: Attack complete - see the Alerts tab for details of any issues found

Contexts
Default Context
Sites
http://nismphp-env.eba-wj5kp8st.us-east-1.elasticbeanstalk.com
GET:/
GET:add
POST:add()(thoughtAuthor,thoughtMessage)
assets
GET:robots.txt
GET:sitemap.xml

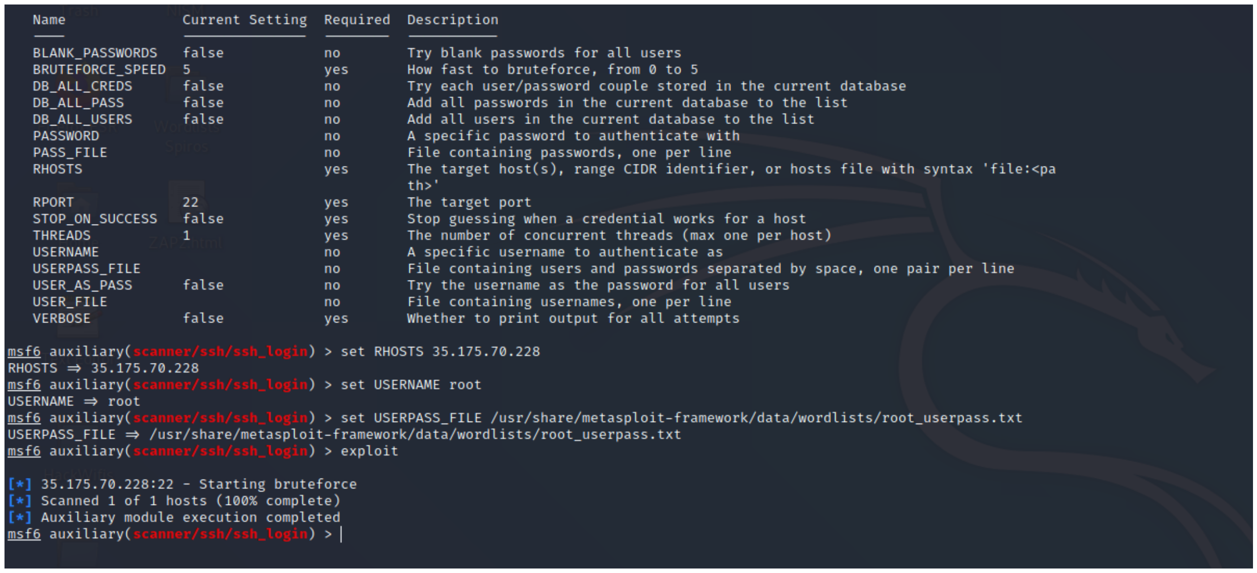
Alerts (5)
X-Frame-Options Header Not Set (4)
Absence of Anti-CSRF Tokens (2)
Cross-Domain JavaScript Source File Inclusion (8)
X-Content-Type-Options Header Missing (7)
Information Disclosure - Suspicious Comments

Full details of any selected alert will be displayed here.

You can manually add alerts by right clicking on the relevant line in the history and selecting 'Add alert'.

You can also edit existing alerts by double clicking on them.

Figure 13: OWASP ZAP GUI



```
   Name              Current Setting  Required  Description
   ----              ---------------  --------  -----------
   BLANK_PASSWORDS   false            no        Try blank passwords for all users
   BRUTEFORCE_SPEED  5                yes       How fast to bruteforce, from 0 to 5
   DB_ALL_CREDS      false            no        Try each user/password couple stored in the current database
   DB_ALL_PASS       false            no        Add all passwords in the current database to the list
   DB_ALL_USERS      false            no        Add all users in the current database to the list
   PASSWORD                           no        A specific password to authenticate with
   PASS_FILE                          no        File containing passwords, one per line
   RHOSTS                             yes       The target host(s), range CIDR identifier, or hosts file with syntax 'file:<pa
                                                th>'
   RPORT             22               yes       The target port
   STOP_ON_SUCCESS   false            yes       Stop guessing when a credential works for a host
   THREADS           1                yes       The number of concurrent threads (max one per host)
   USERNAME                           no        A specific username to authenticate as
   USERPASS_FILE                      no        File containing users and passwords separated by space, one pair per line
   USER_AS_PASS      false            no        Try the username as the password for all users
   USER_FILE                          no        File containing usernames, one per line
   VERBOSE           false            yes       Whether to print output for all attempts

msf6 auxiliary(scanner/ssh/ssh_login) > set RHOSTS 35.175.70.228
RHOSTS => 35.175.70.228
msf6 auxiliary(scanner/ssh/ssh_login) > set USERNAME root
USERNAME => root
msf6 auxiliary(scanner/ssh/ssh_login) > set USERPASS_FILE /usr/share/metasploit-framework/data/wordlists/root_userpass.txt
USERPASS_FILE => /usr/share/metasploit-framework/data/wordlists/root_userpass.txt
msf6 auxiliary(scanner/ssh/ssh_login) > exploit

[*] 35.175.70.228:22 - Starting bruteforce
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/ssh/ssh_login) > 
```

Figure 14: Metasploit CLI

Timeline and Limitations

With the limited time we had to explore the website, we were still able to coordinate the activities successfully, allowing all the team members to try a plethora of tools using the methodology mentioned above. We concentrated our efforts to perform the scanning independently and then converged to discuss the results/observations. However, we believe that we could have subjected the website to more vulnerability assessment tools with more time.

The web server was hosting a basic PHP web site, with minimal functionality. The scans reported only two processes running SSH and HTTP. As a result, we were not left with many directions to pursue, such as SQL injections, SSL/TLS certificate-based vulnerabilities, to name some. A full-fledged penetration testing could have helped us in validating vulnerabilities. Additionally, access to enterprise assessment tools could have made the scanning process more thorough.

Conclusion

The methodology we have used has helped us glean an organised insight while strengthening the understanding of the layered ideology of the TCP/IP thus justifying its popularity in the data networking realm. Kannan et al. (2016) stated that understanding the parts of the network and the relationship between them is essential for a comprehensive network and information security management (NISM) assessment. This focus has allowed us to search for vulnerabilities on each OSI layer (Amos, 2020) and select the appropriate tools for discovering them. Furthermore, the tools and commands that we have used have helped us acquire a good knowledge of penetration testing and NISM assessment fundamentals.

The assessment process was focused on discovering vulnerabilities in the provided server. We intend to expand the assessment with information about e-commerce compliance and standards. Compliance standards serve as vital pillars to address the security and privacy requirements inherent to e-commerce websites.

According to a United Nations Conference on Trade and Development report (UNCTAD, 2020), most governments have legislation about governance and compliance in e-commerce environments. All legislations focus on data protection and privacy and require compliance to GDPR or similar regulations like the California Consumer Privacy Act (CCPA) in the United States. Likewise, the Payment Card Industry Security Standards Council (PCI SSC, N.D.) has defined standards for creating secure e-commerce products and solutions.

References

Amos, J. (2020) 7 Layers of Cybersecurity Threats in the ISO-OSI Model. Available from: https://training.nhlearninggroup.com/blog/7-layers-of-cybersecurity-threats-in-the-iso-osi-model [Accessed 9 June 2021].

Kannan, U., Swamidurai, R., & Umphress, D. (2016) System Dynamics as a Tool for Modeling Application Layer Cyber Security. Available from: https://worldcomp-proceedings.com/proc/p2016/MSV3723.pdf [Accessed 10 June 2021].

PCI Security Standards Council (N.D.) PCI Security Standards Overview. Available from: https://www.pcisecuritystandards.org/pci_security/standards_overview [Accessed 13 June 2021].

Russell, A.L. (2013) OSI: The Internet That Wasn't - IEEE Spectrum. Available from: https://spectrum.ieee.org/tech-history/cyberspace/osi-the-internet-that-wasnt [Accessed 5 June 2021].

UNCTAD (2020) Summary of Adoption of E-Commerce Legislation Worldwide. Available from: https://unctad.org/topic/ecommerce-and-digital-economy/ecommerce-law-reform/summary-adoption-e-commerce-legislation-worldwide [Accessed 5 June 2021].

Bibliography

Geer, D. (2015) *8 penetration testing tools that will do the job | Network World*. Available from: https://www.networkworld.com/article/2944811/8-penetration-testing-tools-that-will-do-the-job.html [Accessed 14 June 2021].

Greenbone Networks GmbH (N.D.) *OpenVAS - Open Vulnerability Assessment Scanner*. Available from: https://www.openvas.org/#about [Accessed 6 June 2021].

Gordon, L. (N.D.) *Nmap: the Network Mapper - Free Security Scanner*. Available from: https://nmap.org/ [Accessed 8 June 2021].

OWASP Foundation Inc. (N.D.) *OWASP Top Ten Web Application Security Risks*. Available from: https://owasp.org/www-project-top-ten/ [Accessed 7 June 2021].

OWASP Foundation Inc. (N.D.) *OWASP ZAP Zed Attack Proxy* | *OWASP*. Available from: https://owasp.org/www-project-zap/ [Accessed 10 June 2021].

Rapid7 Inc. (N.D.) *Metasploit* | *Penetration Testing Software, Pen Testing Security* | *Metasploit*. Available from: https://www.metasploit.com/ [Accessed 11 June 2021].

Sullo, C. (N.D.) *Nikto2* | *CIRT.net*. Available from: https://cirt.net/Nikto2 [Accessed 9 June 2021].

Wireshark Foundation (N.D.) *Wireshark · Go Deep.* . Available from: https://www.wireshark.org/ [Accessed 11 June 2021].