Sri Lanka Institute of Information Technology

# Web Security Audit

IE2062 – Web Security

Submitted by:

| Student Registration Number | Student Name |
| --- | --- |
| [redacted] | Senarathna K.M.G.S.B |

Domain

Verizon media – yahoo.com

# Table of Contents

# Abstract

Web security audits and Bug bounties has been popular terms in web application penetration testing scopes for years. Therefore security researchers and academics often engage in these activities in order to update their knowledge on the industry or because of the monetary benefit that they are receiving if they are able to find a considerable bug/vulnerability in a web application that they are testing.

According to the web security module's assignment students are required to perform a security audit on a selected domain and create necessary documentations along with it. This report is followed by the web security audit that was performed based on the domain Verizon media's yahoo.com.

Initially an introduction to the web security auditing and a brief introduction about the bug bounty programs is included. Following those introductions, in the latter part the results of the information gathering phase is included. After that, the methodology that was carried out when discovering vulnerabilities is described. Discovery of vulnerabilities and bad practices of the web application was done using few penetration testing tools. Therefore, a brief introduction about those tools is also included in this report.

Later, after describing the methodology of the web security audit, the results about the vulnerabilities and bad practices discovered when analyzing the domain is included. A brief analysis of the vulnerabilities and the action that should be taken by the organization is also included under each vulnerability.

Finally, a conclusion and instructions to follow by the web application developers is included to prevent any malicious activities using found vulnerabilities.

# Introduction

## Web security audit

Web security audit is a practice that is analyzing a particular web application's infrastructure in order to discover vulnerabilities and bad practices that could attract attackers to exploit them and perform various malicious activities based on the web application's various processes and it's users data. A web security audit is performed with various series of penetration testing instances that checks the web application's subdomains to discover vulnerabilities.

A web security audit carries numerous benefits to an organization which manages a web application.

- Ensure that the web application's current security infrastructure is adequate or not.
- Reduce cost of a shutdown of the web application in case an unexpected attack happens.
- Ensure the clients that the organization is concerned about their sensitive data, which will help to maintain the organization's reputation [1] (https://www.varonis.com/blog/security-audit/)

In addition, there are few types of web security audits.

- Vulnerability assessment
- Manual web security audit
- Automated web security audit

Web security audit which this report is based is mainly an automated web security audit where most of the time automated tools were used to discover vulnerabilities of the web application infrastructure.

# Domain

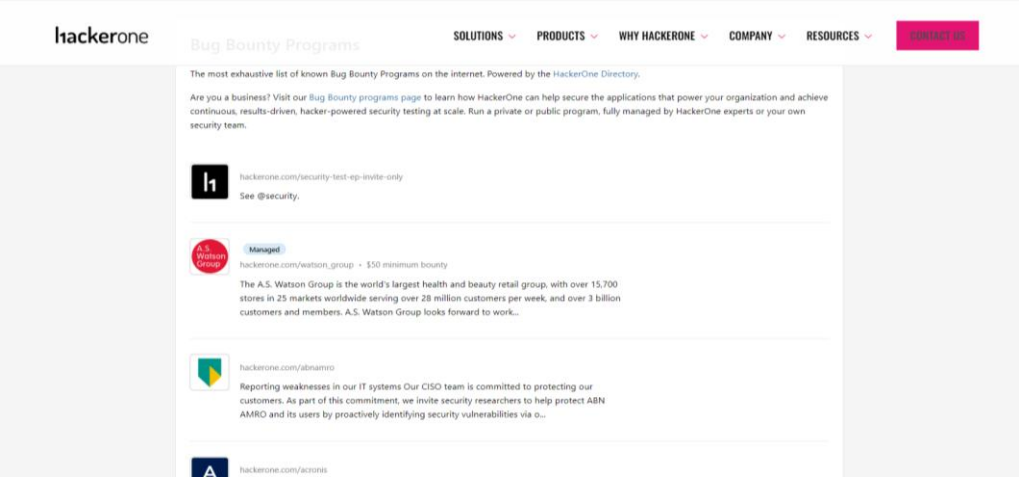
## Discovering the domain

Initially, to perform the web security audit as required in the assignment, a domain of a web application that provides a bug bounty program should be found. There are numerous organizations and platforms that provides opportunity to security researchers to test on their security infrastructure in order to discover vulnerabilities of the web applications. In addition, these platforms pay bounties to bug bounty hunters who are able to find vulnerabilities in their web applications.

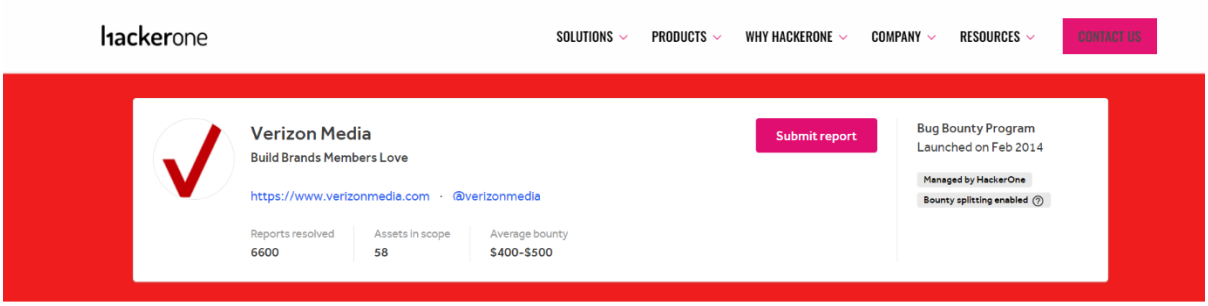
Examples for bug bounty platforms**:**

- **Hackerone**
- **Bug crowd**
- **Cobalt**
- **Synack**
- **Facebook**
- **Google**

To conduct this web security audit, I selected **Hackerone** as the bug bounty platform. In Hackerone, there are numerous bug bounty programs hosted by various organizations.



Among the various bug bounty programs provided by Hackerone, for this web audit the selected program is **Verizon media's bug bounty program**.




Verizon media is the parent organization of the famous entertainment and communication brands such as **yahoo**, **HuffPost** and **TechCrunch**. These web applications host millions of users daily. Therefore, Verizon media's security team allows security researchers to analyze their domains and report any issues in the web application infrastructure.

For this assignment I only selected **yahoo.com** and its subdomains as the main domain to perform the web security audit. Therefore, in this web audit only the aspects of yahoo.com and its subdomains will be analyzed.

Initially before starting the web security audit, all the security researchers must read the program rules and legal aspects of the bug bounty program to make sure that they do not violate the organization's data policies and to not to use or manipulate it's users sensitive data.

Therefore, in the beginning I read and understood the rules of the bug bounty program of Verizon media. All six rules are as listed below.

**Program rules of Verizon media**

1. Test vulnerabilities only against accounts that you own or accounts that you have permission from the account holder to test against.

2. Never use a finding to compromise/exfiltrate data or pivot to other systems. Use a proof of concept only to demonstrate an issue.
3. If sensitive information--such as personal information, credentials, etc.--is accessed as part of a vulnerability, it must not be saved, stored, transferred, accessed, or otherwise processed after initial discovery. All copies of sensitive information must be returned to Verizon Media and may not be retained.

4. Researchers may not and are not authorized to engage in any activity that would be disruptive, damaging or harmful to Verizon Media, its brands or its users. This includes social engineering, phishing, physical security and denial of service attacks against users, employees, or Verizon Media as a whole.

5. Abide by the program scope. Only reports submitted to this program and against assets in scope will be eligible for monetary award.

6. Researchers may not publicly disclose vulnerabilities (sharing any details whatsoever with anyone other than authorized Verizon Media or HackerOne employees), or otherwise share vulnerabilities with a third party, without Verizon Media's express written permission. [2]

Abiding by these rules' security researchers must conduct their vulnerability assessments in order to receive or to be eligible to a monetary payment. In addition, Verizon media has published a list of valued vulnerabilities so that researchers can get a clear idea about what vulnerabilities they should focused on.

**List of valued vulnerabilities**

| Severity (low) | Severity (high) | CWE-ID | Common Weakness Enumeration | Bug Examples |
|---|---|---|---|---|
| Critical | Critical | CWE-78 | OS Command Injection | Remote Code Execution; Code Injection; LDAP Injection |
| Critical | Critical | CWE-120 | Classic Buffer Overflow | Buffer Overflow |
| High | Critical | CWE-89 | SQL Injection | SQL Injection |
| Medium | Critical | CWE-918 | Server-Side Request Forgery | SSRF (unrestricted); Content-Restricted SSRF; Error-based SSRF (true/false); Blind SSRF |
| High | Critical | CWE-732 | Incorrect Permission Assignment for Critical Resource | IDOR; Horizontal Privilege Escalation; Vertical Privilege Escalation |
| Critical | Critical | CWE-91 | XML Injection | XML Injection |
| Critical | Critical | CWE-611 | Improper Restriction of XML External Entity Reference | XXE |
| High | Critical | CWE-134 | Uncontrolled Format String | Insecure Deserialization |
| High | Critical | CWE-250 | Execution with Unnecessary Privileges | Privilege Escalation to System Account |
| Medium | High | CWE-444 | Inconsistent Interpretation of HTTP Requests | HTTP Request Smuggling |
| Low | Critical | CWE-829 | Inclusion of Functionality from Untrusted Control Sphere | Server Side Includes Injection; Local File Inclusion; Directory Traversal |

| Severity (low) | Severity (high) | CWE-ID | Common Weakness Enumeration | Bug Examples |
|---|---|---|---|---|
| Medium | High | CWE-306 | Missing Authentication for Critical Function | Exposed Administrative Interface |
| Medium | Critical | CWE-862 | Missing Authorization | Horizontal Privilege Escalation; Vertical Privilege Escalation; IDOR |
| Low | Critical | CWE-200 | Information Exposure | User Enumeration with PII; Credentials on GitHub; Confidential Information Exposure |
| Informative | High | CWE-863 | Incorrect Authorization | Authorization Bypass; Account Takeover; Social Media Takeover (Brand, <12mo); Social Media Takeover (Personal); Social Media Takeover (Brand, >12mo) |
| Medium | High | CWE-798 | Use of Hard-coded Credentials | Hard Coded Credentials |
| Medium | High | CWE-434 | Unrestricted Upload of File with Dangerous Type | Unfiltered File Upload |
| Low | High | CWE-203 | Information Exposure Through Discrepancy | PHP Admin Information page; MySQL Information page (w/ credentials); Apache Status page |
| Medium | Medium | CWE-494 | Download of Code Without Integrity Check | S3 Bucket Upload |
| Low | Medium | CWE-311 | Missing Encryption of Sensitive Data | Cleartext Submission of Passwords |
| Low | Medium | CWE-807 | Reliance on Untrusted Inputs in a Security Decision | |

| Severity (low) | Severity (high) | CWE-ID | Common Weakness Enumeration | Bug Examples |
|---|---|---|---|---|
| Low | Medium | CWE-79 | Cross-Site Scripting | Stored XSS; POST-Based XSS; GET-Based XSS; DOM-Based XSS; Flash-based XSS; CSS Injection |
| Medium | Medium | CWE-352 | Cross-Site Request Forgery | State-Changing CSRF; Non-State-Changing CSRF |
| Low | Medium | CWE-16 | Misconfiguration | Subdomain Takeover; Dangling DNS Record |
| Medium | Medium | CWE-93 | CRLF Injection | CRLF Injection |
| Low | Low | CWE-601 | Open Redirect | Open Redirect |
| Informative | Low | CWE-327 | Use of a Broken or Risky Cryptographic Algorithm | Weak CAPTCHA |
| Informative | Low | CWE-307 | Improper Restriction of Excessive Authentication Attempts | Lack of Rate Limiting on Login; CAPTCHA Bypass |

**Scope**

Although there are many sub domains described in the Verizon media's scope, subdomains that are mainly focused on this web audit are listed below.

- **data.mail.yahoo.com**
- **mail.yahoo.com**
- **login.yahoo.com**
- **sports.yahoo.com**
- **finance.yahoo.com**
- **twpay.buy.yahoo.com**
- le.yahooapis.com
- onepush.query.yahoo.com
- proddata.xobni.yahoo.com

Besides the above selected subdomains, scope contains various subdomains related to **HuffPost.com (**.huffingtonpost.com, *.huffpost.com ,*.huffpost.net, *.huffingtonpost.co.uk ,*.huffingtonpost.ca, *.huffingtonpost.es, *.huffingtonpost.fr, *.huffingtonpost.gr ,*.huffingtonpost.in**),** subdomains related to **news.yahoo.com, TechCrunch.com** and many other subdomains which will not analyzed due to the limited time of the assignment.

**Some of the out of scope sub domains are,**

yahooshopping.myguide.hk, mail.yahoo.com/cal/ (calender.yahoo.com), *.yahoo.com.tw, news.campaign.yahoo.com.tw, iOS: TPDirect.framework, tw.finance.yahoo.com, *. molo.ch

# Methodology

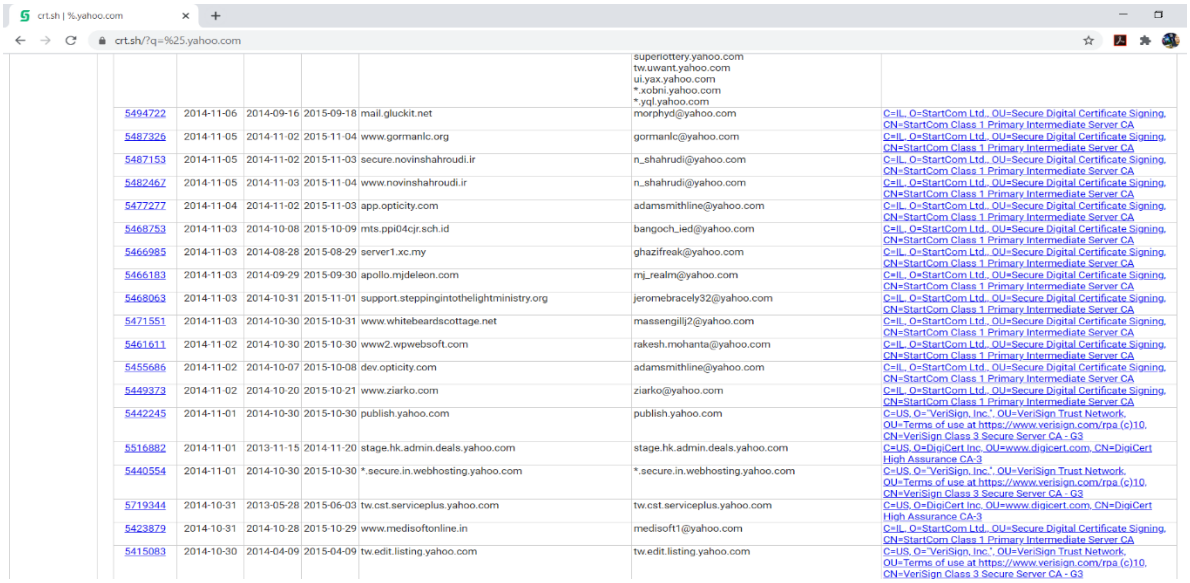
This part of the report contains the detailed description about the methodologies used to conduct this web audit followed by the screenshots of various tools and methods used in various phases of the web security audit.

## 1. Information gathering

When conducting a web audit, gathering information plays a significant role of the audit because all the analyzations that are performed afterwards depends on the gathered information at the initial phase. In this web audit, in order to check subdomains and gather information about them crt.sh and sublist3r tools were used.

### Crt.sh ( https://crt.sh/ )

**crt.sh** is a web interface to a distributed database called the certificate transparency logs. This will be helpful to researchers to get a brief idea about the selected domain and its infrastructure.




| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | | | | | superlottery.yahoo.com<br>tw.uwant.yahoo.com<br>ui.yax.yahoo.com<br>*.xobni.yahoo.com<br>*.yql.yahoo.com | |
| | 5494722 | 2014-11-06 | 2014-09-16 | 2015-09-18 | mail.gluckit.net | morphyd@yahoo.com | C=IL, O=StartCom Ltd., OU=Secure Digital Certificate Signing, CN=StartCom Class 1 Primary Intermediate Server CA |
| | 5487326 | 2014-11-05 | 2014-11-02 | 2015-11-04 | www.gormanlc.org | gormanlc@yahoo.com | C=IL, O=StartCom Ltd., OU=Secure Digital Certificate Signing, CN=StartCom Class 1 Primary Intermediate Server CA |
| | 5487153 | 2014-11-05 | 2014-11-02 | 2015-11-03 | secure.novinshahroudi.ir | n_shahrudi@yahoo.com | C=IL, O=StartCom Ltd., OU=Secure Digital Certificate Signing, CN=StartCom Class 1 Primary Intermediate Server CA |
| | 5482467 | 2014-11-05 | 2014-11-03 | 2015-11-04 | www.novinshahroudi.ir | n_shahrudi@yahoo.com | C=IL, O=StartCom Ltd., OU=Secure Digital Certificate Signing, CN=StartCom Class 1 Primary Intermediate Server CA |
| | 5477277 | 2014-11-04 | 2014-11-02 | 2015-11-03 | app.opticity.com | adamsmithline@yahoo.com | C=IL, O=StartCom Ltd., OU=Secure Digital Certificate Signing, CN=StartCom Class 1 Primary Intermediate Server CA |
| | 5468753 | 2014-11-03 | 2014-10-08 | 2015-10-09 | mts.ppi04cjr.sch.id | bangoch_ied@yahoo.com | C=IL, O=StartCom Ltd., OU=Secure Digital Certificate Signing, CN=StartCom Class 1 Primary Intermediate Server CA |
| | 5466985 | 2014-11-03 | 2014-08-28 | 2015-08-29 | server1.xc.my | ghazifreak@yahoo.com | C=IL, O=StartCom Ltd., OU=Secure Digital Certificate Signing, CN=StartCom Class 1 Primary Intermediate Server CA |
| | 5466183 | 2014-11-03 | 2014-09-29 | 2015-09-30 | apollo.mjdeleon.com | mj_realm@yahoo.com | C=IL, O=StartCom Ltd., OU=Secure Digital Certificate Signing, CN=StartCom Class 1 Primary Intermediate Server CA |
| | 5468063 | 2014-11-03 | 2014-10-31 | 2015-11-01 | support.steppingintothelightministry.org | jeromebracely32@yahoo.com | C=IL, O=StartCom Ltd., OU=Secure Digital Certificate Signing, CN=StartCom Class 1 Primary Intermediate Server CA |
| | 5471551 | 2014-11-03 | 2014-10-30 | 2015-10-31 | www.whitebeardscottage.net | massengilj2@yahoo.com | C=IL, O=StartCom Ltd., OU=Secure Digital Certificate Signing, CN=StartCom Class 1 Primary Intermediate Server CA |
| | 5461611 | 2014-11-02 | 2014-10-30 | 2015-10-30 | www2.wpwebsoft.com | rakesh.mohanta@yahoo.com | C=IL, O=StartCom Ltd., OU=Secure Digital Certificate Signing, CN=StartCom Class 1 Primary Intermediate Server CA |
| | 5455686 | 2014-11-02 | 2014-10-07 | 2015-10-08 | dev.opticity.com | adamsmithline@yahoo.com | C=IL, O=StartCom Ltd., OU=Secure Digital Certificate Signing, CN=StartCom Class 1 Primary Intermediate Server CA |
| | 5449373 | 2014-11-02 | 2014-10-20 | 2015-10-21 | www.ziarko.com | ziarko@yahoo.com | C=IL, O=StartCom Ltd., OU=Secure Digital Certificate Signing, CN=StartCom Class 1 Primary Intermediate Server CA |
| | 5442245 | 2014-11-01 | 2014-10-30 | 2015-10-30 | publish.yahoo.com | publish.yahoo.com | C=US, O="VeriSign, Inc.", OU=VeriSign Trust Network, OU=Terms of use at https://www.verisign.com/rpa (c)10, CN=VeriSign Class 3 Secure Server CA - G3 |
| | 5516882 | 2014-11-01 | 2013-11-15 | 2014-11-20 | stage.hk.admin.deals.yahoo.com | stage.hk.admin.deals.yahoo.com | C=US, O=DigiCert Inc, OU=www.digicert.com, CN=DigiCert High Assurance CA-3 |
| | 5440554 | 2014-11-01 | 2014-10-30 | 2015-10-30 | *.secure.in.webhosting.yahoo.com | *.secure.in.webhosting.yahoo.com | C=US, O="VeriSign, Inc.", OU=VeriSign Trust Network, OU=Terms of use at https://www.verisign.com/rpa (c)10, CN=VeriSign Class 3 Secure Server CA - G3 |
| | 5719344 | 2014-10-31 | 2013-05-28 | 2015-06-03 | tw.cst.serviceplus.yahoo.com | tw.cst.serviceplus.yahoo.com | C=US, O=DigiCert Inc, OU=www.digicert.com, CN=DigiCert High Assurance CA-3 |
| | 5423879 | 2014-10-31 | 2014-10-28 | 2015-10-29 | www.medisoftonline.in | medisoft1@yahoo.com | C=IL, O=StartCom Ltd., OU=Secure Digital Certificate Signing, CN=StartCom Class 1 Primary Intermediate Server CA |
| | 5415083 | 2014-10-30 | 2014-04-09 | 2015-04-09 | tw.edit.listing.yahoo.com | tw.edit.listing.yahoo.com | C=US, O="VeriSign, Inc.", OU=VeriSign Trust Network, OU=Terms of use at https://www.verisign.com/rpa (c)10, CN=VeriSign Class 3 Secure Server CA - G3 |

## Sublist3r

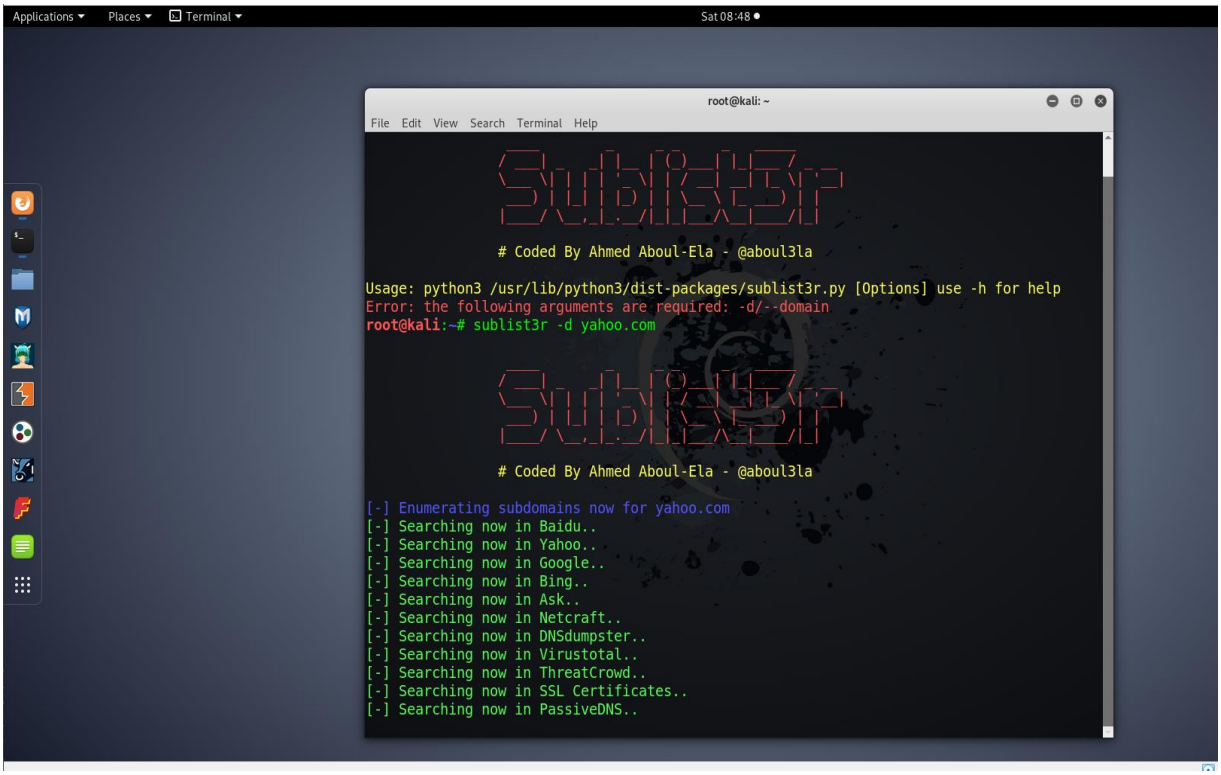
According to the description given in GitHub, **Sublist3r** is a python tool designed to enumerate subdomains of websites using OSINT. It helps penetration testers and bug hunters collect and gather subdomains for the domain they are targeting. Sublist3r enumerates subdomains using many search engines such as Google, Yahoo, Bing, Baidu and Ask. Sublist3r also enumerates subdomains using Netcraft, Virustotal, ThreatCrowd, DNSdumpster and ReverseDNS. [3]

In this audit **sublis3r** was used to discover subdomains of yahoo.com. Anyone can use this sublist3r by aboul3la using the git clone,

```
git clone https://github.com/aboul3la/Sublist3r.git
```

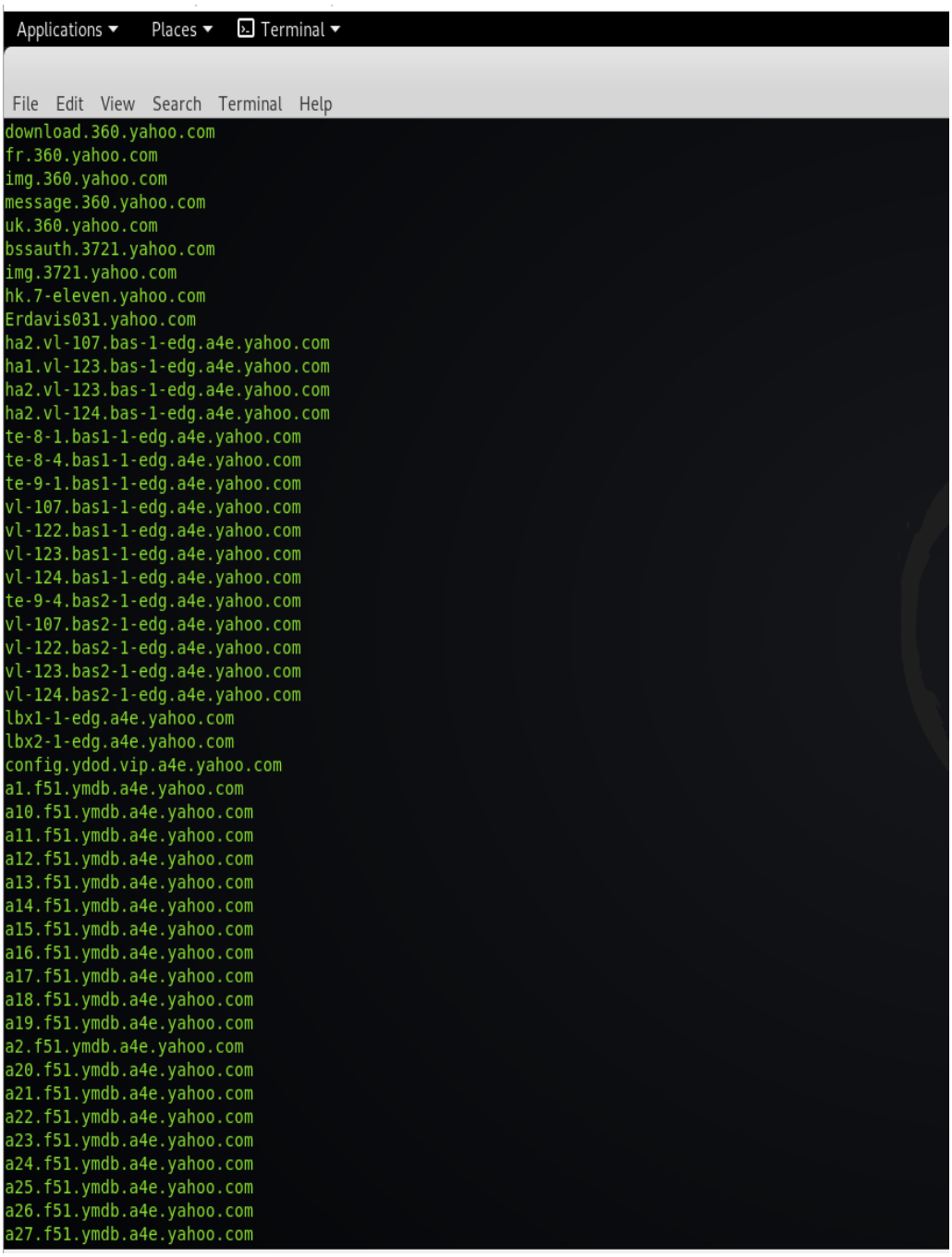
or,

now it is available on kali Linux apt repos. To install sublist3r using apt repos we must use, **apt install sublist3r** command.



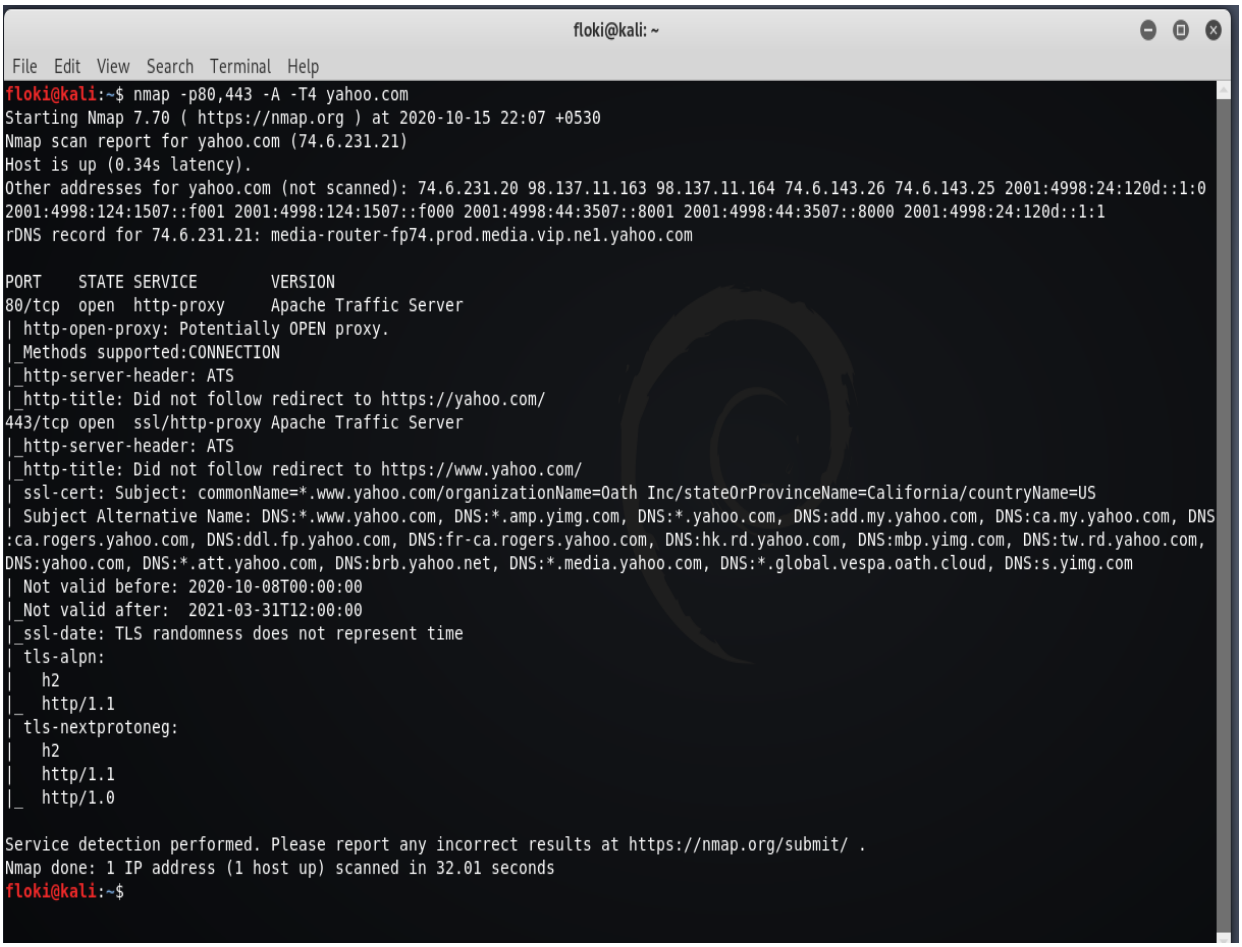
If the output of the sublist3r tool is directed to a text file, it will be much easier to identify the subdomains clearly.

Sample results from sublist3r on yahoo.com:



## Nmap scan

**Network-mapper** or **Nmap** is an open source utility that is widely used in network scanning and web auditing. Nmap is able to perform OS detections, server type detections and finding open ports which will give a security researcher performing a web audit a broader image of the infrastructure of a web application.




The above screenshot shows the results of the Nmap scan followed by,
**nmap -p80,443 -A -T4 yahoo.com** command. In this command ports 80 and 443 stands for http and https scripts respectively and -A means an active scan including OS detection, version detection, script scanning and traceroute. -T is the timing template and yahoo.com is the target that we need to scan. When analyzing this Nmap result, it identified the server used in web application which is an **Apache Traffic Server**.

## 2. Threat modeling and vulnerability identification

Scanning is the phase where that domains and target IP addresses will be scanned using various methods in order to discover vulnerabilities, loopholes and bad practices in the web application. Therefore, this is considered as the phase that gathers all the results that is needed in the exploitation because this is the phase that the vulnerabilities are gathered and analyzed so that later they can be used in exploiting the web application.

To begin the scan a security researcher must go through the information that was gathered during the information gathering phase. By analyzing scope and the rules of the Verizon media bug bounty program I selected few subdomains to scan for vulnerabilities and bad practices. To perform these scans, **Nikto**, **Netsparker**, **Portswigger's Burp suite** and **OWASP ZAP** tools are used.

Selected subdomains:

- **data.mail.yahoo.com**
- **mail.yahoo.com**
- **login.yahoo.com**
- **sports.yahoo.com**
- **finance.yahoo.com**
- **twpay.buy.yahoo.com**
- le.yahooapis.com
- onepush.query.yahoo.com
- proddata.xobni.yahoo.com

These subdomains were selected randomly based on considering their importance and functionalities. In this web audit, a full scanning for vulnerabilities covering all sub domains of yahoo.com will not be included because of the limited time given.

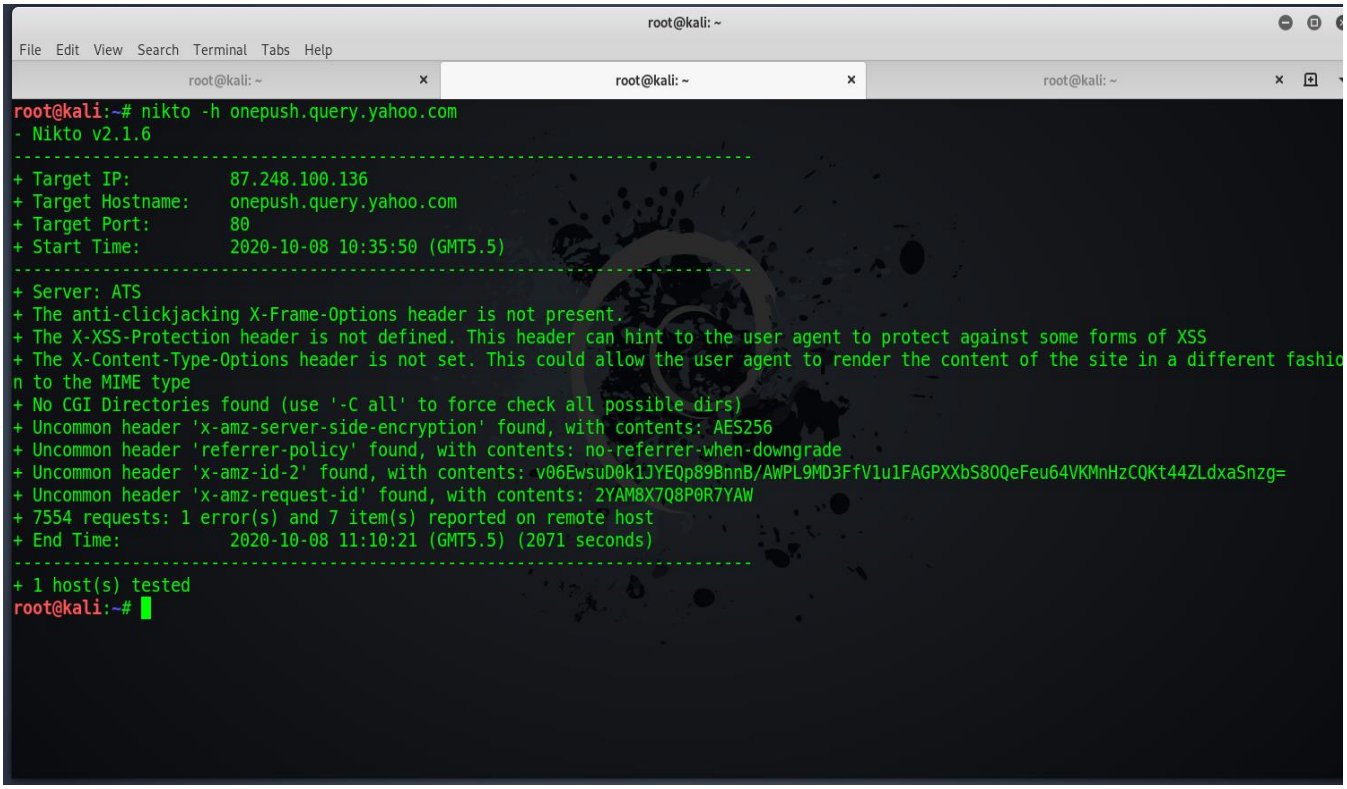
## Nikto

**Nikto** is an open source command line vulnerability scanner that is used in Linux platforms that scans web servers for dangerous files/CGIs, outdated server software and other problems. It performs generic and server type specific checks. It also captures and prints any cookies received. Nikto is able to detect potentially dangerous files and CGIs. In addition it can check server configuration items, index files and http header options. Therefore, Nikto is a very valuable tool when performing a web audit.

But most of the occasions Nikto generates too much traffic towards the server so that server can identify it as a denial of service attack and block the researcher's ip address form making requests to the server.

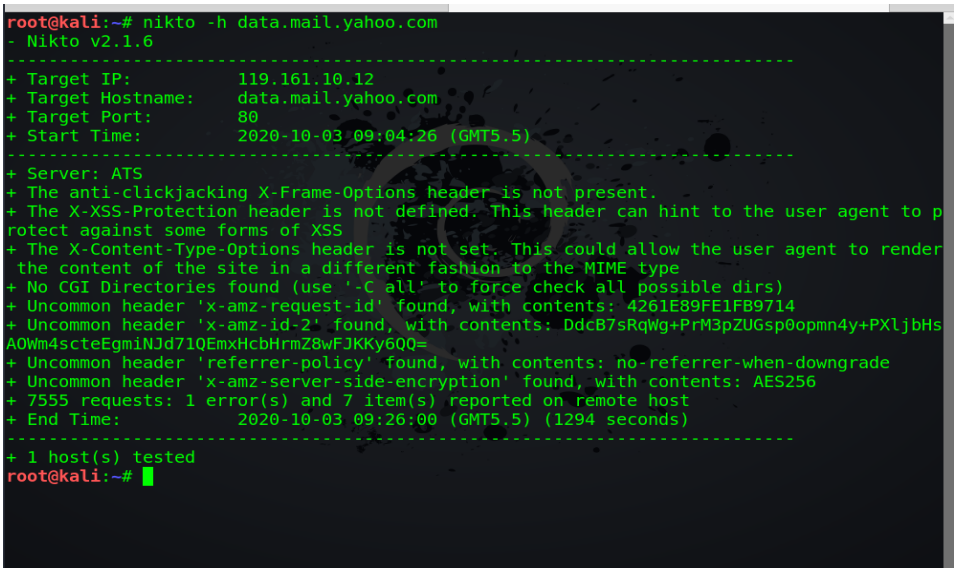
Command that was used to scan using Nikto:

nikto -h subdomain name

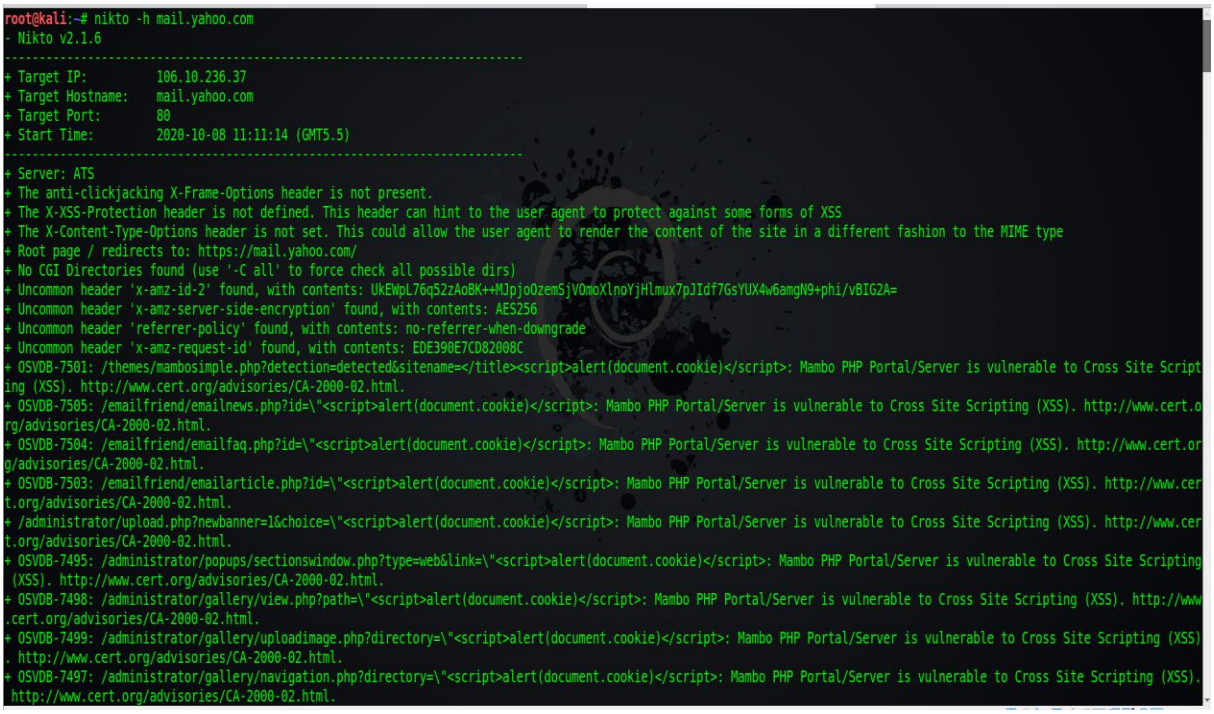


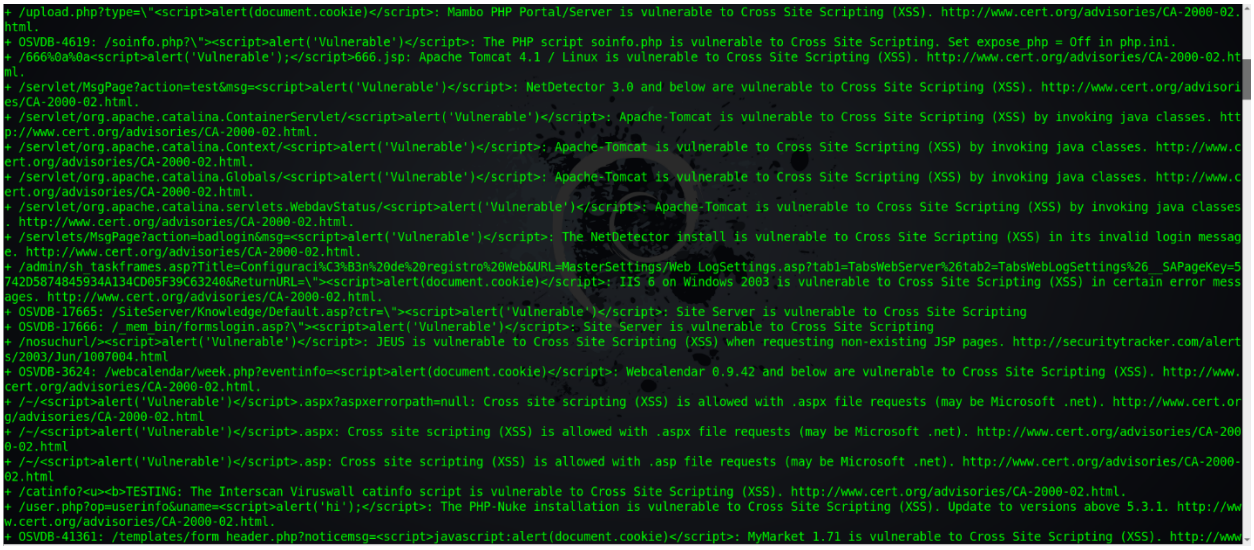
## Results

- **data.mail.yahoo.com**




- **mail.yahoo.com**

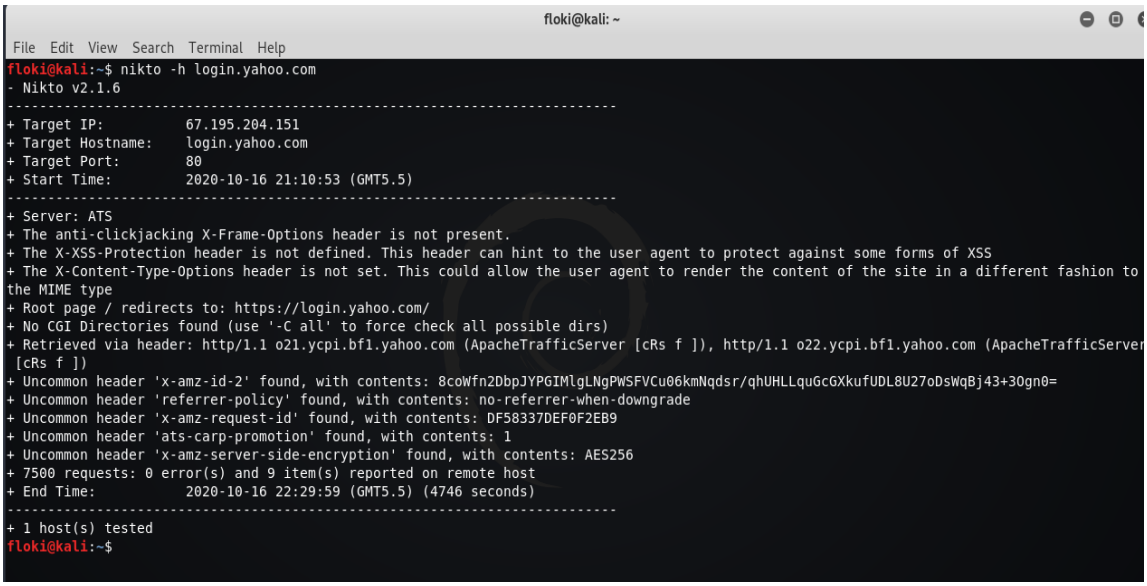



+ /upload.php?type=\"<script>alert(document.cookie)</script>: Mambo PHP Portal/Server is vulnerable to Cross Site Scripting (XSS). http://www.cert.org/advisories/CA-2000-02.html.
+ OSVDB-4619: /soinfo.php?\"><script>alert('Vulnerable')</script>: The PHP script soinfo.php is vulnerable to Cross Site Scripting. Set expose_php = Off in php.ini.
+ /666%0a%0a<script>alert('Vulnerable');</script>666.jsp: Apache Tomcat 4.1 / Linux is vulnerable to Cross Site Scripting (XSS). http://www.cert.org/advisories/CA-2000-02.html.
+ /servlet/MsgPage?action=test&msg=<script>alert('Vulnerable')</script>: NetDetector 3.0 and below are vulnerable to Cross Site Scripting (XSS). http://www.cert.org/advisories/CA-2000-02.html.
+ /servlet/org.apache.catalina.ContainerServlet/<script>alert('Vulnerable')</script>: Apache-Tomcat is vulnerable to Cross Site Scripting (XSS) by invoking java classes. http://www.cert.org/advisories/CA-2000-02.html.
+ /servlet/org.apache.catalina.Context/<script>alert('Vulnerable')</script>: Apache-Tomcat is vulnerable to Cross Site Scripting (XSS) by invoking java classes. http://www.cert.org/advisories/CA-2000-02.html.
+ /servlet/org.apache.catalina.Globals/<script>alert('Vulnerable')</script>: Apache-Tomcat is vulnerable to Cross Site Scripting (XSS) by invoking java classes. http://www.cert.org/advisories/CA-2000-02.html.
+ /servlet/org.apache.catalina.servlets.WebdavStatus/<script>alert('Vulnerable')</script>: Apache-Tomcat is vulnerable to Cross Site Scripting (XSS) by invoking java classes. http://www.cert.org/advisories/CA-2000-02.html.
+ /servlets/MsgPage?action=badlogin&msg=<script>alert('Vulnerable')</script>: The NetDetector install is vulnerable to Cross Site Scripting (XSS) in its invalid login message. http://www.cert.org/advisories/CA-2000-02.html.
+ /admin/sh_taskframes.asp?Title=Configuraci%C3%B3n%20de%20registro%20Web&URL=MasterSettings/Web_LogSettings.asp?tab1=TabsWebServer%26tab2=TabsWebLogSettings%26_SAPageKey=5742D5874845934A134CD05F39C63240&ReturnURL=\"><script>alert(document.cookie)</script>: IIS 6 on Windows 2003 is vulnerable to Cross Site Scripting (XSS) in certain error messages. http://www.cert.org/advisories/CA-2000-02.html.
+ OSVDB-17665: /SiteServer/Knowledge/Default.asp?ctr=\"><script>alert('Vulnerable')</script>: Site Server is vulnerable to Cross Site Scripting
+ OSVDB-17666: /_mem_bin/formslogin.asp?\"><script>alert('Vulnerable')</script>: Site Server is vulnerable to Cross Site Scripting
+ /nosuchurl/><script>alert('Vulnerable')</script>: JEUS is vulnerable to Cross Site Scripting (XSS) when requesting non-existing JSP pages. http://securitytracker.com/alerts/2003/Jun/1007004.html
+ OSVDB-3624: /webcalendar/week.php?eventinfo=<script>alert(document.cookie)</script>: Webcalendar 0.9.42 and below are vulnerable to Cross Site Scripting (XSS). http://www.cert.org/advisories/CA-2000-02.html.
+ /~/<script>alert('Vulnerable')</script>.aspx?aspxerrorpath=null: Cross site scripting (XSS) is allowed with .aspx file requests (may be Microsoft .net). http://www.cert.org/advisories/CA-2000-02.html
+ /~/<script>alert('Vulnerable')</script>.aspx: Cross site scripting (XSS) is allowed with .aspx file requests (may be Microsoft .net). http://www.cert.org/advisories/CA-2000-02.html
+ /~/<script>alert('Vulnerable')</script>.asp: Cross site scripting (XSS) is allowed with .asp file requests (may be Microsoft .net). http://www.cert.org/advisories/CA-2000-02.html
+ /catinfo?<u><b>TESTING: The Interscan Viruswall catinfo script is vulnerable to Cross Site Scripting (XSS). http://www.cert.org/advisories/CA-2000-02.html.
+ /user.php?op=userinfo&uname=<script>alert('hi');</script>: The PHP-Nuke installation is vulnerable to Cross Site Scripting (XSS). Update to versions above 5.3.1. http://www.cert.org/advisories/CA-2000-02.html.
+ OSVDB-41361: /templates/form_header.php?noticemsg=<script>javascript:alert(document.cookie)</script>: MyMarket 1.71 is vulnerable to Cross Site Scripting (XSS). http://www

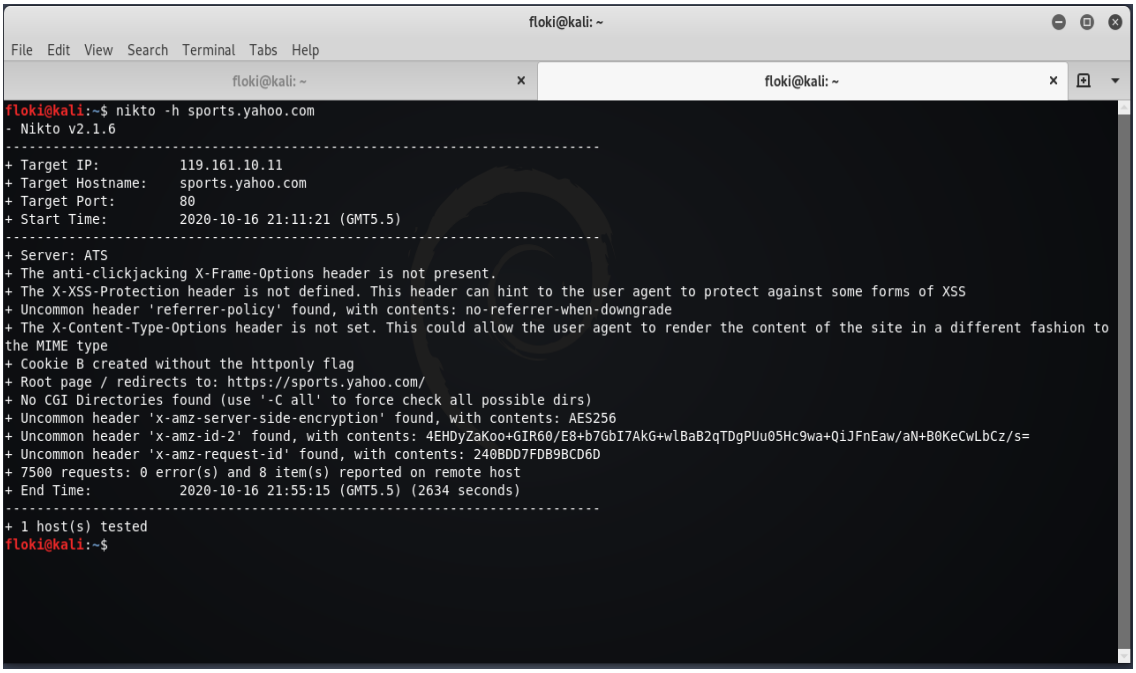

+ OSVDB-5049: /setup.exe?<script>alert('Vulnerable')</script>&page=list_users&user=P: CiscoSecure ACS v3.0(1) Build 40 allows Cross Site Scripting (XSS). http://www.cert.org/advisories/CA-2000-02.html.
+ OSVDB-2689: /servlet/ContentServer?pagename=<script>alert('Vulnerable')</script>: Open Market Inc. ContentServer is vulnerable to Cross Site Scripting (XSS) in the login-error page. http://www.cert.org/advisories/CA-2000-02.html.
+ OSVDB-2322: /search.php?searchstring=<script>alert(document.cookie)</script>: Gallery 1.3.4 and below is vulnerable to Cross Site Scripting (XSS). Upgrade to the latest version. http://www.securityfocus.com/bid/8288.
+ OSVDB-50551: /search.php?searchfor=\"><script>alert(1776)</script>: Siteframe 2.2.4 is vulnerable to Cross Site Scripting (XSS). http://www.cert.org/advisories/CA-2000-02.html.
+ /search.asp?term=<%00script>alert('Vulnerable')</script>: ASP.Net 1.1 may allow Cross Site Scripting (XSS) in error pages (only some browsers will render this). http://www.cert.org/advisories/CA-2000-02.html.
+ /samples/search.dll?query=<script>alert(document.cookie)</script>&logic=AND: Sambar Server default script is vulnerable to Cross Site Scripting (XSS). http://www.cert.org/advisories/CA-2000-02.html.
+ /replymsg.php?send=1&destin=<script>alert('Vulnerable')</script>: This version of PHP Nuke's replymsg.php is vulnerable to Cross Site Scripting (XSS). http://www.cert.org/advisories/CA-2000-02.html.
+ /postnuke/modules.php?op=modload&name=Web_Links&file=index&req=viewlinkdetails&lid=666&ttitle=Mocosoft Utilities\"%3<script>alert('Vulnerable')</script>: Postnuke Phoenix 0.7.2.3 is vulnerable to Cross Site Scripting (XSS). http://www.cert.org/advisories/CA-2000-02.html.
+ OSVDB-4599: /pm_buddy_list.asp?name=A&desc=B%22%3E<script>alert('Vulnerable')</script>%3Ca%20s=%22&code=1: Web Wiz Forums ver. 7.01 and below is vulnerable to Cross Site Scripting (XSS). http://www.cert.org/advisories/CA-2000-02.html.
+ /phpwebsite/index.php?module=search&SEA_search_op=continue&PDA_limit=10\"><script>alert('Vulnerable')</script>: phpWebSite 0.9.x and below are vulnerable to Cross Site Scripting (XSS). http://www.cert.org/advisories/CA-2000-02.html.
+ /phpwebsite/index.php?module=pagemaster&PAGE_user_op=view_page&PAGE_id=10\"><script>alert('Vulnerable')</script>&MMN_position=[X:X]: phpWebSite 0.9.x and below are vulnerable to Cross Site Scripting (XSS). http://www.cert.org/advisories/CA-2000-02.html.
+ /phpwebsite/index.php?module=fatcat&fatcat[user]=viewCategory&fatcat_id=1%00+\"><script>alert('Vulnerable')</script>: phpWebSite 0.9.x and below are vulnerable to Cross Site Scripting (XSS). http://www.cert.org/advisories/CA-2000-02.html.
+ /phpwebsite/index.php?module=calendar&calendar[view]=day&month=2&year=2003&day=1+%00\"><script>alert('Vulnerable')</script>: phpWebSite 0.9.x and below are vulnerable to Cross Site Scripting (XSS). http://www.cert.org/advisories/CA-2000-02.html.
+ OSVDB-59093: /phptonuke.php?filnavn=<script>alert('Vulnerable')</script>: PHP-Nuke add-on PHPToNuke is vulnerable to Cross Site Scripting (XSS). http://www.cert.org/advisories/CA-2000-02.html.
+ OSVDB-32774: /phpinfo.php?VARIABLE=<script>alert('Vulnerable')</script>: Contains PHP configuration information and is vulnerable to Cross Site Scripting (XSS).
+ OSVDB-32774: /phpinfo.php3?VARIABLE=<script>alert('Vulnerable')</script>: Contains PHP configuration information and is vulnerable to Cross Site Scripting (XSS).
+ OSVDB-2193: /phpBB/viewtopic.php?topic_id=<script>alert('Vulnerable')</script>: phpBB is vulnerable to Cross Site Scripting (XSS). Upgrade to the latest version. http://www.cert.org/advisories/CA-2000-02.html.
+ OSVDB-4297: /phpBB/viewtopic.php?t=17071&highlight=\">\"<script>javascript:alert(document.cookie)</script>: phpBB is vulnerable to Cross Site Scripting (XSS). http://www.cert.org/advisories/CA-2000-02.html.
+ OSVDB-11145: /phorum/admin/header.php?GLOBALS[message]=<script>alert('Vulnerable')</script>: Phorum 3.3.2a and below from phorum.org is vulnerable to Cross Site Scripting (XSS). http://www.cert.org/advisories/CA-2000-02.html.


0-02.html.
+ OSVDB-701: /pls/help/<script>alert('Vulnerable')</script>: Oracle 9iAS is vulnerable to Cross Site Scripting (XSS). http://www.cert.org/advisories/CA-2000-02.html.
+ OSVDB-20954: /shopadmin.asp?Password=abc&UserName=\"><script>alert(foo)</script>: VP-ASP Shopping Cart 5.50 shopadmin.asp UserName Variable XSS.
+ OSVDB-20406: /phpinfo.php?GLOBALS[test]=<script>alert(document.cookie);</script>: PHP contains a flaw that allows a remote cross site scripting attack.
+ OSVDB-24484: /phpinfo.php?cx[]=yGNXZ35aDv38InU0ik6GnLFXuLjwb6aTqgaQXCInoW1Ux9yg5mcWu736yizhyI7OPGe7q716eLBN3CMFjaT9hEcPpL5V3fDYBwdWgf99ILXjNC1trdoIGndpvykbtJf475k1r7Fxjif
87LT7vzWEh1SIBvKHbkWLKFOfAX65maXcX3R43eYXFsxp8zZ03H94IkWjUNON8ac8bzKXZH4coKpVfknbuL2SK9HhB8GyD09xtKCs3bdtppx7JXXLdgl2krI3IZvKVR1lluvmyF58PzklQvLdYzeiIqzH4DWErUwKgVjLDulhwMCi
b0kcDo8IwOE61LGgMrjABAAihslfi9VpCoWLYiK7aT04k7QPFoaSBQ1woE3WjYccp17J0wXbNSpgnWpyoqUHWKQkCLx5pKYZ5eQS1vWE7DjGtXX8nq2ey0Nup8vlxKYJObZm4S7ZNPePGcrkkA2kYn9bGjwpZI06oWsUt5njzqQqB
mhklTzSQkfNLkgNVosLJRwkrp3H0qvXGXlnJb73328vWrmTy3VR132KLNeI8cEEuYqYLvCf0xykgJqnJSBEhkC9QUNvtWslArBp5jGHuteR3Xz87nJxJ1NAy6OW4HnEktWYaKz7HKuour6vIwL1oxdRxwEjVx6b76dgkHNOWSYY3b
lRa6G8FDFHlepapkf1RFXlMdHOlmWz0S5NdGVQ4jhvYqcL3FwcF6r6ivpykrviGjno0OzVQLqAixusplzvLC59TfRz2duPTDJaAP4IWCVTNIwQ9YmbNhL9bBfHE0kI4YJhdUfrJpWmhYSQjeg3a9vHCApiAXo0yR9aRNSn4h0BVHHtRcxyUa9vuWLZo9rDB7kruBuBdTSlaZuDdkH3uCl9SwetVGElDyCvCoJyZJDb3frwhvbhenyd8scITqw911qbAX2aMnpYnB84iKTSAMl4CaekLLvrkmk298PWXgkHiy70h0143cIIJPip8RvDSFOQHqWoTxmTICA1hxOD2aHAi3Zi
tRcxyUa9vuWLZo9rDB7kruBuBdTSlaZuDdkH3uCl9SwetVGElDyCvCoJyZJDb3frwhvbhenyd8scITqw911qbAX2aMnpYnB84iKTSAMl4CaekLLvrkmk298PWXgkHiy70h0143cIIJPip8RvDSFOQHqWoTxmTICA1hxOD2aHAi3Zi
o21nTvaHRgGzyg5u910t7vEK9q4kg7B2LGks4lQlRo4WFFvmplVa62YfIncuDRwHOa35q01WW3JiqUR7EuGNaxuSws4cSDYhho10PpLhOcW4NNcNoiciTxlHIWzMvtADbhmZN89jbpmCk0iCOEO23EAIhHGn3XBZzWWZOse61bP0G
h8KWgaQ6cqzkhaphnXfjO2TZ8Qfex1Hs6H4hM4KYUXn045aTzN8CijSqCaxTc2GVqtXlJuK2bCNrONce575CE9az8ldU4LGXPadlFqhfC395XKLK1GOyb3Hy5ImldrCTD9L5j0xvZQJ8BQ8g38PcBF2so41WiOfFA7C0LuC9rufse
B25mjvP1J7q3Dlzhn4IusF7GnCKdnptWHOVCOagQClUlZsLLxlhkVJuFphYBZEAAAJAQQHKjTzEILs5zCtqlTimySSHnZ3CmjbApp66GZsjOoh3KYJlvHXzTQFumqGiHfpMsP7Hd0CRLDTlWYL9k0cpz5yvm2iTbV7wZOFfQ6MKbEK
exuWcHrHj9toES7Ar9GxS7YqDGndWIkK2kzI7KLJnmlv4yhmaLB9HTRRAR5dsUoOELea5AWfn8LQ3MLhngNooWDrPuVnQO4lVgGBpQC0KxcwPNgyoR3qC6aPoSovPMnbJTWRbna1Gdh4QdCXfdgYqtzBjfHm0T0uZIXKkUJA8CoWj
BdGVSzwVHkUrFM3IJXjzq26EXIH3MIUaoABie5sdGQAFCLxMDuBcMO2hCf4lGOFaPskHvlx4tsgmzztCm6q1dAX3ycq6iMeEf3zc8tlJD204m0f9eEKlg4sxSbuHm3BdqVskjT9uIXNryWw4Fxg0cFgKoUf4stl7eTbJWVCtjEyAv
3KPgbXSsWVxyKSKdidKmIkfAN049yerzzgd8iy4hQxNPASezpNwpGPJVBfEJKaKSCuqRyLc83kdNN44flk9R2wplGnU9GBSgyvLOgzlpY8XDCAfiCi9Iib7MVYvhXznQWLZ2b09PZRJmzOoJUjJMnnUOtYjfXNn1un69lbw6gIYsl
q3vpT3DCDu4olXjB3ChAlQm9QCpSzkhG61fclJmLx9RMbYS0wqFSkGjtZfZuXaIIGHxDyMTB4zZHQm37wgRpAuOX7DKkdzOnIpZM3K9CR0WHivivogQLeEAeEGLEpczaBLD2OkEvPhh6wJCMFiuQJVWBPuRBfHD153kzlbbtwoaNf
8Rvm5xCOKqrJphBM3fDjIv3HgKtYMrXF1tzAK7qpbg0Zh8Zbl8MlDEdvdvek6zetItZktypgq1bsp3fjtx43JoG4kgeMVfFRMTnVfFqo0RuDwNwcpeIGjjDwJKspJOlXUkD55h3di3Z0wOIumEUCEbHYYp8oUoH9XZrKxj3GYeD9e
NOPi9VUq5NJMa8Z2SlJLTTnA21SGHjQXpUZEc12iR5spytVavVE4D06KWwLFxqN3Jh9fI5jnASA2Puzqjeqj8ilNKenzTctc2lQmvQBTbKsMyYAwjjctYfgrYG7eTuxMdy35EaHEavPI28gJq46GWXccM6wHQik1zaUP8FY0zq4Ci
vhC8Gs3TuDBgH581L13h4K5krHjUkLBZvmiYtwEwuiz34E1L1SScnwjJgGFnV0wAccvBP34Yzhhn7Mb8hZhi5F9DItw1UXmNvg9Vim6rT0Xp5KeU1ctqusJ0QTrfvj99GXDDUabyxkg73sK6gGVRLvgbPmufGV9FkzEDCWSD1JLhz
YMPVuF4b6L04e4NFknZyIVtZdVlDjNcL2jMrMXbzt7Dm6xQlorjIqb4Y6D40XlsybUSf7rNpU8Kr67BWVLWfJeuVWdzo5EkmPfHEmww2V3vD6wtqv4De6duNe7s8mjXV4YBGlHH75UV4RkgZgwV4WAryhT4MwmNDNTIcX0zbeW03
oE2wzL1VyecMixSNI7mzByey3cDBhWQBraf58XTzYjvkHlq1sSb1j5pmYGDv2YgH8c1ALaDlVPTLZpbDQEMIzYCscfH8KjW1qx1udz07MZUiOmuxLPgLQ3UZ8HcwKUsW05eTB5biIkrIVV0m34RRUPtrsCg0f4LGyZ3mjhu6Uh5gi
VST2D8LRejjBWoItYMe8w6X9lLO0RJOYscze3agPRYZWITTINAisHqqeUsWbxMh54y9sskj0dEuZ3JmhjgX1Hbsg384TOxGbzVs20cX9sVexHjUTqIMq7ueJ6YDet5R6ijv3m9WOv1aCTSv8MSAKgPz5YmjDHtdBwSH8TwU8IcV6DM
6ugWTOZgizTB0CbWSZSbFR3ciqv9fzMWOhqlXMD7wAv0lwu8wlV9yqJslNrKj9zyQgPY3e8c4IurqDmIZEVJ9m5sCOYxp0iMAmRi1N7CKbeO46kYujptdCpSCHMitSBBlayC360iUU9Ugho6vyKQBauSYBDlXdVjksOoEo86P52j6
PUepnEruIMzEpS9FwCtcqAFXoWxpk1F4fs8KcOVvTrQFutqP6euU9Br7S4aAdEVCa5Y346RlkPIV3Fz87hq8BRi9vFzcCq1c7vlEPff9j0RCDeK22pGwRNQMkq0qgyuyXzyLrV5Zn5RxS8a0oYM54jzKy1DN6euvSLeDwltIedDbA
839UV6vnbj90arOZZSX20316gPKXp2CayLyVeCuKxgdN0JTbCBoFnXFmtANn9fabZIclvC0IOE7PlhcVi2E2fg7JI71xN3WphkzAgBLZuPdkZ4H91W6HhpBY8Y1IzV2VmSJmlOm6x3nh0N17v69CQDaB4w4gEaMKvhadUfbMzKI1b
KG7hzbAlfQhnLZ1DHsuaQ75DbpoEsgtgC2gaQLGTIhwgCmFbX7zuYFwzcNcV6lKsGyd8sjLSv9tnrsx1s1tlIdQ8IyzMcEpHLT4nt27ZH7UDMkv6qXsUrQfGXQ4We4RgLm6Tshe7gHjfFK6d9isWe5tkDGy3XgFZSYObqkyzfwcqI
mXAnXjtroFgMcX7WXRWj1Fsdjyhyvygttl5k1hkYbNstmhxW5aKRmd4INynBxssodZnZkOTIA9ILbf1EokcNYXoDv5DDZM5uncEOEAMIDu7ic2WcbfFGkwQeuUjVGfakuYjLXZTthbYjBKF0ndOE7v</script>alert(foo)</script>: PHP 5.1.2 and 4.4.2 phpinfo() Function Long Array XSS
+ OSVDB-35935: /rpc.php?q=\"><script>alert(document.cookie)</script>: Unobtrusive Ajax Star Rating Bar is vulnerable to XSS in the q variable.
+ OSVDB-34879: /jsp-examples/jsp2/jspx/textRotate.jspx?name=<script>alert(111)</script>: The tomcat demo files are installed, which are vulnerable to an XSS attack
+ OSVDB-34878: /jsp-examples/jsp2/el/implicit-objects.jsp?foo=<script>alert(112)</script>: The tomcat demo files are installed, which are vulnerable to an XSS attack
+ OSVDB-12721: /jsp-examples/jsp2/el/functions.jsp?foo=<script>alert(113)</script>: The Tomcat demo files are installed, which are vulnerable to an XSS attack
+ OSVDB-58463: /scripts/message/message_dialog.tml?how_many_back=\"><script>alert(1)</script>: Lyris ListManager Cross-Site Scripting.
+ 7499 requests: 0 error(s) and 223 item(s) reported on remote host
+ End Time:           2020-10-08 11:24:18 (GMT5.5) (784 seconds)

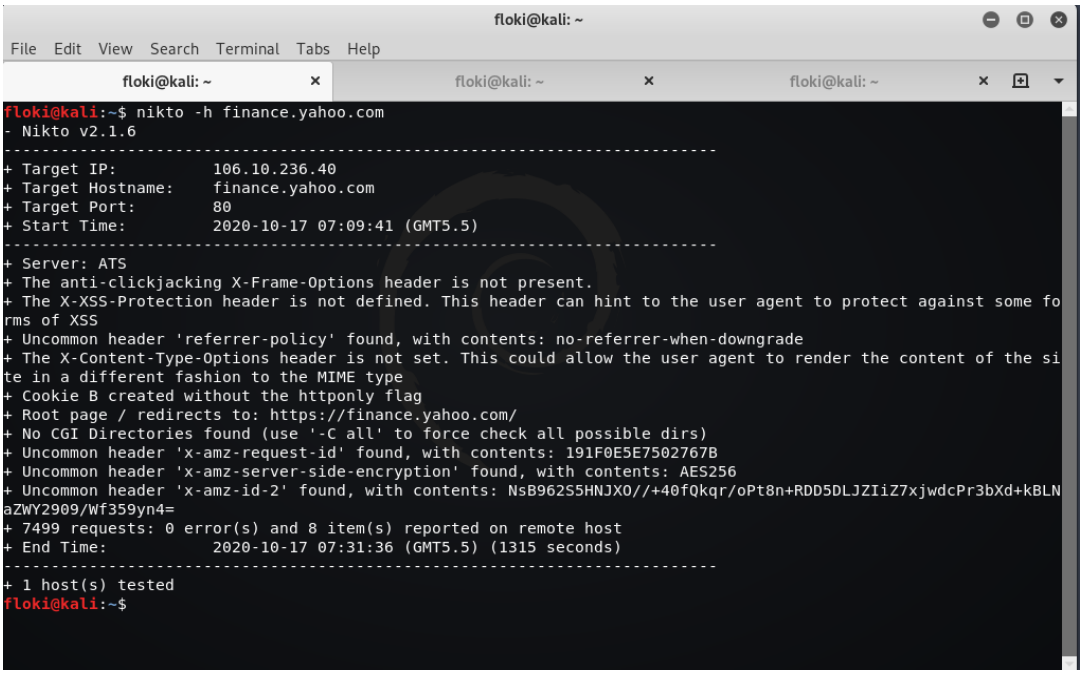
- **login.yahoo.com**



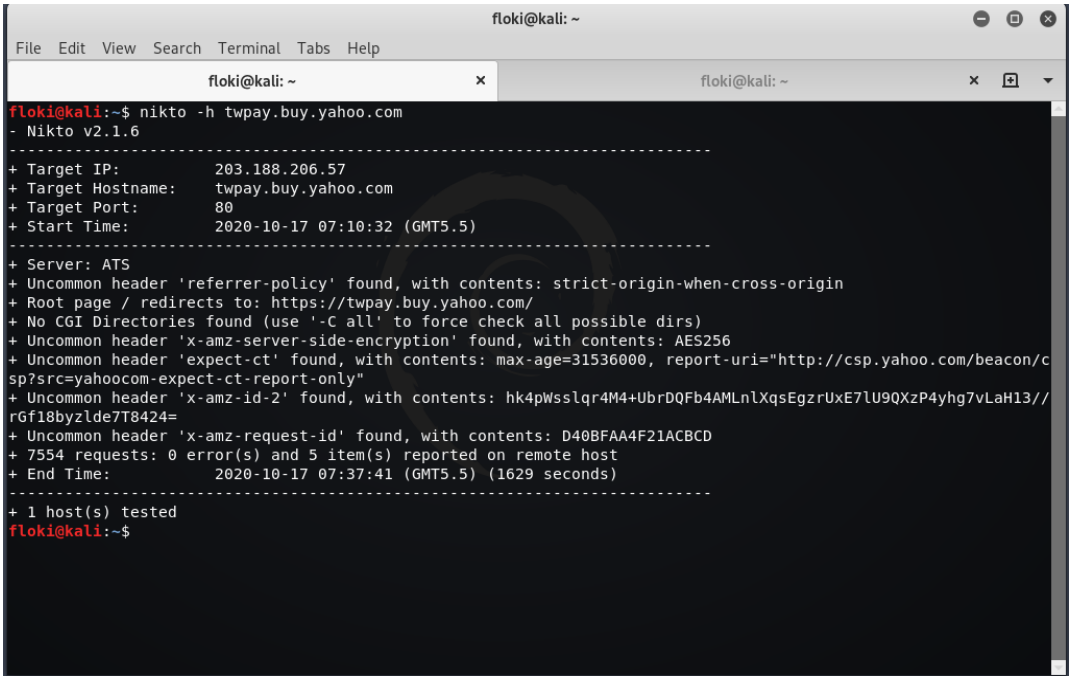

- **sports.yahoo.com**



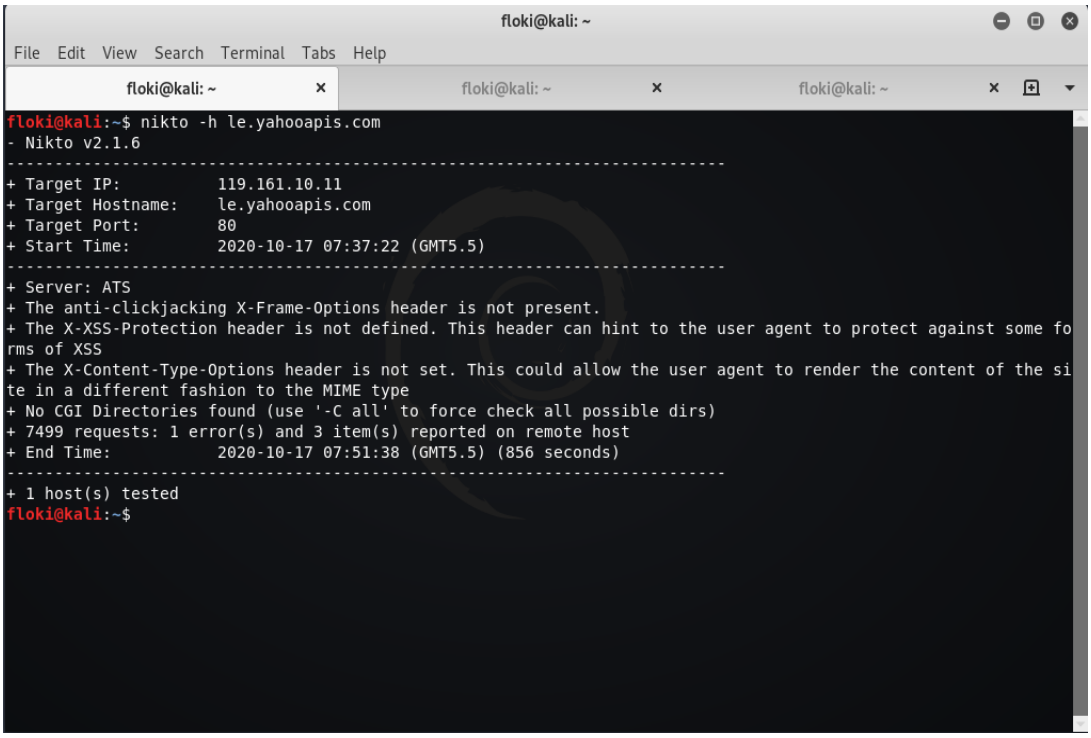
- **finance.yahoo.com**



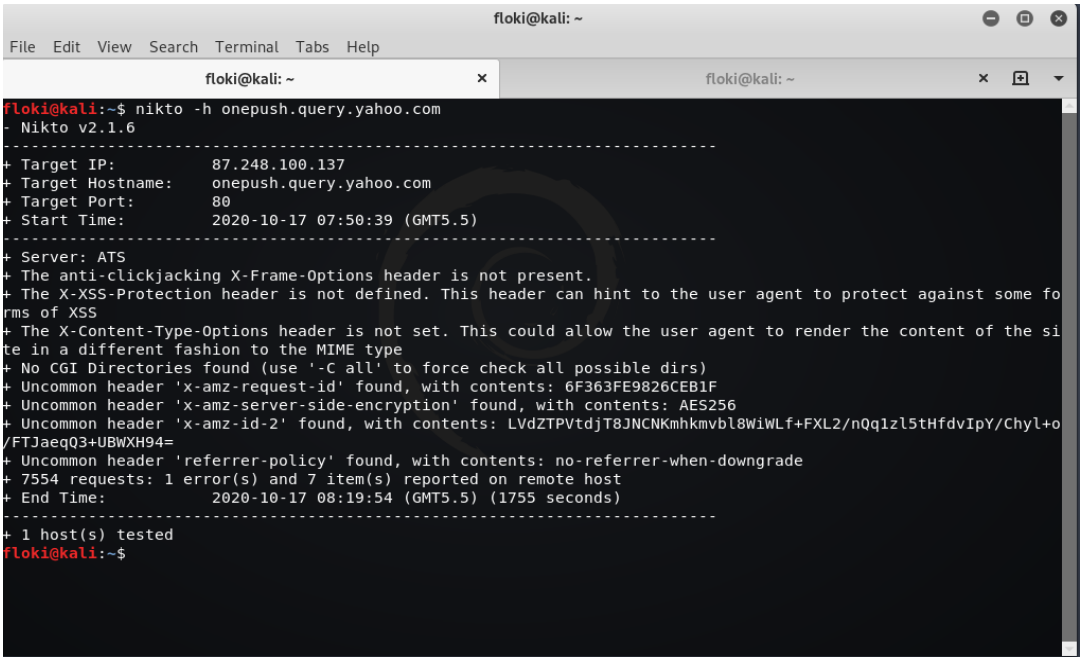

- **twpay.buy.yahoo.com**



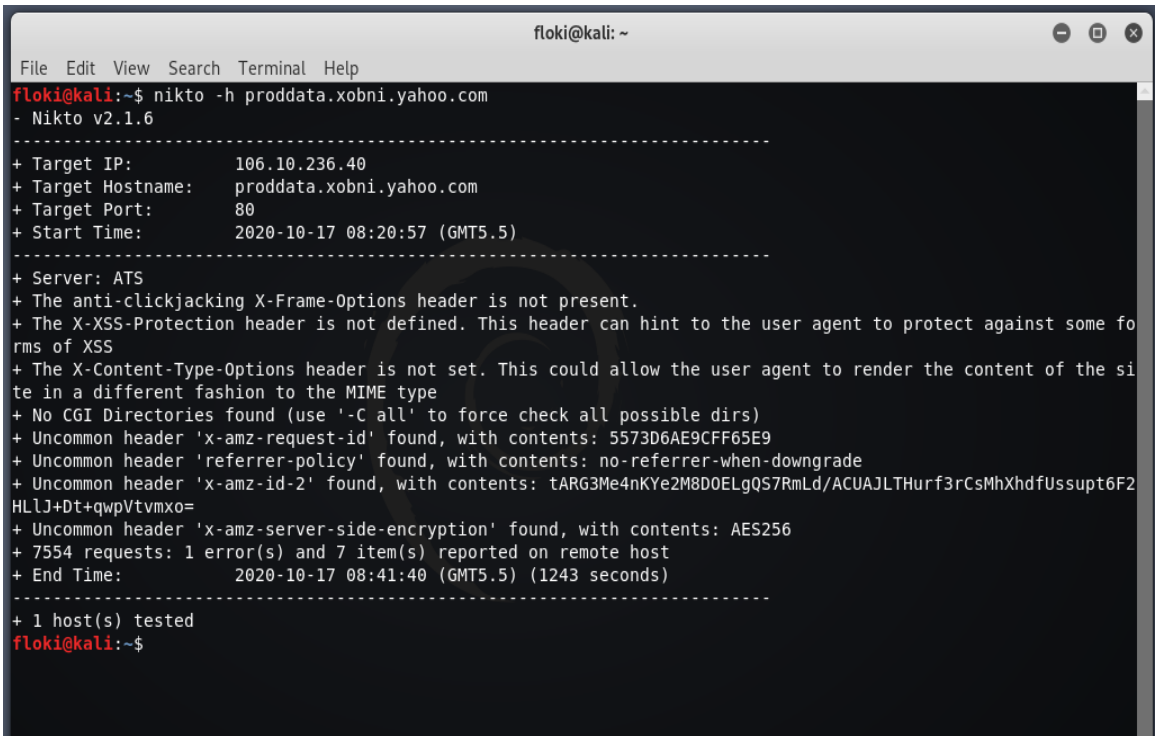
- **le.yahooapis.com**




- **onepush.query.yahoo.com**



- **proddata.xobni.yahoo.com**




**These results with the screenshots are included just for the process of describing methodology used in vulnerability identification and threat modeling. These results and the results from the other tools will be analyzed under the vulnerability analysis part.

## Netsparker

**Netsparker** is an automated web application vulnerability scanner that is often use by security researchers while conducting web security audits on web applications. Netsparker scanner has the ability to detect vulnerabilities including OWASP's top 10 vulnerabilities, logical flaws, proper usage of security libraries and not only that, it is able to confirm these vulnerabilities with an automatically generated reports and proof of exploits.

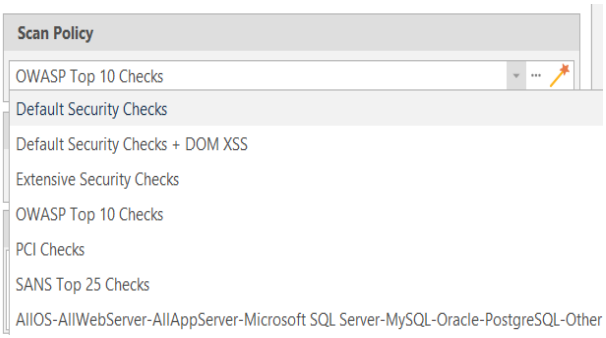
When using the Netsparker scans in this audit, I only checked OWASP's top 10 vulnerabilities against the yahoo.com web servers.

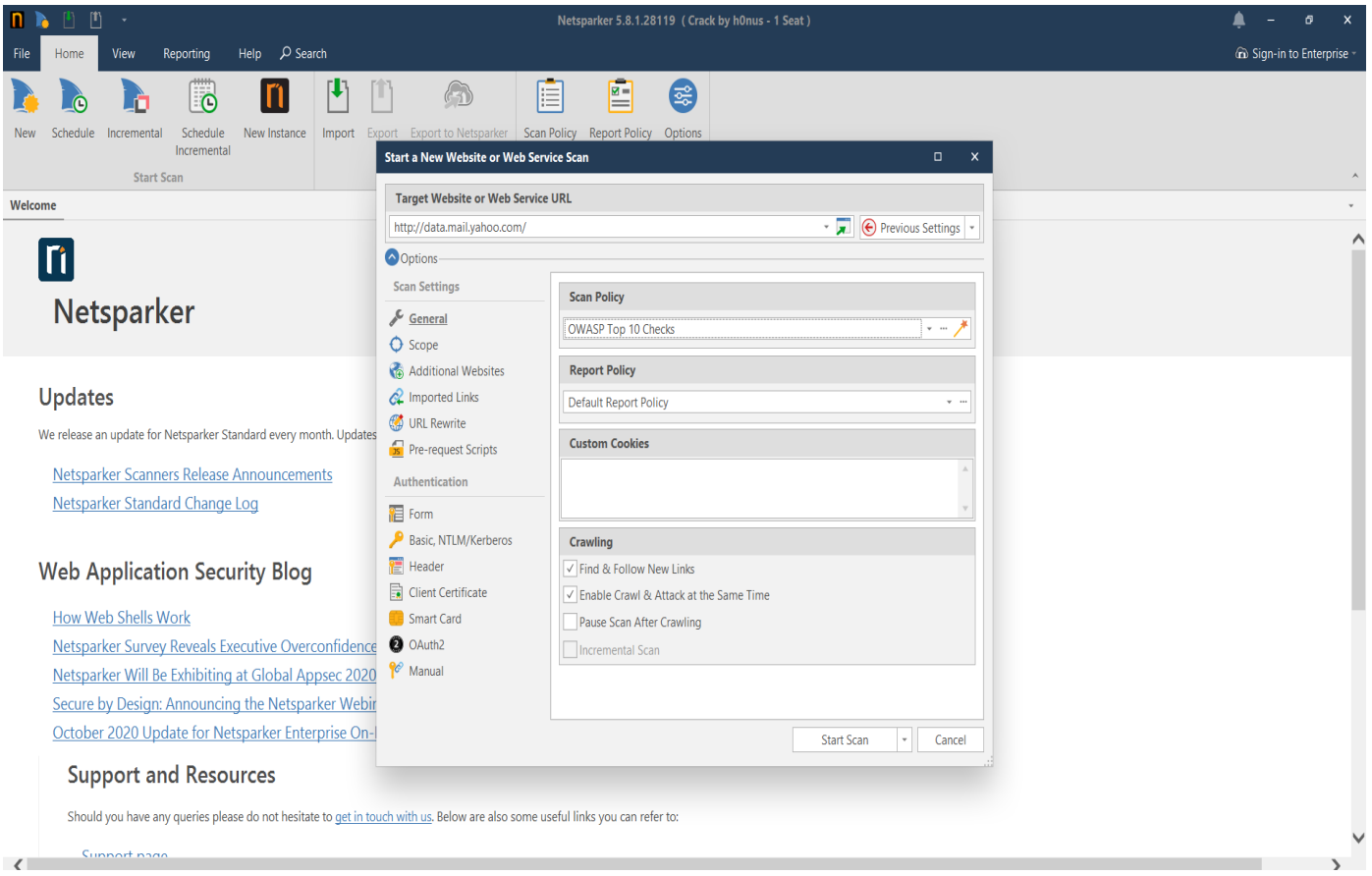
OWASP's top 10 vulnerabilities

- Injection
- Broken Authentication
- Sensitive Data Exposure
- XML External Entities (XXE)
- Broken Access control
- Security misconfigurations
- Cross Site Scripting (XSS)
- Insecure Deserialization
- Using Components with known vulnerabilities
- Insufficient logging and monitoring

**Initializing a vulnerability scan using Netsparker**

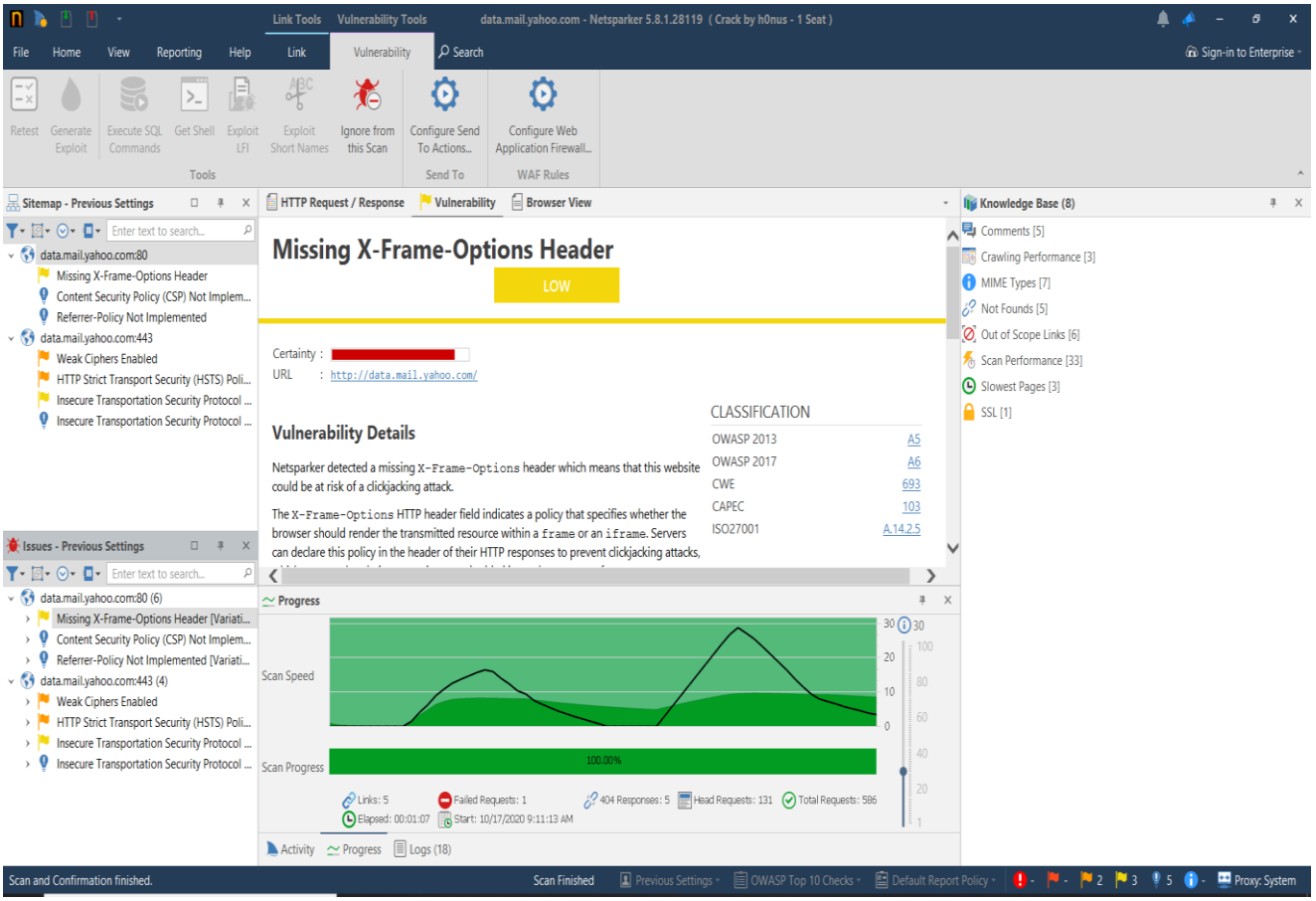
We must provide the target website's domain followed by the scan policy that we need to use in the vulnerability scan. For the scans in this web security audit, I chose OWASP top 10 checks scan policy because this audit is conducted around OWASP's top 10 vulnerabilities.



Screenshots of Netsparker scanning results are included below.

- **data.mail.yahoo.com**



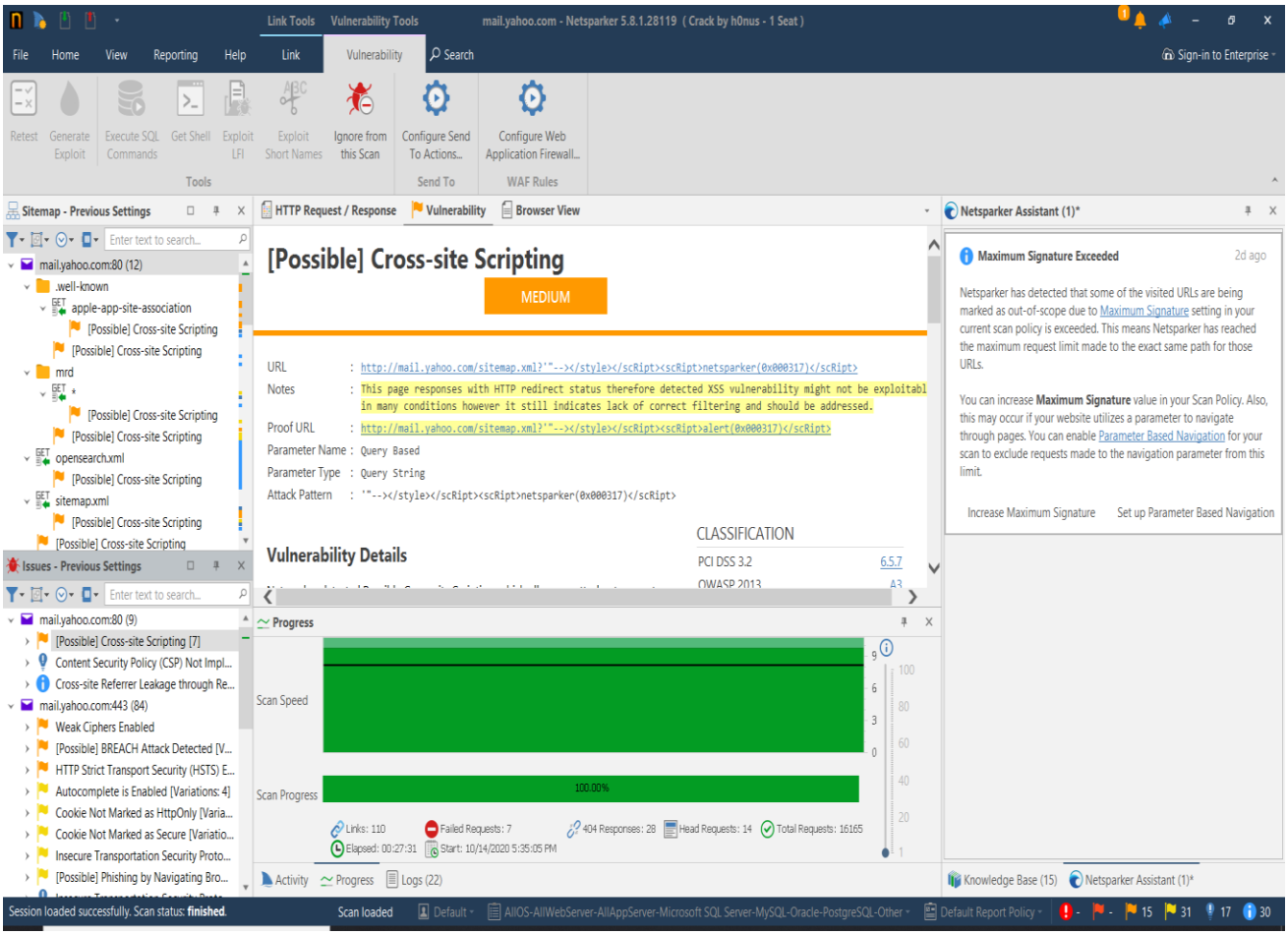

Total issues found: 10

**Medium**: 2

**Low**: 2

**Best practice issues:** 3

- **mail.yahoo.com**




Total issues found: 93

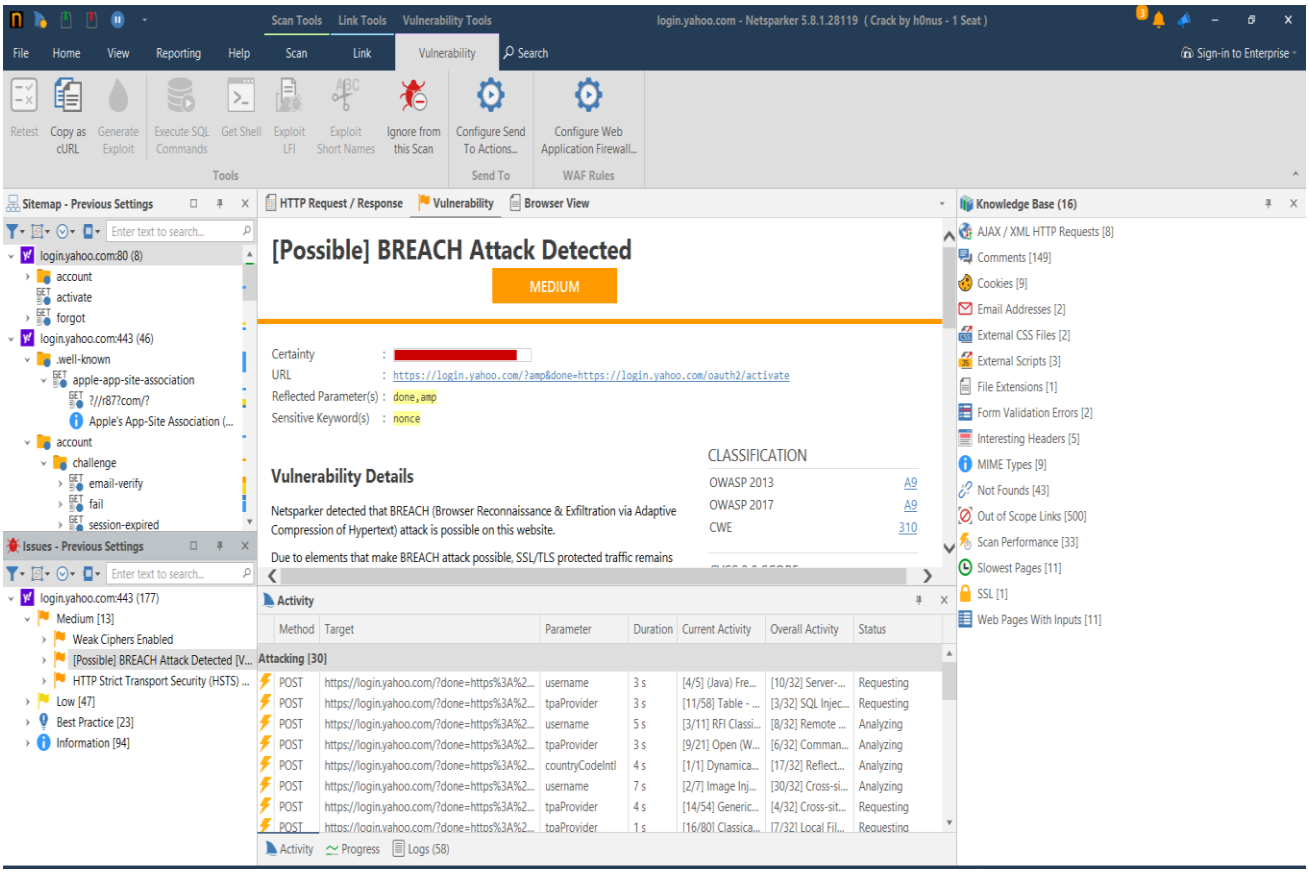
**Medium**: 15

**Low**: 31

**Best practice issues:** 17

**Information:** 30

- **login.yahoo.com**




Total issues found: 177

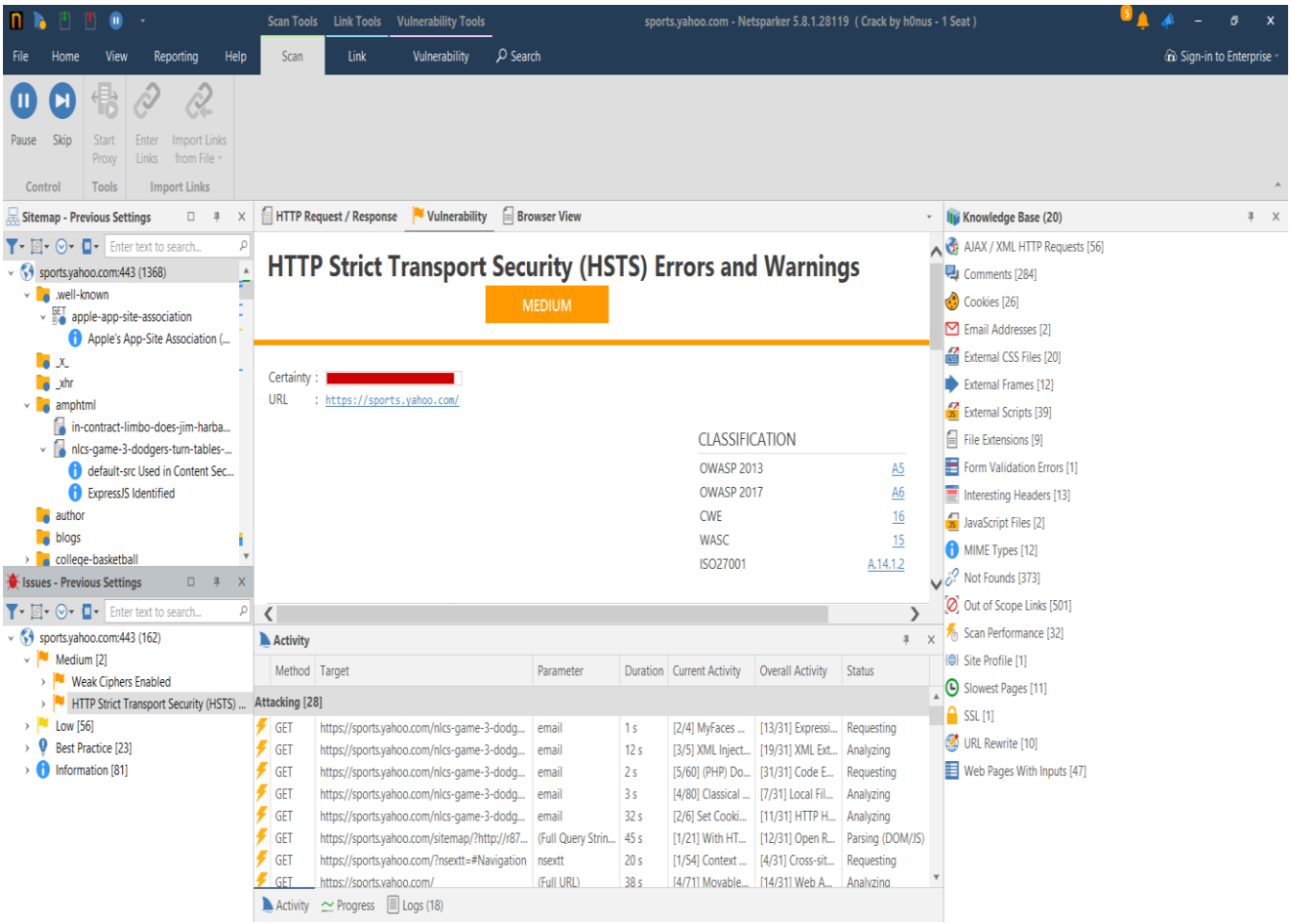
**Medium**: 13

**Low**: 47

**Best practice issues:** 23

**Information:** 94

- **sports.yahoo.com**




Total issues found: 162

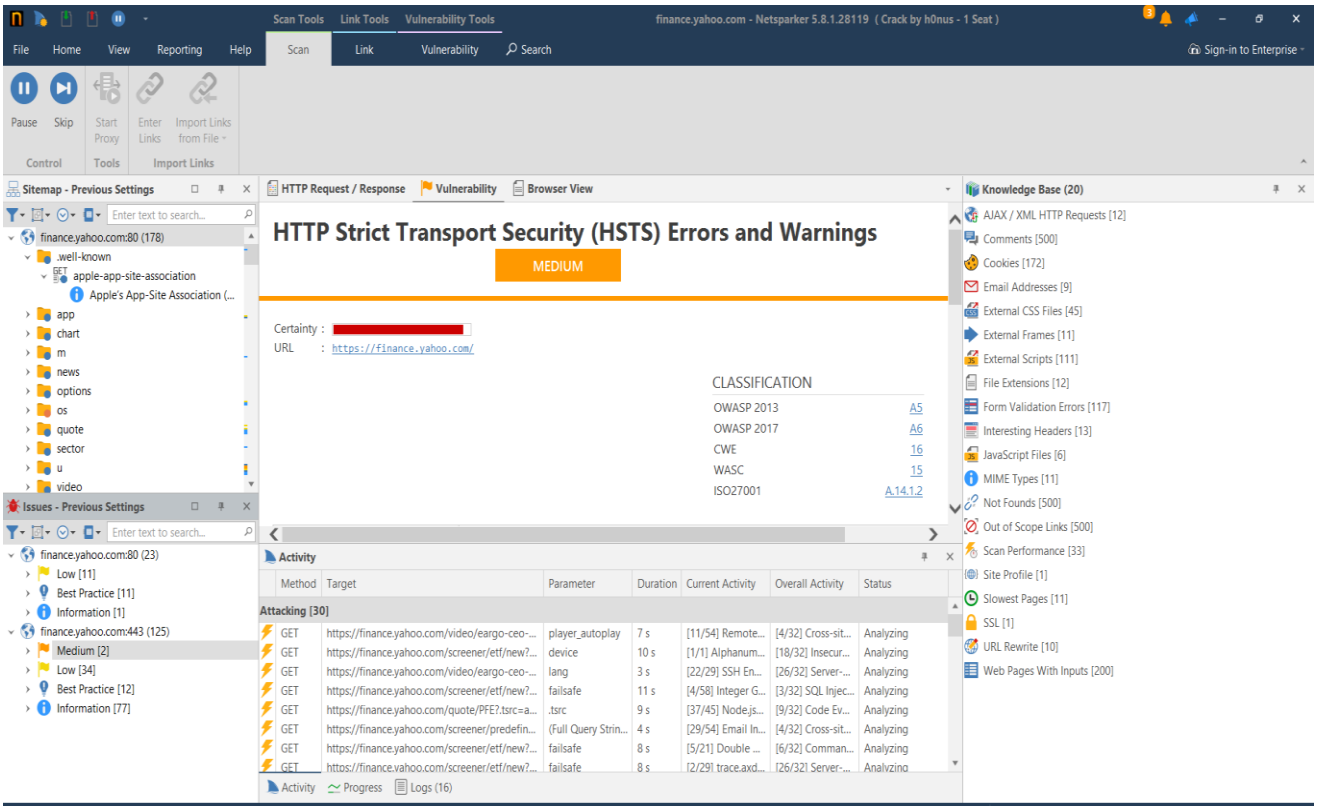
**Medium**: 2

**Low**: 56

**Best practice issues:** 23

**Information:** 81

- **finance.yahoo.com**




Total issues found: 148

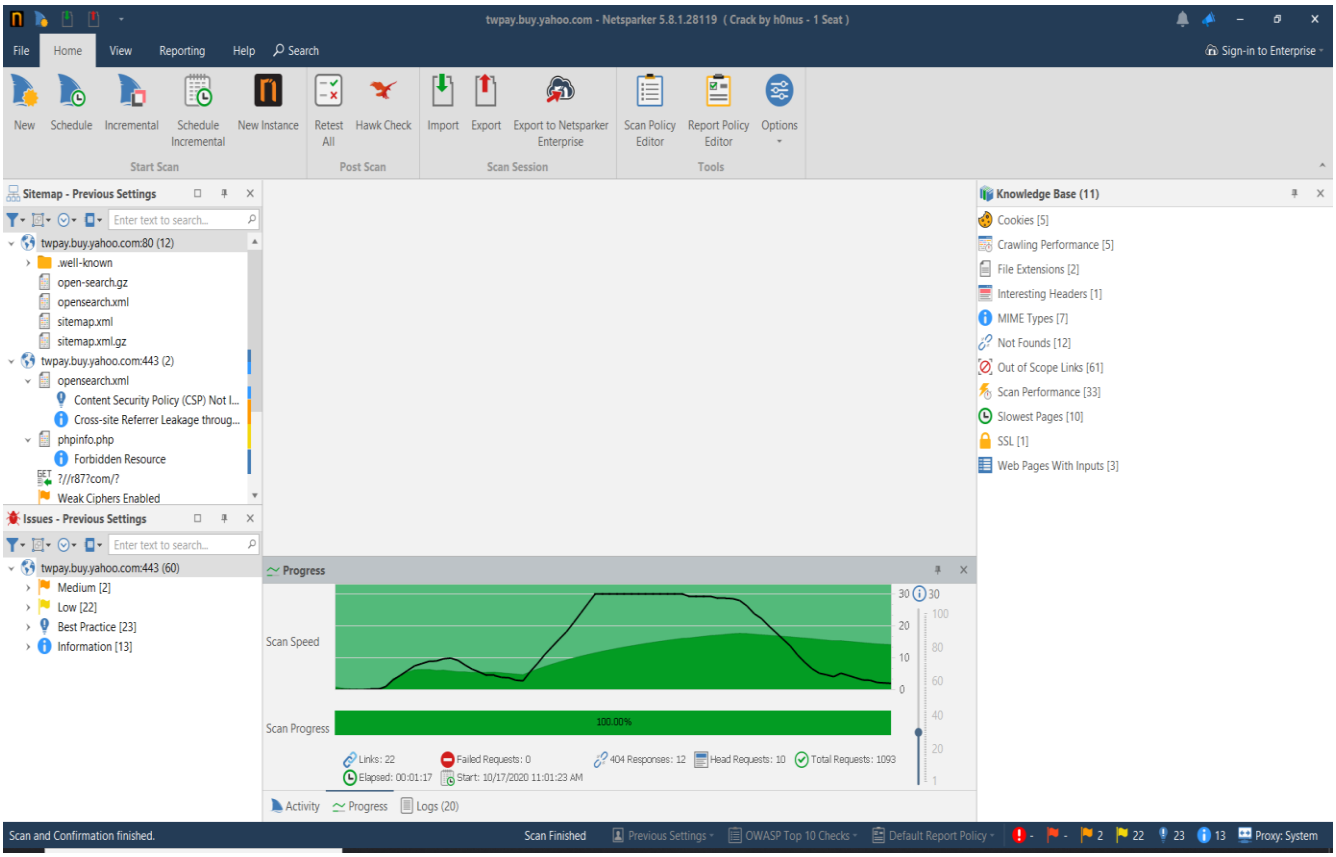
**Medium**: 2

**Low**: 45

**Best practice issues:** 23

**Information:** 30

- **twpay.buy.yahoo.com**




Total issues found: 60

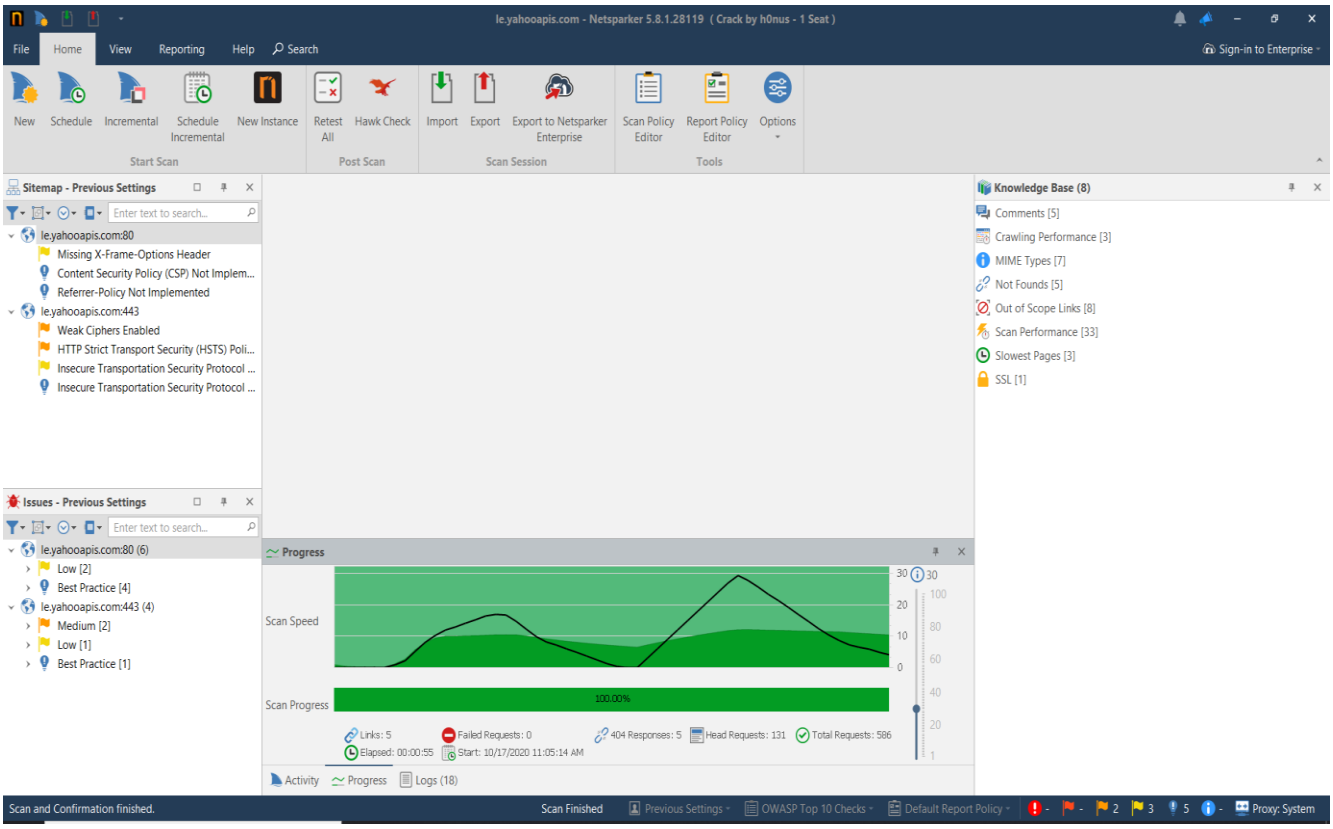
**Medium**: 2

**Low**: 22

**Best practice issues:** 23

**Information:** 13

- **le.yahooapis.com**




Total issues found: 10

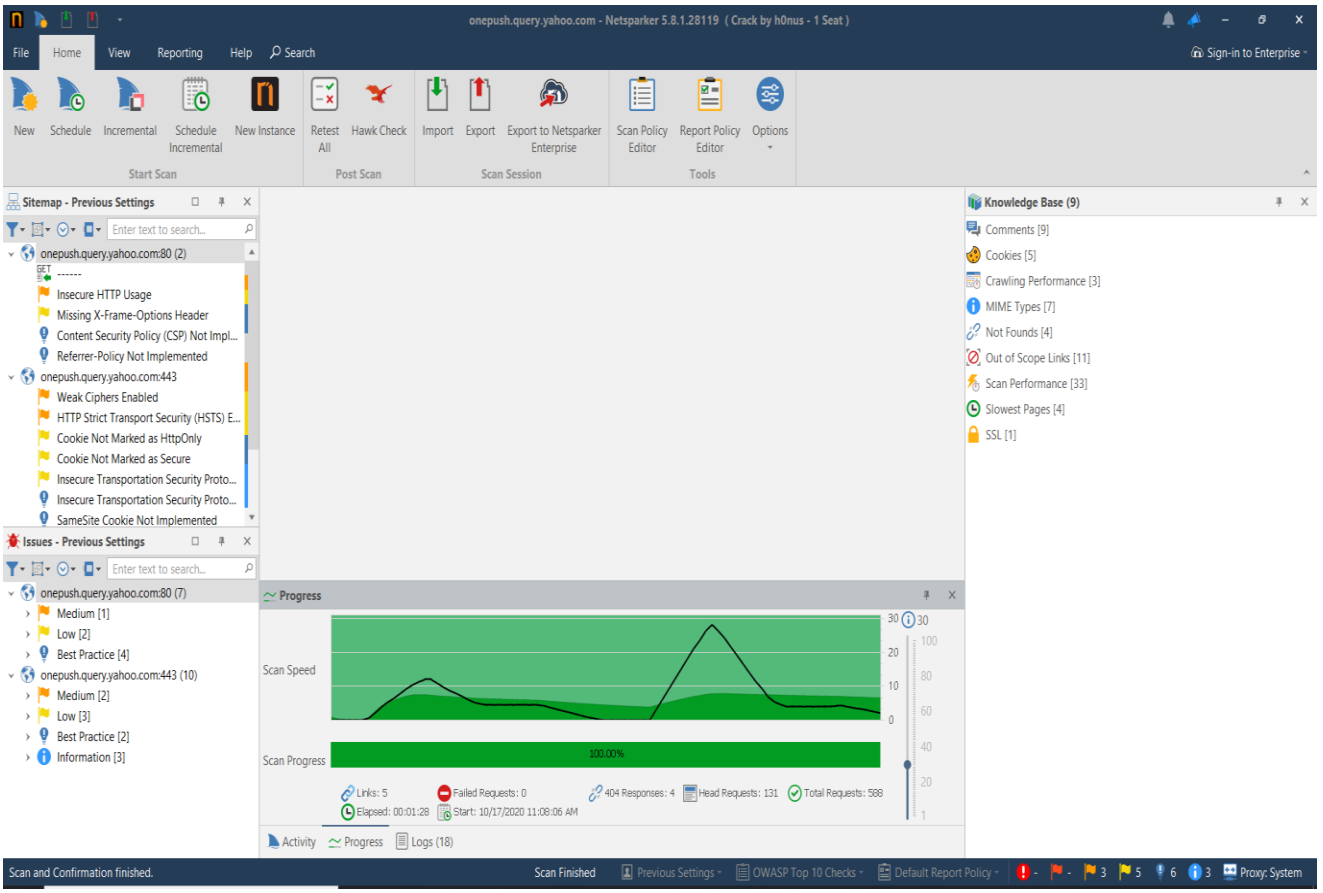
**Medium**: 2

**Low**: 3

**Best practice issues:** 5

**Information:** none

- **onepush.query.yahoo.com**




Total issues found: 17

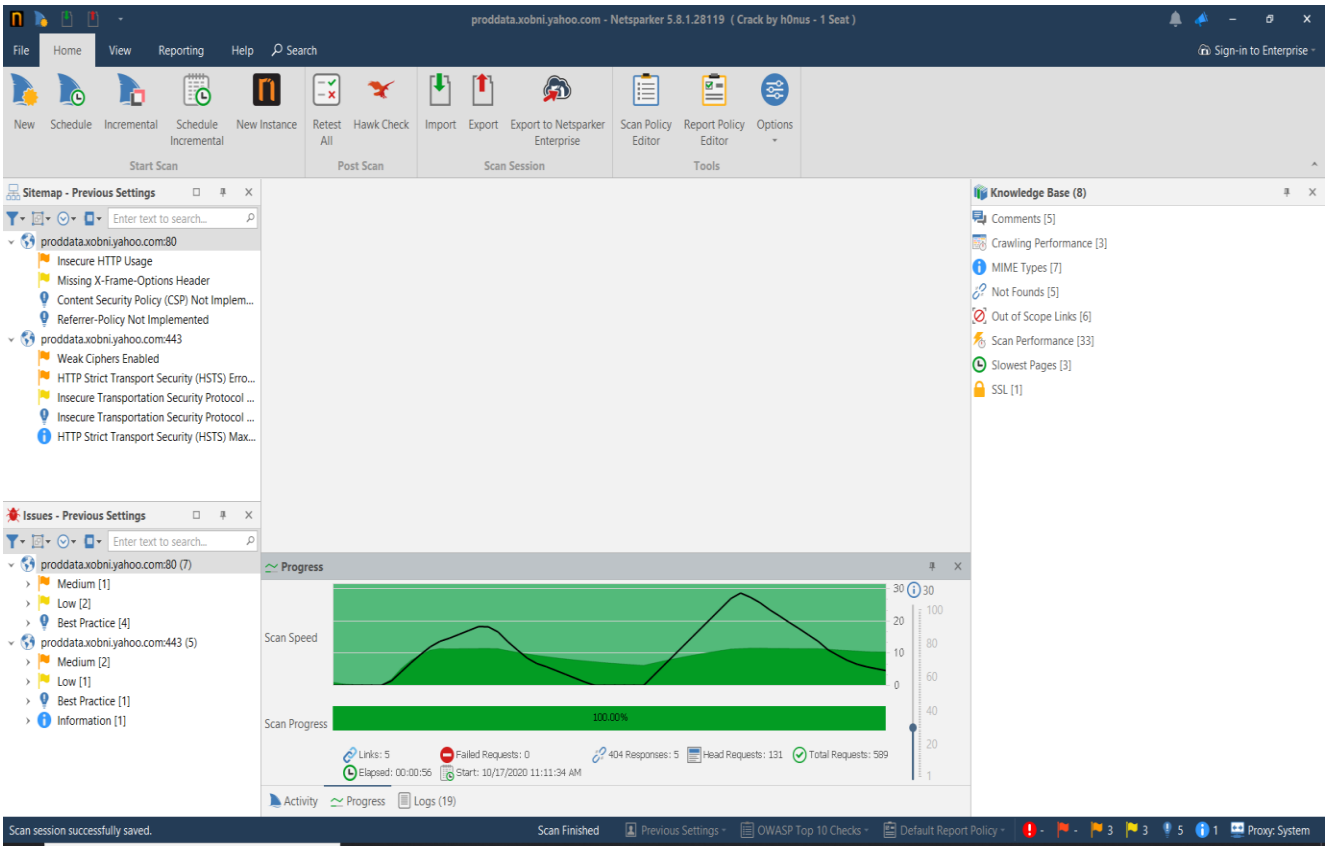
**Medium**: 2

**Low**: 5

**Best practice issues:** 6

**Information:** 3

- **proddata.xobni.yahoo.com**




Total issues found: 12

**Medium**: 3

**Low**: 3

**Best practice issues:** 5

**Information:** 1

# 3. Vulnerability analysis

This part contains a brief analysis on most of the vulnerabilities, logical flaws and information about usage of security libraries discovered in the vulnerability identification and threat modeling phase.

These vulnerabilities are categorized according to each sub domain in the scope and they are analyzed based on the severity of the vulnerability.

- Critical
- High
- Medium
- Low
- Information
- Best practices

This analysis contains details about each issue and the actions to be taken in order to overcome these vulnerabilities.

## data.mail.yahoo.com

Vulnerabilities with critical or high severity were not found.

### Medium

1. **Weak Ciphers Enabled** (OWASP A3 Sensitive data exposure)

   Weak ciphers are enabled during secure communication (SSL). Web applications should allow only strong ciphers on their web server to protect secure communication with the visitors.

   Impact:
   Attackers might decrypt SSL traffic between web server and visitors.

   Actions to take:
   Because yahoo.com uses Apache traffic servers, they should modify the SSLCipherSuite directive in the httpd.conf.
   SSLCipherSuite HIGH:MEDIUM:!MD5:!RC4
   Lighttpd:
   ssl.honor-cipher-order = "enable"
   ssl.cipher-list = "EECDH+AESGCM:EDH+AESGCM"

2. **HTTP Strict Transport Security (HSTS) Policy Not Enabled**

   The target website is being served from not only HTTPS but also HTTP and it lacks HSTS policy implementation

   Remedy:
   Configure the webserver to redirect HTTP requests to HTTPS.
   i.e. for Apache, they should have modification in the httpd.conf.

## Low

1. **Missing X-Frame-Options Header**

   missing X-Frame-Options header means that this website could be at risk of a clickjacking attack. Clickjacking is when an attacker uses multiple transparent or opaque layers to trick a user into clicking on a button or link on a framed page when they were intending to click on the top-level page. Thus, the attacker is "hijacking" clicks meant for their page and routing them to other another page, most likely owned by another application, domain, or both.

   Servers can declare this policy in the header of their HTTP responses to prevent clickjacking attacks, which ensures that their content is not embedded into other pages or frames.

2. **Insecure Transportation Security Protocol Supported (TLS 1.0)**

   TLS 1.0 has several flaws. An attacker can cause connection failures and they can trigger the use of TLS 1.0 to exploit vulnerabilities like BEAST (Browser Exploit Against SSL/TLS). Websites using TLS 1.0 are considered non-compliant by PCI since 30 June 2018.

   **Actions to take:**
   disable TLS 1.0 and replace it with TLS 1.2 or higher is recommended.

## Best practice issues

1. Content Security Policy (CSP) Not Implemented
2. Referrer-Policy Not Implemented

## mail.yahoo.com

Vulnerabilities with critical or high severity were not found.

**Medium**

1. **[Possible] BREACH Attack Detected**

   Due to elements that make BREACH (Browser Reconnaissance & Exfiltration via Adaptive Compression of Hypertext) attack possible, SSL/TLS protected traffic remains vulnerable and can be attacked to uncover information from the website. Even if the server use an SSL/TLS protected connection, an attacker can still view the victim's encrypted traffic and cause the victim to send HTTP requests to the vulnerable web server.

   Actions to take:

   If possible, disable HTTP level compression

   Separate sensitive information from user input

   Protect vulnerable pages with CSRF token.

2. **[Possible] Cross-site Scripting**

   Possible Cross-site Scripting, which allows an attacker to execute a dynamic script (JavaScript, VBScript) in the context of the application. This allows several different attack opportunities, mostly hijacking the current session of the user or changing the look of the page by changing the HTML on the fly to steal the user's credentials. This happens because the input entered by a user has been interpreted as HTML/JavaScript/VBScript by the browser.

   To avoid this, all input and output from the application should be filtered / encoded. Output should be filtered / encoded according to the output format and location.

3. **Weak Ciphers Enabled (**Described earlier)**
4. **HTTP Strict Transport Security (HSTS) Errors and Warnings (**Described earlier)**

## Low

1. **Autocomplete is Enabled**

   Autocomplete is Enabled in one or more of the form fields which might contain sensitive information like "username", "credit card" or "CVV". If user chooses to save, data entered in these fields will be cached by the browser. An attacker who can access the victim's browser could steal this information. This is especially important if the application is commonly used in shared computers, such as cyber cafes or airport terminals.

   Actions to take:

   Add the attribute `autocomplete="off"` to the form tag or to individual "input" fields. However, since early 2014, major browsers don't respect this instruction, due to their integrated password management mechanism, and offer to users to store password internally.
   Find all instances of inputs that store private data and disable autocomplete. Fields which contain data such as "Credit Card" or "CCV" type data should not be cached.

2. **Cookie Not Marked as HTTPOnly**
   HTTPOnly cookies cannot be read by client-side scripts, therefore marking a cookie as HTTPOnly can provide an additional layer of protection against cross-site scripting attacks. During a cross-site scripting attack, an attacker might easily access cookies and hijack the victim's session.

Action to take:

Mark the cookie as HTTPOnly.

3. **Cookie Not Marked as Secure**

   cookie not marked as secure and transmitted over HTTPS. This means the cookie could potentially be stolen by an attacker who can successfully intercept and decrypt the traffic or following a successful man-in-the-middle attack.

   Action to take:

   Mark all cookies used within the application as secure.

4. **Insecure Transportation Security Protocol Supported (TLS 1.0) (**Described earlier**)**

**Best practices and informational**

1. SameSite Cookie Not Implemented.
2. Subresource Integrity (SRI) Not Implemented.
3. Robots.txt Detected.
4. Content Security Policy (CSP) Contains Out of Scope report-uri Domain.
5. default-src Used in Content Security Policy (CSP).
6. Unsupported Hash Detected in Content Security Policy (CSP).
7. Wildcard Detected in Domain Portion of Content Security Policy (CSP) Directive.

## login.yahoo.com

Vulnerabilities with critical or high severity were not found.

**Medium**

1. Weak Ciphers Enabled.
2. [Possible] BREACH Attack Detected.
3. HTTP Strict Transport Security (HSTS) Errors and Warnings.

All of these vulnerabilities were described under previous domains vulnerability analysis. Same descriptions mentioned there are applied here.

**Low**

1. **[Possible] Cross-site Request Forgery**

   CSRF is a very common vulnerability. It's an attack which forces a user to execute unwanted actions on a web application in which the user is currently authenticated.

   Actions to take:

   Send additional information in each HTTP request that can be used to determine whether the request came from an authorized source. This "validation token" should be hard to guess for attacker who does not already have access to the user's account.

2. **[Possible] Cross-site Request Forgery in Login Form**

   In this case CSRF affects the login form in which the impact of this vulnerability is decreased significantly. Unlike normal CSRF vulnerabilities this will only allow an attacker to exploit some complex XSS vulnerabilities otherwise it cannot be exploited.

Same remedies that are described under number 1 applies here.

3. **Autocomplete is Enabled. (****Described earlier**)**
4. **Cookie Not Marked as HttpOnly. (****Described earlier**)**
5. **Cookie Not Marked as Secure. (****Described earlier**)**
6. **Insecure Transportation Security Protocol Supported (TLS 1.0). (****Described earlier**)**

**Best practices and informational**

1. Insecure Transportation Security Protocol Supported (TLS 1.1)
2. SameSite Cookie Not Implemented
3. Subresource Integrity (SRI) Not Implemented
4. Autocomplete Enabled (Password Field)
5. Cross-site Referrer Leakage through Referrer-Policy
6. Robots.txt Detected
7. Content Security Policy (CSP) Contains Out of Scope report-uri Domain
8. Content Security Policy (CSP) Nonce Without Matching Script Block
9. Wildcard Detected in Domain Portion of Content Security Policy (CSP) Directive
10. Weak Nonce Detected in Content Security Policy (CSP) Declaration

## sports.yahoo.com

Vulnerabilities with critical or high severity were not found.

### Medium

1. **Weak Ciphers Enabled**
2. **HTTP Strict Transport Security (HSTS) Errors and Warnings**

These vulnerabilities were described under previous domains vulnerability analysis. Same descriptions mentioned there are applied here.

### Low

1. **Missing X-Frame-Options Header**

   missing X-Frame-Options header which means that this website could be at risk of a clickjacking attack. Clickjacking is when an attacker uses multiple transparent or opaque layers to trick a user into clicking on a button or link on a framed page when they were intending to click on the top-level page. Thus, the attacker is "hijacking" clicks meant for their page and routing them to other another page, most likely owned by another application, domain, or both.

   Actions to take:
   Sending the proper X-Frame-Options in HTTP response headers that instruct the browser to not allow framing from other domains.

   - X-Frame-Options: DENY It completely denies to be loaded in frame/iframe.
   - X-Frame-Options: SAMEORIGIN It allows only if the site which wants to load has a same origin.

- X-Frame-Options: ALLOW-FROM *URL* It grants a specific URL to load itself in a iframe.

2. **Cookie Not Marked as HttpOnly. (****Described earlier**)**
3. **Cookie Not Marked as Secure. (****Described earlier**)**
4. **Insecure Transportation Security Protocol Supported (TLS 1.0). (****Described earlier**)**

**Best practices and informational**

1. Subresource Integrity (SRI) Not Implemented
2. SameSite Cookie Not Implemented
3. Insecure Transportation Security Protocol Supported (TLS 1.1)
4. Autocomplete Enabled (Password Field)
5. Cross-site Referrer Leakage through Referrer-Policy
6. Robots.txt Detected

## finance.yahoo.com

Vulnerabilities with critical or high severity were not found.

### Medium

1. **Weak Ciphers Enabled**
2. **HTTP Strict Transport Security (HSTS) Errors and Warnings**

These vulnerabilities were described under previous domains vulnerability analysis. Same descriptions mentioned there are applied here.

### Low

1. **Insecure Frame (External)**

   identified an external insecure or misconfigured iframe. IFrame sandboxing enables a set of additional restrictions for the content within a frame in order to restrict its potentially malicious code from causing harm to the web page that embeds it.

   Actions to take:

   Apply sandboxing in inline frame
   `<iframe sandbox src="framed-page-url"></iframe>`

   For untrusted content, avoid the usage of `seamless` attribute and `allow-top-navigation`, `allow-popups` and `allow-scripts` in sandbox attribute.

2. **Cookie Not Marked as HttpOnly (**** Described earlier**)**
3. **Insecure Transportation Security Protocol Supported (TLS 1.0) (**** Described earlier**)**
4. **Missing X-Frame-Options Header (**** Described earlier**)**

**Best practices and informational**

1. Insecure Transportation Security Protocol Supported (TLS 1.1)
2. Subresource Integrity (SRI) Not Implemented
3. Cross-site Referrer Leakage through Referrer-Policy
4. Robots.txt Detected
5. Content Security Policy (CSP) Contains Out of Scope report-uri Domain
6. Missing object-src in CSP Declaration
7. Out-of-date Version (Modernizr)
8. Wildcard Detected in Domain Portion of Content Security Policy (CSP) Directive

## twpay.buy.yahoo.com

Vulnerabilities with critical or high severity were not found.

**Medium**

1. **Weak Ciphers Enabled**
2. **HTTP Strict Transport Security (HSTS) Errors and Warnings**

These vulnerabilities were described under previous domains vulnerability analysis. Same descriptions mentioned there are applied here.

**Low**

1. **Cookie Not Marked as HttpOnly. (*********Described earlier**)**
2. **Cookie Not Marked as Secure. (*********Described earlier**)**

**Best practices and informational**

1. Insecure Transportation Security Protocol Supported (TLS 1.1)
2. Content Security Policy (CSP) Not Implemented
3. SameSite Cookie Not Implemented
4. Cross-site Referrer Leakage through Referrer-Policy
5. Forbidden Resource

## le.yahooapis.com

Vulnerabilities with critical or high severity were not found.

### Medium

1. **Weak Ciphers Enabled**
2. **HTTP Strict Transport Security (HSTS) Errors and Warnings**

These vulnerabilities were described under previous domains vulnerability analysis. Same descriptions mentioned there are applied here.

### Low

1. **Missing X-Frame-Options Header**
2. **Insecure Transportation Security Protocol Supported (TLS 1.0)**

These vulnerabilities were described under previous domains vulnerability analysis. Same descriptions mentioned there are applied here.

### Best practices and informational

1. Insecure Transportation Security Protocol Supported (TLS 1.1)

## onepush.query.yahoo.com

Vulnerabilities with critical or high severity were not found.

**Medium**

1. **Insecure HTTP Usage**

   Target website allows web browsers to access to the website over HTTP and doesn't redirect them to HTTPS. Users will not be able to take advantage of HSTS which almost renders the HSTS implementation useless. Not having HSTS will make Man in the middle attacks easier for attackers.

   Actions to take:
   Configure your webserver to redirect HTTP requests to HTTPS.

2. **Weak Ciphers Enabled**
3. **HTTP Strict Transport Security (HSTS) Errors and Warnings**

These vulnerabilities were described under previous domains vulnerability analysis. Same descriptions mentioned there are applied here.

**Low**

1. **Missing X-Frame-Options Header**
2. **Insecure Transportation Security Protocol Supported (TLS 1.0)**
3. **Cookie Not Marked as HttpOnly**
4. **Cookie Not Marked as Secure**

**Best practices and informational**

1. Insecure Transportation Security Protocol Supported (TLS 1.1)
2. SameSite Cookie Not Implemented
3. Cross-site Referrer Leakage through Referrer-Policy
4. Forbidden Resource
5. HTTP Strict Transport Security (HSTS) Max-Age Value Too Low

## proddata.xobni.yahoo.com

Vulnerabilities with critical or high severity were not found.

**Medium**

1. **Insecure HTTP Usage**
2. **Weak Ciphers Enabled**
3. **HTTP Strict Transport Security (HSTS) Errors and Warnings**

These vulnerabilities were described under previous domains vulnerability analysis. Same descriptions mentioned there are applied here.

**Low**

1. **Missing X-Frame-Options Header**
2. **Insecure Transportation Security Protocol Supported (TLS 1.0)**

These vulnerabilities were described under previous domains vulnerability analysis. Same descriptions mentioned there are applied here.

**Best practices and informational**

1. Content Security Policy (CSP) Not Implemented
2. Referrer-Policy Not Implemented
3. Insecure Transportation Security Protocol Supported (TLS 1.1)
4. HTTP Strict Transport Security (HSTS) Max-Age Value Too Low

# Conclusion

When analyzing this entire web audit report, few vulnerabilities and logical flaws/best practices issues has noticed repeatedly in almost all subdomains selected. Those vulnerabilities are listed below.

1. Weak Ciphers Enabled **(Medium)**
2. HTTP Strict Transport Security (HSTS) Errors and Warnings **(Medium)**

3. Weak Ciphers Enabled (**Low**)
4. HTTP Strict Transport Security (HSTS) Errors and Warnings (**Low**)

5. Referrer-Policy Not Implemented (**Best practices and informational**)
6. Insecure Transportation Security Protocol Supported (TLS 1.1) (**Best practices and informational**)

Because these vulnerabilities were commonly detected in all of the scanned nine sub domains, it is recommended that the necessary actions mentioned in the vulnerability analysis should be taken in order to prevent any malicious activities.

In addition, these vulnerabilities are common vulnerabilities that are detected in most of the web applications. Therefore when considering overall web audit, it can be assured that yahoo.com is a web application that maintains a proper security infrastructure in order to protect it's sensitive data.

# References


[1] J. Petters, "www.varonis.com," 29 03 2020. [Online]. Available: https://www.varonis.com/blog/security-audit/. [Accessed 16 10 2020].

[2] v. media, "hackerone.com," [Online]. Available: https://hackerone.com/verizonmedia?type=team. [Accessed 16 10 2020].

[3] Aboul3la, "GitHub," [Online]. Available: https://github.com/aboul3la/Sublist3r. [Accessed 16 10 2020].

[4] Aboul3la, "github," [Online]. Available: https://github.com/aboul3la/Sublist3r. [Accessed 15 10 2020].