Hao Shu

Weekly Report 2020-10-12

This week I had the initial meeting with Professor Huang. During the meeting, we discussed research ground rules, expectations, and the two projects that are going on. The first project is to add a vulnerability scanning feature into IoT inspector, while the second one is to fingerprint IoT TLS traffic. For this week, I have been focusing on project 1.

Problem tried to address this week:

I am trying to add the vulnerability scanner feature into the IoT inspector project. Currently the IoT inspector can check for IoT network traffic, and analyze them, but can't do any CVE lookup or associate any service with known vulnerability.

Solutions:

There are two solutions: building from scratch a vulnerability scanner with CVE lookup ability, or integrating an existing open source scanner. The logic/flow of solution 1 is as the following: (the vulnerability scanner needs to do the following)

1. Port scanning. The goal of port scanning is to check the detectability of an IoT device, and identify vulnerable open ports.(Can be achieved with python nmap library) The output of the Nmap scan will be saved in a file. Then, the scanner will line by line find discovered open ports, and compare those with top vulnerabile ports files(pre-configured). Ports that are highly vulnerable, should be reported.
2. OS fingerprinting. The goal is to identify IoT device OS, manufacturer, type. This can be done by capturing DHCP discovery sent by IoTs, and by using nmap OS fingerprinting.
3. Process identification, enumeration. The goal is to find out what processes are running on the IoT device. This can be achieved by nmap advanced scans.

4. CVE lookup. This goal is to look up CVEs. This can be done by querying a CVE database online or locally. The identified OS, and processes from step 2,3 will be used as input of the query.
5. Vulnerability reporting. The goal of this is to clean queried information and prepare them for a user-friendly output format.

   Reference(https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=8565917)

However, one drawback of this approach is that it is reinventing the wheel. It is building something from scratch and is error-prone. Because of this, this feature can't immediately be included in the IoT inspector, and many tests have to be done. Another solution is to find an existing free vulnerability scanner, and integrate it into the IoT inspector.

After researching online and testing on a virtual machine, the following are some of the vulnerability scanners in python, each with pros and cons.

Name: IoT Scanner

License:GNU General Public License

Url:https://github.com/JurreitJ/IoTScanner

Pros:	**Cons:**
Easy to install, and use	**No CVE lookup**
Clear structure and simple source code	**No vulnerability reporting**
Can port scan, analyse http, ssh, Zigbee network	

Hao Shu

Name: SeeSec---IoT-Vulnerability-Scanner

License:MIT

Url:https://github.com/ruthogunnnaike/SeeSec---IoT-Vulnerablity-Scanner

Pros:	Cons:
Has ability to prevent vulnerable device from using the network	Unclear instruction/install
Attempts to fix the vulnerability	Relies on Nmap, Nessus for vulnerability reporting, scanning

Name: Rapidscan

License:GNU General Public License

Url:https://github.com/skavngr/rapidscan

Pros:	Cons:
Fast installation	Not made specially for IoTs
Multiple checks to zero out false positive	Seems to focus on web vulnerability(Can be used if IoT has web admin panel or web interface)
Classification of vulnerabilities	
Remediation report	

Hao Shu

Name: Routersploit

License:BSD with condition(giving credit)
Url:https://github.com/threat9/routersploit

Pros:	Cons:
A lot of usage examples	No CVE lookup
Easy install	Seems to be only able to scan http, ssh, telnet

Name: Xunfeng

License:GNU v3
Url:https://github.com/ysrc/xunfeng

Pros:	Cons:
GUI	Many dependencies, install/config steps
Instruction in Chinese	Preconfigured sets of vulnerabilities, seem to be able to dynamically pull vulnerabilities from Kunpeng data source

Hao Shu

Name: Vfeed

License:Free for non-commercial use

Url:https://github.com/toolswatch/vFeed

Pros:	Cons:
Detailed, structured reference for a CVE	Not a scanner
Include multiple description of a CVE from multiple source	Can't find documentation

This is a CVE database wrapper, it shows the detailed description of a CVE from multiple sources.

Hao Shu

The following scanning frameworks are designed, focused on IoTs:

Name: IoTSecFuzz

License:MIT

Url:https://gitlab.com/invuls/iot-projects/iotsecfuzz/-/tree/master/

Pros:	Cons:
Easy install, python3 compatible	Standard module installation is not finished
Can customize modules	Small database of modules

Name: HomePwn

License:GNU

Url:https://github.com/ElevenPaths/HomePWN

Pros:	Cons:
Clear, detailed documentation	Hard to install, lots of dependencies
Support module creation for customized vulnerability testing	

Hao Shu

Recommendation:

Name: Cotopaxi

License:GNU General Public License
Url:https://github.com/Samsung/cotopaxi

Pros:	Cons:
Support security testing of MQP, CoAP, DTLS, HTCPCP, mDNS, MQTT, MQTT-SN, QUIC, RTSP, SSDP protocols.	Limited Device Identification, see its instruction page
Advanced usage	Static vulnerability database(ex. Only 1 vulnerability Yaml file)
CoAp,DTLS server fingerprinting	

Name: Expliot

License:GNU Affero General Public License
Url:https://gitlab.com/expliot_framework/expliot

Pros:	Cons:
Both interactive mode and CLI mode	Hard to install, dependencies cause problems
Plugins can be extended	Predefined limited set of plugins
Made specially for IoTs	

Hao Shu

Name: Nettacker

License:Apache license

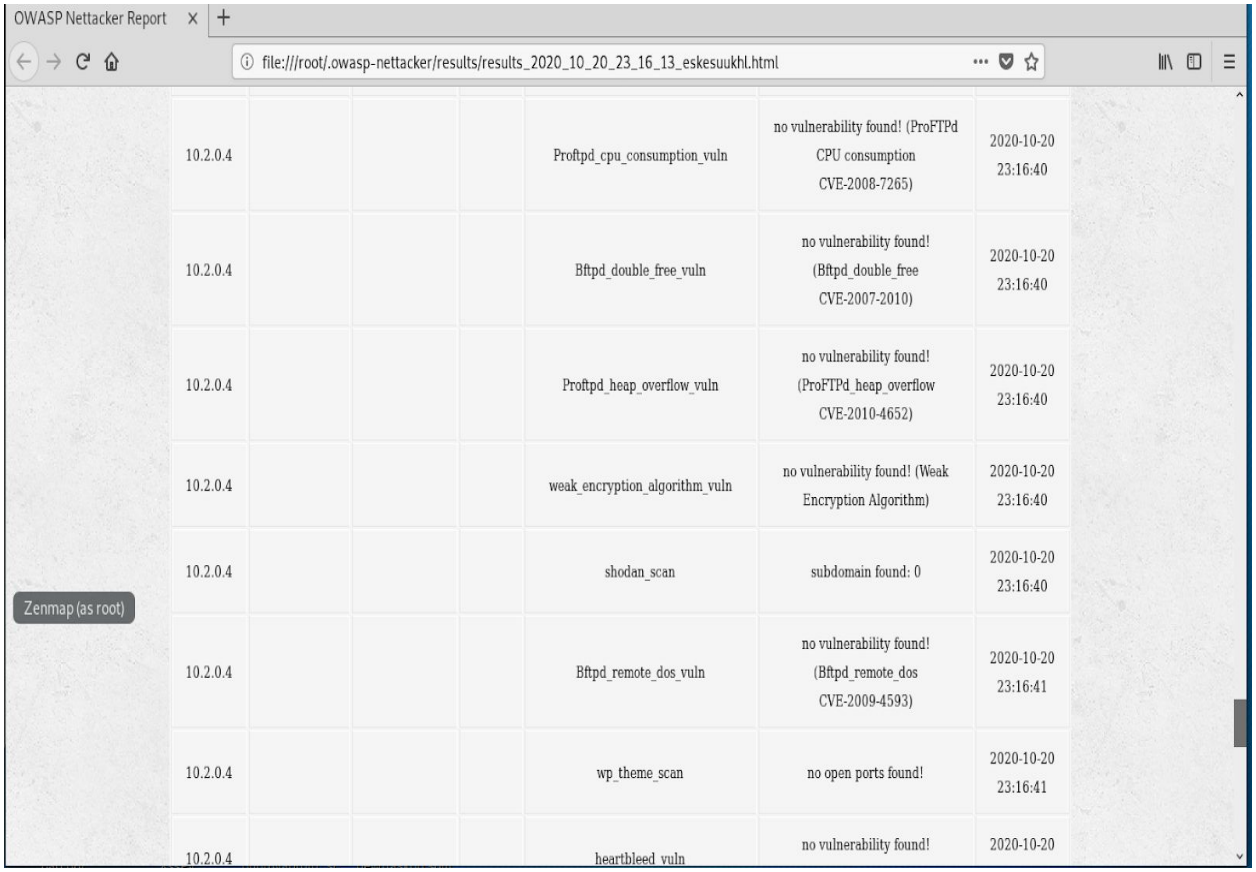
Url:https://github.com/OWASP/Nettacker

Pros:	Cons:
Owasp framework, easy install	No CVE lookup
Lots of vulnerabilities to test, and language support	No OS fingerprinting
Shodan included	Pre defined wordlist, vulnerability don't seem to work perfectly against an actual IoT
GUI, and visualization supported via graphs, allows user to select which graph	

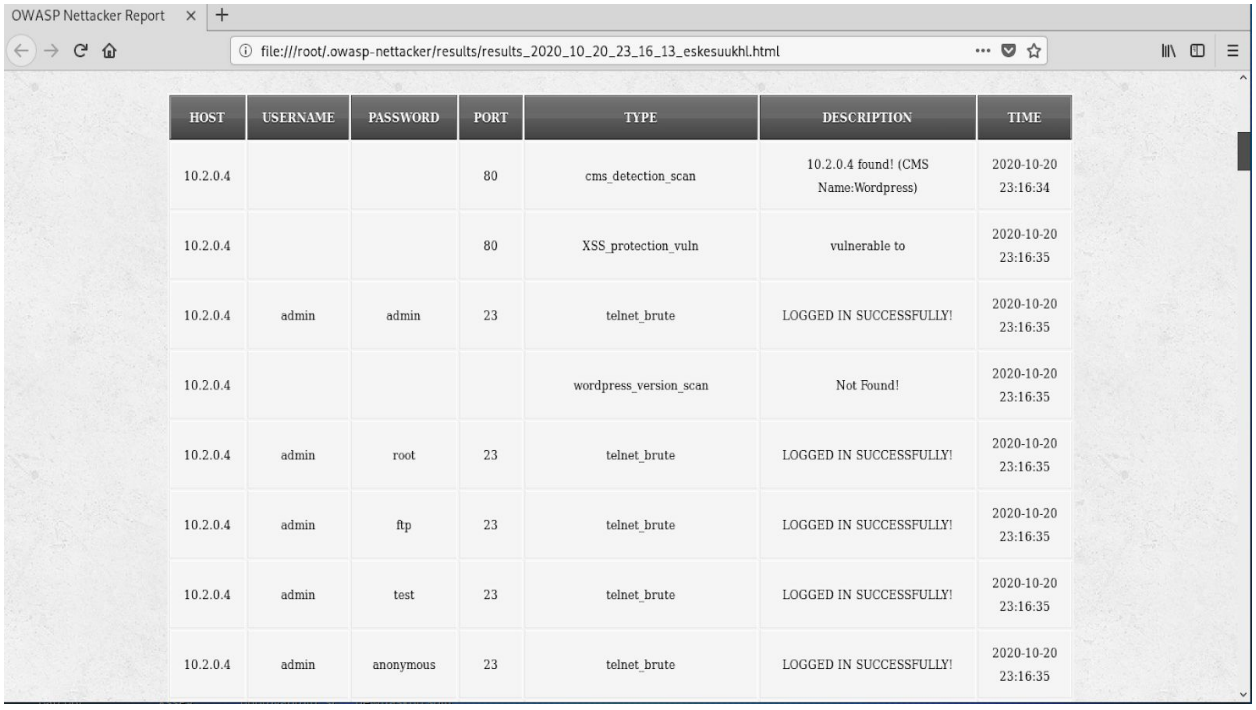
Instead of figuring out what OS, processes running on the IoT then look up associated CVEs, this scanner checks a predefined file of vulnerabilities.
After testing Nettacker on a virtual machine, this is a recommended vulnerability tester. The vms consist of a Kali linux running as host, a metasploitable 2 as target. The test run finished fast.

The following figures show the vulnerability scanning result against metasploitable 2 vm.

Hao Shu



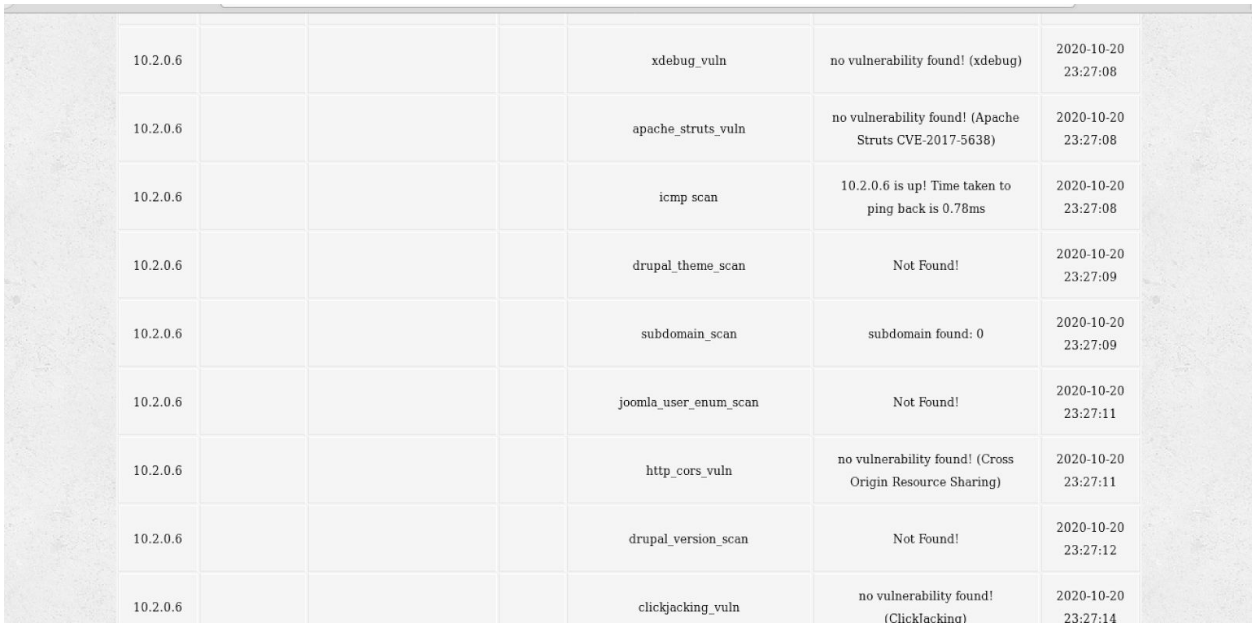
10.2.0.4				Proftpd_cpu_consumption_vuln	no vulnerability found! (ProFTPd CPU consumption CVE-2008-7265)	2020-10-20 23:16:40
10.2.0.4				Bftpd_double_free_vuln	no vulnerability found! (Bftpd_double_free CVE-2007-2010)	2020-10-20 23:16:40
10.2.0.4				Proftpd_heap_overflow_vuln	no vulnerability found! (ProFTPd_heap_overflow CVE-2010-4652)	2020-10-20 23:16:40
10.2.0.4				weak_encryption_algorithm_vuln	no vulnerability found! (Weak Encryption Algorithm)	2020-10-20 23:16:40
10.2.0.4				shodan_scan	subdomain found: 0	2020-10-20 23:16:40
10.2.0.4				Bftpd_remote_dos_vuln	no vulnerability found! (Bftpd_remote_dos CVE-2009-4593)	2020-10-20 23:16:41
10.2.0.4				wp_theme_scan	no open ports found!	2020-10-20 23:16:41
10.2.0.4				heartbleed_vuln	no vulnerability found!	2020-10-20



HOST	USERNAME	PASSWORD	PORT	TYPE	DESCRIPTION	TIME
10.2.0.4			80	cms_detection_scan	10.2.0.4 found! (CMS Name:Wordpress)	2020-10-20 23:16:34
10.2.0.4			80	XSS_protection_vuln	vulnerable to	2020-10-20 23:16:35
10.2.0.4	admin	admin	23	telnet_brute	LOGGED IN SUCCESSFULLY!	2020-10-20 23:16:35
10.2.0.4				wordpress_version_scan	Not Found!	2020-10-20 23:16:35
10.2.0.4	admin	root	23	telnet_brute	LOGGED IN SUCCESSFULLY!	2020-10-20 23:16:35
10.2.0.4	admin	ftp	23	telnet_brute	LOGGED IN SUCCESSFULLY!	2020-10-20 23:16:35
10.2.0.4	admin	test	23	telnet_brute	LOGGED IN SUCCESSFULLY!	2020-10-20 23:16:35
10.2.0.4	admin	anonymous	23	telnet_brute	LOGGED IN SUCCESSFULLY!	2020-10-20 23:16:35

Hao Shu

To see how it works, I also tested it against an IoT vm, called IoT goat, which is created by OWASP as an educational material for IoT security.



10.2.0.6				xdebug_vuln	no vulnerability found! (xdebug)	2020-10-20 23:27:08
10.2.0.6				apache_struts_vuln	no vulnerability found! (Apache Struts CVE-2017-5638)	2020-10-20 23:27:08
10.2.0.6				icmp scan	10.2.0.6 is up! Time taken to ping back is 0.78ms	2020-10-20 23:27:08
10.2.0.6				drupal_theme_scan	Not Found!	2020-10-20 23:27:09
10.2.0.6				subdomain_scan	subdomain found: 0	2020-10-20 23:27:09
10.2.0.6				joomla_user_enum_scan	Not Found!	2020-10-20 23:27:11
10.2.0.6				http_cors_vuln	no vulnerability found! (Cross Origin Resource Sharing)	2020-10-20 23:27:11
10.2.0.6				drupal_version_scan	Not Found!	2020-10-20 23:27:12
10.2.0.6				clickjacking_vuln	no vulnerability found! (Clicklacking)	2020-10-20 23:27:14

Although it does a pretty good job at information gathering, like what is running on which port, its drawback is obvious. When a vulnerability is novel, and is not in its pre defined vulnerability list, then it's not very effective.