Benchmarking Vulnerability Scanners: An Experiment on SCADA Devices and Scientific Instruments

By:

Malaka EL

A Master's Paper Submitted to the Faculty of the

DEPARTMENT OF MANAGEMENT INFORMATION SYSTEMS

ELLER COLLEGE OF MANAGEMENT

In Partial Fulfillment of the Requirements

For the Degree of

MASTER OF SCIENCE

In the Graduate College

THE UNIVERSITY OF ARIZONA

2017

STATEMENT BY AUTHOR

This thesis has been submitted in partial fulfillment of requirements for an advanced degree at the University of Arizona.

Brief quotations from this thesis are allowable without special permission, provided that an accurate acknowledgement of the source is made. Requests for permission of extended quotation from or reproduction of this manuscript in whole or in part may be granted by the head of the major department or the Dean of the Graduate College when in his or her judgment the proposed use of the material is in the interests of scholarship. In all other instances, however, permission must be obtained from the author.

SIGNED: Malaka EL

APPROVAL BY MASTERS PAPER ADVISOR

This thesis has been approved on the date shown below:

	MM/DD/2017
Dr. Hsinchen Chen	Date
Advisor Title of Management Information Systems	

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

Cybersecurity is a critical concern in society today. One common avenue of attack for malicious hackers is exploiting vulnerable websites. It is estimated that there are over one million websites that are attacked daily. Two emerging targets of such attacks are Supervisory Control and Data Acquisition (SCADA) devices and scientific instruments. Vulnerability assessment tools can provide owners of these devices with the knowledge on how to protect their infrastructure. However, owners face difficulties in identifying which tools are ideal for their assessments. This research aims to benchmark two state-of-the-art vulnerability assessment tools, Nessus and Burp Suite (Burp), in the context of SCADA devices and scientific instruments. We specifically focus on identifying the accuracy, scalability, and vulnerability results of the scans. Results of our study indicate that both tools together can provide a comprehensive assessment of the vulnerabilities in SCADA devices and scientific instruments.

The following are some keywords used throughout this paper: benchmark, SCADA, scientific instruments, vulnerabilty assessment tools, Nessus, Burp

1 INTRODUCTION

Cyberattacks have steadily increased in recent years (Figure 1). One of the most common avenues of attack is exploiting vulnerable websites [1]. It is estimated that over one million websites are attacked daily, with 75% of these websites containing unpatched vulnerabilities. Two emerging targets of such attacks are Supervisory Control and Data Acquisition (SCADA) systems and scientific instruments [2].

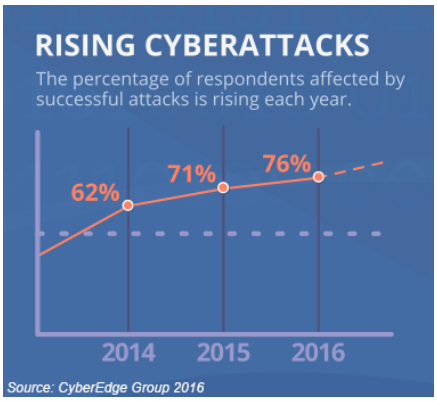

Figure 1: Rising Cyberattacks

SCADA systems control critical infrastructure (e.g., power plants, electrical grids, etc.) and have reported 46 website exploitations in the energy industry, 31 in water and dams, and 23 in transportation [3]. Scientific instruments aim to assist scientists in pursuing their scientific endeavors (e.g., genome sequencing, particle physics, astronomy). Today, scientific instruments not only collect data in labs but also from telescopes, particle accelerators, smartphones, drones, balloons, and sensors [4]. Unfortunately, scientific instruments have not been immune to cyber-attacks. NASA reported 631 video feeds have been stolen from aircraft and weather radars, and a US weather network suffered an electronic attack [5][6]. The broad range of SCADA devices and scientific instruments results in a vast potential attack surface for malicious cyber threats [7].

Vulnerability assessments can help safeguard SCADA devices and scientific instruments. These assessments determine the actual security posture of a network's environment [8]. They also identify weaknesses and detail mitigation procedures to eliminate the vulnerability or reduce the risk [9]. Previous vulnerability assessments utilized the National Vulnerability Database (NVD), Nessus, and custom scripts to identify SCADA vulnerabilities [10][11]. However, these tools were not evaluated or benchmarked for performance and accuracy. Benchmarking these assessment tools can provide valuable knowledge of their performance, accuracy, vulnerability coverage, and scalability. This is especially useful when evaluating device vulnerabilities on Shodan (search

engine for the Internet of Things, ~1.5 billion records). As such, this research aims to benchmark state-of-the-art vulnerability assessment tools in the context of SCADA devices and scientific instruments. We specifically focus on identifying the performance, accuracy, and scalability of vulnerability assessment tools as well as discover the vulnerabilities of the scanned devices.

The remainder of the paper is organized as follows. First, there is a review of multi-purpose and web application vulnerability assessment tools as well as benchmarking literature to understand key methods of evaluating selected tools. Subsequently, the research testbed and design are detailed. Then, the key findings and results are summarized. Finally, this research is concluded by highlighting several promising directions for future research.

2 LITERATURE REVIEW

To form the basis of this research, two major categories of vulnerability assessment tools (multi-purpose and web-application) are reviewed to identify tools ideal for benchmarking. The focus is on understanding each tool's scalability and vulnerability scanning performance. Lastly, there is a review of benchmarking literature in order to identify which assessment tools are appropriate to perform vulnerability assessments.

2.1 Vulnerability Assessment Tools

Multi-purpose tools provide users the capability to assess vulnerabilities of a multitude of devices (e.g., routers, printers, web applications, SCADA, etc.). Eleven such tools were reviewed: Nessus, OpenVAS, Nexpose, GFI LanGuard, QualsGuard, MBSA, Retina, Nipper, SAINT, Core Impact, and Secunia PSI (Table 1). Of these, Nessus is the most well-maintained and well suited for large-scale vulnerability assessments. Nessus offers over 80,000 plug-ins such as web

application testing, SCADA assessments, and discovering system issues. Nessus is a highly scalable tool, often used in networks with tens of thousands of devices. Nessus categorizes each vulnerability as Critical, High, Medium, Low, or Information based on the Common Vulnerability Scoring System (CVSS). Although tools such as OpenVAS, Nexpose, and QualysGuard are valuable, their scanning coverage and scalability are limited compared to Nessus. Other tools, such as Core Impact and Secunia PSI, did not provide scalability or efficiency details.

Tool Name	Company	Description	Max # of Hosts Available	Vulnerability Detection List
Nessus	Tenable	Large-scale vulnerability assessment tool with 80,000+ plug-ins designed to access various vulnerabilities	Default – 30 Licensed – Unlimited	Systems, Networks, Applications, Malware, Control Systems, Mobile, etc.
OpenVAS	Open Source	Similar to Nessus, except Open Source	Default-30	Network, Server, and Web Application
Nexpose	Rapid7	Integrates Metasploit for vulnerability assessment	Default – 32 Express & Consultant– 1,024 Enterprise & Ultimate – Unlimited	Browser and Operating Systems
GFI LanGuard	GFI Software	Designed to help with patch management and network/software audits	No default or max number specified	Multi-platform Vulnerability Scans available for Windows, Max, Linux, iOS, Android, Windows Phone, etc.
QualysGuard	Qualys	Offers network discovery, mapping, prioritization, and reporting	Default – 30 Express Lite – 256 Express – 5,120 Enterprise – unlimited	Web-Application, Malware, Firewall, IT systems, etc.
MBSA	Open Source	Checks to see if Microsoft products are secure	64 hosts	Passwords, IIS administration, SQL Server administration, Security, Web-Application, etc.
Retina	BeyondTrust	Assesses and prioritizes vulnerabilities in networks	Community – 256	Network Systems, Web Applications, Databases, Virtual Environments
Nipper	Titania	Audits network configuration files	No default or max number specified	Web Application, Banking and Financial Systems, SSL Scanners, etc.
SAINT	SAINT	Vulnerability assessment	Scans all hosts in a target's subnet	Operating Systems, Databases, and Web Applications
Core Impact	Core Security	Powerful exploitation tool, can import other tools such as Burp Suite, SAINT, etc.	No default or max number specified	Web Application, Password, Mobile Device, Wireless Network, etc.
Secunia PSI	Flexera Software	Free security tool that is able to detect vulnerable and outdated programs and vulnerable plug-ins	No default or max number specified	Hardware, Firmware, Middleware, ICS, etc.

Table 1: Multi-Purpose Vulnerability Assessment Tools

Like multi-purpose tools, there are a plethora of vulnerability assessment tools dedicated solely to web application assessments. Seventeen such tools are reviewed: Burp Suite (Burp), Nikto, Paros Proxy, WebScarab, SQLMap, Skipfish, Acunetix WVS, AppScan, Netsparker, HP WebInspect, Samurai Web Testing Framework, Firebug, Ratproxy, Websecurify, Grendal-Scan, Wfuzz, and Wapiti (Table 2). Of these, Burp, Nikto, Netsparker, and Acunetix can handle large-scale web application vulnerability scans. However, Burp is the only tool with a defined list of built-in web application vulnerabilities to scan. Burp also allows users to create custom plug-ins. Similar to Nessus, Burp categorizes each vulnerability based on its severity into four categories: high, medium, low, or information. In addition to this categorization, Burp will also assign a confidence score (e.g., certain, firm, tentative) to the detected vulnerability. These functionalities make Burp an ideal tool to assess vulnerabilities of SCADA device and scientific instrument web applications.

Tool Name	Company	Description	Max # of Hosts Available	Vulnerability Detection List
Burp Suite	Portswigger	Integrated platform designed to attack web applications	Scans multiple hosts via text file	114 vulnerabilities built in to scan such as SQL Injection, XSS, OS command injection, ASP.NET tracing enabled, File path traversal, etc. Also supports multiple plug-ins.
Nikto	Open Source	Assesses vulnerabilities of web servers	Scans multiple hosts via text file	Accesses online vulnerability database that contain vulnerabilities such as XSS, SQL Injection and CRLF
Paros Proxy	Open Source	Java application to test SQL Injection and XSS	Configures how many hosts you want to scan	XSS and SQL Injection
WebScarab	Open Source	Observes HTTP requests and responses	Supports multiple host scanning	XSS, SQL Injection, CRLF
SQLMap	Open Source	Exploits SQL Injection flaws	Scans multiple hosts via text file	SQL Injection
Skipfish	Open Source	Creates sitemaps of web applications	Supports multiple host scanning	MySQL Injection, XSS, blind SQL or XML injection and blind shell injection
Acunetix WVS	Acunetix	Checks for web app vulnerabilities	Scans multiple hosts through multiple instances	SQL Injection, XSS, XXE, SSRF, Host Header Attacks
AppScan	IBM	Provides secuirty testing for app development lifecycle	Adds multiple AppScann scanners to Qradar	SQL Injection, XSS, CSRF, Improper Error Handling
Netsparker	Netsparker	Web app scanner	Scans multiple hosts by running multiple instances	SQL Injection, XSS, DOM XSS, Command Injection, Blind Command Injection

HP WebInspect	HP	Identifies web app layer vulnerabilities	Can only scan one host at a time	11 vulnerabilities are able to scan such as SQL Injection, XSS, DOM-based XSS
Samurai Web Testing Framework	Open Source	Linux distribution with several web app vulnerability assessment tools built-in	Can only scan one host at a time	SQL Injection, XSS, and Remote File Includes with network attacks such as port scanning
Firebug	Open Source	Firefox add on for editing HTML	Can only scan one host at a time	JS script code injection
Ratproxy	Open Source	Passive web app audit tool	Scans for one host at a time	XSS, XSRF
Websecurify	Open Source	Web app security testing environment	Scans for one host at a time	SQL Injection, Expression Injection and XSS
Grendal-Scan	Open Source	Web app security testing tool	Scans for one host at a time	SQL Injection, XSS, session fixation
Wfuzz	Open Source	Bruteforcing web apps for injections and passwords	Scans for one host at a time	SQL Injection, XSS, LDAP injection
Wapiti	Open Source	Scans for web app forms and injects payloads	Scans for one host at a time	XSS, XXE, CRLF, Database Injection

Table 2: Web-Application Vulnerability Assessment Tools

2.2 Benchmarking Literature

Our review of multi-purpose and web application tools reveals that Burp and Nessus are the most well-maintained and robust tools. Both can perform large-scale scans, assess similar web application vulnerabilities, and categorize detected vulnerabilities into varying levels of severity. Moreover, companies like Northrop Grumman and HP use Nessus while Sandia National Laboratories and Raytheon have used Burp Suite. However, such tools need benchmarking and performance evaluation to determine which tool accurately and efficiently scans SCADA devices and scientific instruments; current literature focuses on benchmarking tools within the same category (e.g. web application to web application). Benchmarking literature indicates that benchmarks must have six attributes [12]:

- **Relevance –** provides a meaningful performance measure within a target domain.
- **Understandable –** provide results that are easy to comprehend.
- **Good metrics –** define linear, orthogonal and monotonic metrics.
- **Scalable –** relevant to a variety of systems based on cost and performance.

- **Coverage –** do not oversimplify the environment.
- **Acceptable –** presents impartial results that are accepted by the industry.

Assuming these requirements are met, the tools test procedure must include two key activities: accuracy assessment and results verification [13]. Accuracy assessment is when each scanner is tested against tools like the Web Application Vulnerability Scanner Evaluation Project (WAVSEP), an evaluation platform that determines the accuracy of web application scanners. The setup of WAVSEP was done with the following steps [16]:

- Download and install Apache Tomcat 7.x
- Download and install MySQL Community Server 5.5x
- Copy the wavsep.war file into the tomcat webapps directory (Usually "C:\Program Files\Apache Software Foundation\Tomcat 6.0\webapps" - Windows 32/64 Installer)
- Restart the application server
- Initiate the install script at: http://localhost:8080/wavsep/wavsep-install/install.jsp
- Provide the database host, port and root credentials to the installation script (username and password of the root database user (mysql).
- Access the application at http://localhost:8080/wavsep/

Each scanner is then tested against a variety of false positive scenarios and test cases (e.g., XSS, SQL Injection, Path Traversal, etc.). Next, the input vectors are determined such as the more input delivery of scanner reports, the more versatility of the scanning applications. Therefore, after testing Burp and Nessus' performance during the WAVSEP evaluation, it will be easier to determine the accuracy of their results from scanning SCADA and scientific instruments.

Web Input Vector Extractor Teaser (WIVET) can be used to assess crawling coverage and crawling capabilities as well as to assess the attack vector coverage [14]. In order to implement WIVET, the following steps were taken [17]:

- Download WIVET from https://github.com/bedirhan/wivet
- Extract the contents of the zip file from your web server (htdocs\wivet for Apache Tomcat)
- Browse the WIVET site to make sure that it is up and running (e.g. http://127.0.0.1/wivet)

Some companies such as Portswigger, Rapid7, IBM, and OWASP have used WAVSEP/WIVET to evaluate their own tools. After the WAVSEP/WIVET evaluation, we focus on verifying our results. The Results verification stage focuses on scanning categories multiple times to ensure accuracy of results [15]. Additional benchmarking applications will test the tools (e.g., WAVSEP, WIVET, etc.) to ensure accuracy of the scan results [13, 15].

3 RESEARCH GAPS AND QUESTIONS

We identified several research gaps from our review of vulnerability assessment tools and benchmarking literature. Although Nessus and Burp are the premier multi-purpose and web assessment tools (respectively), they have not been benchmarked against one another. As such, it is unclear how well these tools compare in terms of accuracy, scalability, and overall performance. Furthermore, minimal literature has attempted to understand the vulnerabilities afflicting SCADA devices and scientific instruments. Based on these gaps, the following research questions are posed for study:

- How can Burp and Nessus be used for large-scale vulnerability assessments?

- How consistent are Burp and Nessus in providing vulnerability scan results?
- Which tool should be used to scan SCADA devices and/or scientific instruments?

3.1 Research Testbed

As we are aiming to benchmark Nessus and Burp in the context of SCADA devices and scientific instruments, our research testbed has two major components. The first part of our testbed contains details about 20,641 SCADA devices from [11]. Devices include various vendors such as Siemens, Rockwell Automation, Schneider Electric, etc. The most common ports used by SCADA devices include web services (e.g., 80, 8080, 443, etc.), SCADA specific (e.g., 502, 44818, etc.) and general service (e.g., Telnet, FTP, etc.). The second component of our testbed contains scientific instrument related devices. Our team has partnered with a local institution that hosts a variety of computing related scientific instruments for life sciences. Our partner institution granted our team permission to scan three of their publicly available IP ranges. Table 3 details the number of devices identified on Shodan. To protect the privacy of our partner, we will refer to each range as /23, /24/ or / 25.

IP Range	**Number of Possible Devices**	**Number of Shodan Devices**	**Percent Exposure**	**Most Popular Protocol on Each Range**
/23	512	159	31.055%	SSH (70.44%)
/24	256	17	6.641%	HTTP/HTTPS (88.24%)
/25	128	6	6.25%	HTTP/HTTPS (75.00%)
Total:	**896**	**184**	**20.536%**	**-**

Table 3: CyVerse IP Ranges

Given the characteristics of our dataset, we constructed three major components of our research design: (1) vulnerability assessment using Burp, (2) vulnerability assessment using Nessus, and (3) benchmarking of both tools. Details of each component are illustrated in Figure 2 and detailed in the following subsections.

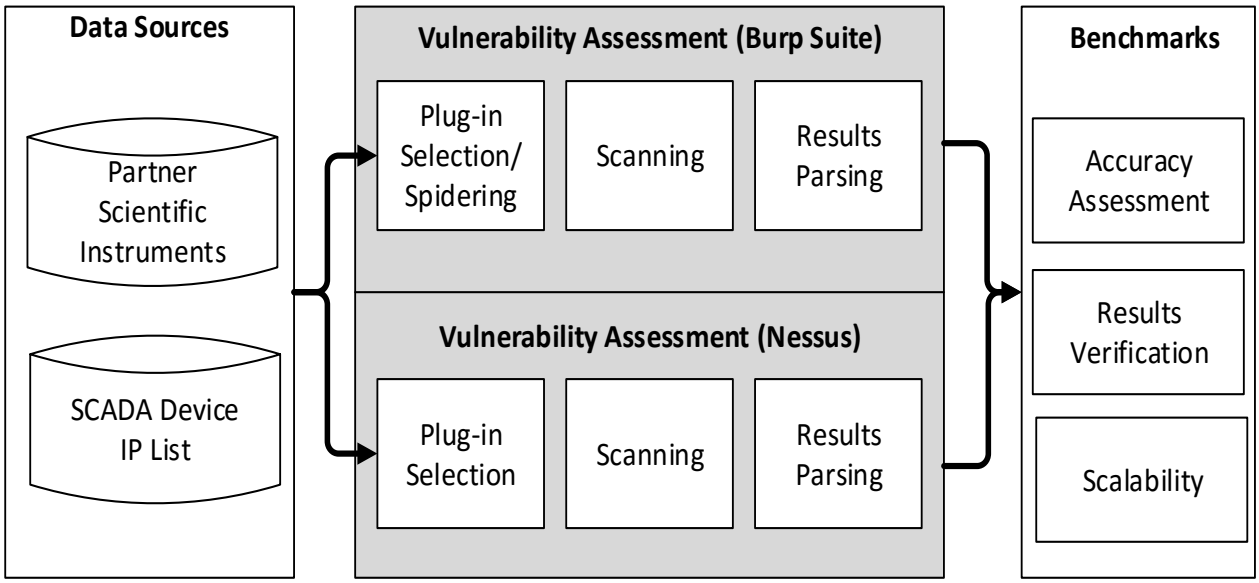

Figure 2: Burp Suite and Nessus Vulnerability Assessment Framework

3.2 *Burp Suite and Nessus Vulnerability Assessments*

Burp was configured to scan files of associated IP's and ports from specified devices and used appropriate plug-ins to identify vulnerabilities. All scans from Burp were done over a Kali Linux virtual machine (VM). However, for Burp to perform these scans, it had to be automated utilizing the Integris extension package, Carbonator. Carbonator allows the user to control Burp through the command line: which allows the user to initiate custom scans. Once the scans are completed, Carbonator returns a HTML file with the scanned results. However, through viewing the Carbonator code and speaking with individuals from Integris, it was determined that some of the functionality

was hardcoded and could not be modified. For example, the user is unable to scan multiple IPs at once, change the name of the output files, or change the output from HTML to CSV. Thus, a few Python scripts were written to enable Burp to scan multiple files at once, change the name of the output file per the user's will, as well as change the output format (Figure 3). The overall design of Burp's automation is shown in Appendix A: Burp Automation.

```
## Read first line of Text File and Run Burp with http/https address

        f = open("Websites.txt", "r")
        line = f.readline()
        os.system("bash /root/carbonator-master/launch_burp.sh " + line)

## If the file is not empty, keep running Burp with http/https addresses

        while line:

                line = f.readline()

                os.system("bash /root/carbonator-master/launch_burp.sh " + line)

                os.chdir("/root")

                for file in glob.glob("Integris*.html"):

                        os.system("python csvConverter.py " + file)


                dst = '/root/CSV_Files/'
                # create destination directory, if needed (similar to mkdir -p)
                for txt_file in glob.iglob('Int*.csv'):
                        shutil.move(txt_file, dst)
                dst = '/root/BurpTest/'
                for txt_file in glob.iglob('Int*.html'):
                        shutil.move(txt_file, dst)


        f.close()
```

Figure 3: Python File for Burp Automation

Also, Burp performed scans over Tor: which allows scans to be performed undetected. By doing so, Burp will not drastically harm the target device. However, to connect Tor over HTTP, Privoxy is needed to be installed on the VM where Burp was set up. To set up Privoxy on Kali Linux, the following commands were used: # apt-get update, #apt-get install Tor privoxy. In the /etc/privoxy/config file for Privoxy, the following line is added in order to create a socks port

connection: # echo "forward-socks4a / 127.0.0.1:9050." >> /etc/privoxy/config. Once this is accomplished, under /etc/tor/torrc, SocksListenAddress 127.0.0.1 and a SocksPort 9050 need to be added. By doing so, Burp can use a SOCKS Proxy setup through Tor to perform scans on the SCADA devices (Figure 4).

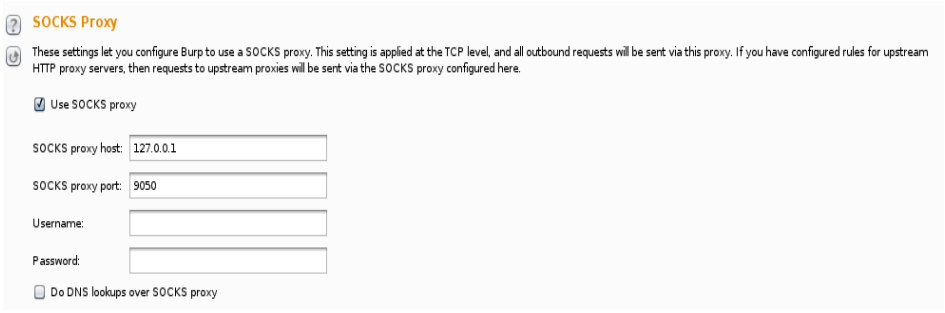

Figure 4: Proxy Set Up For Burp Suite

Nessus also searched for SSH, web application/server (e.g., outdated software, XSS, SQL injection), general Windows/Linux vulnerabilities, and default credential issues. Nessus was also automated in a similar fashion as Burp Suite by Sagar, a fellow Scholarship for Service student. However, since Nessus is unable to run via Tor or socks, the same anonymity achieved for Burp's scans was unable to be implemented. Therefore, ports were not scanned because such activity could cause unintentional harm to systems. Scan results from Nessus and Burp are parsed into a database.

3.3 Benchmarks

Consistent with standard practices, scans were benchmarked in three fashions: accuracy assessment, result verification, and scalability as shown below (Table 4).

Benchmark Name	**Description of Benchmark**
Accuracy Assessment:	
WAVSEP Evaluation	Determine the accuracy of Nessus and Burp, such as what vulnerabilities can be detected by each tool.
False Positive Review	Burp and Nessus is tested against a variety of false-positives to determine which tool to see if they can detect the error

WIVET Evaluation	Assess Burp and Nessus' crawling coverage
Result Verification	Each scan is run multiple times to determine accuracy of results as well as document each case

Table 4: Benchmark Descriptions

4 RESULTS AND DISCUSSION

The results are organized into three subsections. The first subsection focuses on comparing and contrasting the identified SCADA vulnerabilities from Nessus and Burp. The vulnerabilities are subsequently detailed and identified from our scientific instrument vulnerability scans. Finally, the benchmarking is summarized (WAVEP, WIVET, and scalability) results.

4.1 Nessus and Burp SCADA Vulnerabilities

Burp found 100 devices out of a possible 1,182 (8.46%) running web protocols with web vulnerabilities. Devices stemmed from vendors like Siemens, Echelon Corp., and Moore Industries. Table 5 details the number of vulnerabilities detected at each level for SCADA devices in port 80 and 8080:

Severity	Vulnerabilities Found	Example Vulnerabilities Found
High	10	Cross Site Scripting (Reflected)
Medium	3	Cross-site Request Forgery
Low	81	Unencrypted Communications

Table 5: Burp Suite SCADA Vulnerabilities by Severity

Devices in the "High" level either have cleartext submission of password or reflected cross-site scripting. For example, one device enabled passwords to be transmitted over unencrypted connections while two others enable attackers to potentially inject browser executable code within

a single HTTP response. Devices in “Medium” category have cross-site request forgery, allowing users to transmit unauthorized commands from a trusted website. Furthermore, “Low” risk level had 81 devices that were all unencrypted communications. For example, five devices enable an attacker to view a user’s network traffic and record and monitor their interactions with applications.

Compared to Burp’s 100 device vulnerabilities, Nessus found 60 devices (3.38%) with web vulnerabilities. Table 6 details the number of vulnerabilities found in each range.

Severity	Vulnerabilities Found	Example Vulnerabilities Found
Critical	0	None Found
High	4	Dropbear SSH Server < 2016.72 multiple vulnerabilities found
Medium	13	Unencrypted Telnet Server, DNS Server Recursive Query Cache Poisoning Weakness, DNS Server Cache Snooping Remote Information Disclosure
Low	11	SSH Server CBC Mode Ciphers Enabled, SSH Weak MAC Algorithms Enabled

Table 6: Nessus SCADA Vulnerabilities by Severity

All devices in the “High” level have Dropbear SSH Server vulnerabilities. Four devices have a series of flaws in which an attacker can discloser process memory and execute arbitrary code with root privileges. Devices in “Medium” category have 13 cases of Unencrypted Telnet Server, DNS Server Recursive Query Cache Poisoning Weakness, and DNS Server Cache Snooping Remote Information Disclosure. Two of these devices enable usernames, passwords and commands to be transferred over cleartext through the Telnet vulnerability, allowing an attacker to perform cache poisoning attacks against a name server.

Attackers can also discover if the DNS server has a specific record cached to determine if the owner has recently visited a specific site for the DNS Snooping vulnerability. The primary vulnerability in the “Low” risk level had 11 devices that have SSH Server CBC Mode Ciphers Enabled and SSH Weak MAC Algorithms Enabled. Such issues can allow attackers to recover a

plaintext message from ciphertext for the CBC mode vulnerability, and allows for weak encryption algorithms (MD5 of 96-bit MAC).

Despite the differences in the detected vulnerabilities by Burp and Nessus, both found unencrypted connection vulnerabilities and cleartext submission of password. Burp determined IP ranges with unencrypted communications whereas Nessus determined that the unencrypted communications stemmed from a Telnet Server. Also, Burp determined that some IP ranges enabled passwords to be transmitted over unencrypted networks while Nessus found that the Unencrypted Telnet Server enabled passwords to be transferred over cleartext.

4.2 Nessus and Burp Scientific Instrument Vulnerabilities

During the scientific instrument vulnerability scans, Burp found 21/184 (11.4%) of the IP's with web vulnerabilities. Table 7 details the number of vulnerabilities found in each severity range. For the sake of space, the 19 devices were omitted that had informational vulnerabilities.

Severity	Vulnerabilities Found	Example Vulnerabilities Found
High	2	Cross Site Scripting, HTTP response header injection
Medium	0	None Found
Low	0	None Found

Table 7: Burp Scientific Instrument Vulnerabilities by Severity

All devices in the "High" level have both cross-site scripting (XSS) and HTTP response header injection vulnerabilities. XSS enables attackers to inject client-side scripts into web pages viewed by other users. Such scripts can significantly damage the website and allow attacker to potentially control the entire machine. HTTP response header injection allows attackers to poison the cache of any proxy server when the user accesses the application. Burp did not detect any "Medium" or "Low" vulnerabilities from its analysis. Conversely, Nessus found 75/184 (40.76%) devices with

vulnerabilities. This is higher than the 21 found by Burp. Table 8 details the vulnerabilities found in each risk level.

Severity	Vulnerabilities Found	Example Vulnerabilities Found
Critical	2	PHP Unsupported Version Detection, Unix OS Unsupported Version Detection
High	3	Apache HTTP Server Byte Range DoS, Multiple PHP Vulnerabilities
Medium	86	SSH Weak Algorithms Supported, Apache Server Etag Header Information Disclosure, HTTP TRACE/TRACK Methods Allowed, Browsable web Directories
Low	153	SSH Server CBC Mode Ciphers Enabled, SSH Weak MAC Algorithms Enabled, Web Server Transmits Cleartext Credentials, Web Server Uses Basic Authentication Without HTTPS, X Server Detection

Table 8: Nessus Scientific Instrument Vulnerabilities by Severity

Devices in the "Critical" level have PHP Unsupported Version Detection and Unix OS Unsupported Version Detection issues. Unsupported PHP software can enable anyone to retrieve sensitive information, such as memory usage. Outdated Unix indicates the operating system running that host is no longer supported and thus has no new security patches.

Devices in "High" have Apache HTTP Server DoS, Multiple PHP Vulnerabilities. "Medium" risk level has 86 devices that were vulnerable to SSH Weak Algorithms Supported, Apache Server Etag Header Information Disclosure, HTTP TRACE/TRACK Methods Allowed, and Browsable web Directories. For example, one device has an SSH server that is configured to allow weak encryption algorithms, is providing sensitive information due to an Etag header. This device also enables debugging functions to be enabled on a remote server as well as enables the attacker to browse user directories, and determine which programs are installed as well as their versions.

Similar to the SCADA scans, there were some overlap between the Nessus and Burp scans for scientific instruments. Nessus noted there were multiple PHP Vulnerabilities. Burp found similar XSS issues. Burp and Nessus both found HTTP Trace/Track Methods allowed or enabled in /24.

4.3 Benchmarks: WAVSEP, WIVET, and Scalability

After vulnerability scans on SCADA devices and scientific instruments, the accuracy of Burp and Nessus were analyzed using WAVSEP. This tool is used solely to determine the accuracy of finding common web-application vulnerabilities and is separate from the SCADA and scientific instrument analysis. Through this analysis, we can determine whether the aforementioned web-application results from Nessus and Burp are accurate. Table 9 details the 1,200 vulnerabilities in web applications tested by Burp and Nessus.

Vulnerability	Test Cases	Description
Local File Inclusion (LFI)	816	Includes files on a server through a web browser, capable of allowing for directory traversal characters to be injected.
SQL Injection	130	Used to attack data-driven applications by inserting SQL statements into an entry field for execution
Remote File Inclusion (RFI)	108	Enables attacker to run malicious code on the server
Cross Site Scripting (XSS)	64	Enables attackers to inject client-side scripts into web pages
Open Redirection	60	Security flaw that enables a web page to fail properly authenticating URL's
Unreferenced Files	22	Grant an intruder access to inner workings, back doors, administrative interfaces by accessing these files to gain knowledge about the infrastructure or credentials

Table 9: WAVSEP Vulnerabilities

In Figure 5, Burp, indicated in orange, outperformed Nessus in detecting common web application vulnerabilities. Burp performed 2.34 times better than Nessus. There were 936/1200 (78%) vulnerabilities found by Burp versus 399/1,200 (33.3%) vulnerabilities found by Nessus.

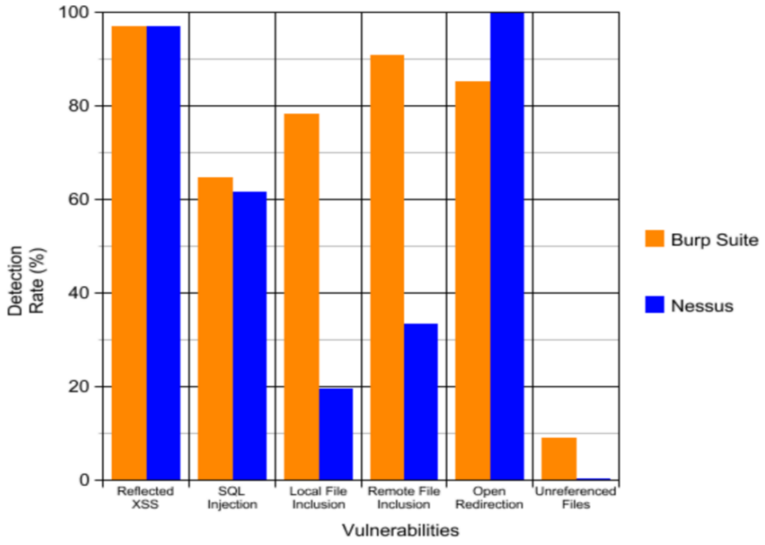

Figure 5: WAVSEP Results

In Figure 6, Nessus, indicated in blue, generated more false positives than Burp. Nessus generated 2.39 times more false positives than Burp. Consequently, Burp incorrectly determined 5/44 (11.4%) of its cases while Nessus incorrectly determined 12/44 (27.3%) of its cases.

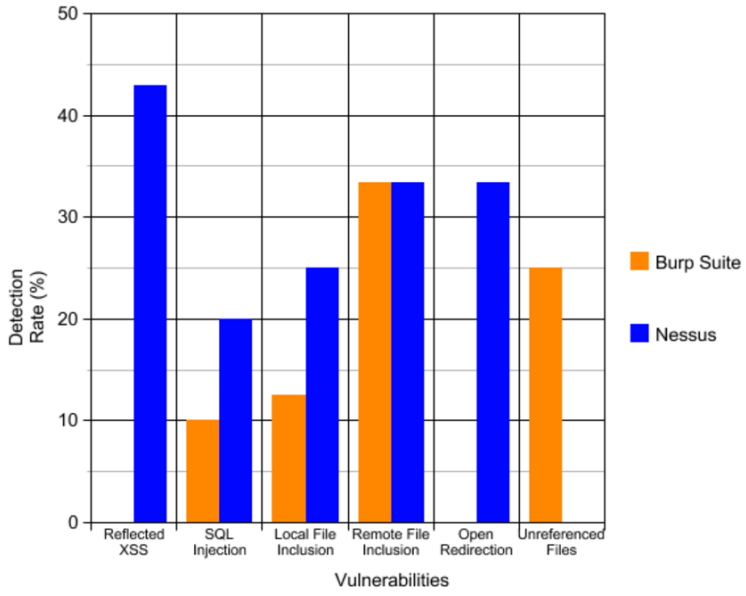

Figure 6: WAVSEP False-Positive Results

In Figure 7, WIVET assessed the capability and accuracy of Burp and Nessus in crawling websites. Burp was able to identify 53% of the content. However, Nessus was able to identify 23% of the content.

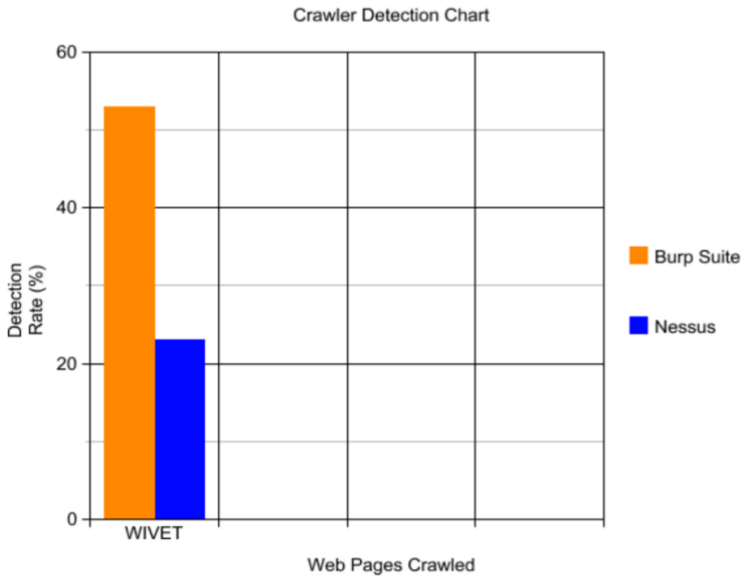

Figure 7: WIVET Results

The aforementioned results indicate that Burp out-performed Nessus in vulnerability assessments in terms of accuracy and false positive detection. Burp detected more vulnerabilities in test cases then Nessus and performed well in XSS, LFI, RFI, and Open Redirection. Thus, when Burp found XSS vulnerabilities in SCADA devices, they are accurate based on the WAVSEP results. Conversely, Nessus returned more false-positives than Burp. In XSS, SQL Injection, LFI, and Open Redirection, Nessus incorrectly determined cases that were not present. Furthermore, Burp Suite outperformed Nessus in crawl coverage. Burp had 2.3 times more coverage than Nessus.

Lastly, Burp and Nessus were assessed for their scalability in performing large-scale vulnerability assessments. Burp was not created to scan millions of devices at once. Instead, it is used to determine web application vulnerabilities one website at a time. By running multiple

instances of Burp on Kali Linux with more processing power, Burp is able to scan web applications faster. Nessus enabled the scanning of a potentially unlimited amount of hosts at once.

Based on the scalability analysis, the following results occurred when scanning SCADA and scientific instrument IPs. Using one virtual machine (4 CPUs, 16 GB RAM), Burp completed its scan of 1,182 SCADA IPs in 12 hours. Conversely, Nessus, using one machine (4 CPUs, 16 GB RAM), completed the same scan of 1,182 IPs in 8 hours. Regarding scientific instruments, utilizing one virtual machine (4 CPUs, 16 GB RAM), Burp completed its scan of 184 scientific instrument devices in 6 hours. Nessus completed the same scan on one machine (4 CPUs, 16 GB RAM) in 3 hours.

Results indicate Nessus outperforms Burp in terms of speed (Table 10). When scanning 12,000 devices, Nessus completed scans in 15 hours compared to the 24 hours needed by Burp. Such results were similar when scanning 25,000 devices; Nessus completed in 30 hours, while Burp finished in 48 hours.

Tool	# of IP's Scanned	CPU Usage	Completion Time
Burp	12,000	105%	24 hours
	25,000	120%	48 hours
Nessus	12,000	25%	15 hours
	25,000	100%	30 hours

Table 10: Scalability of Burp Suite

5 CONCLUSION

This research aims to benchmark state-of-the-art vulnerability assessment tools in the context of SCADA devices and scientific instruments. Results indicate that Burp is the ideal tool to scan solely for web application vulnerabilities for SCADA and scientific instruments. Burp Suite could find web application vulnerabilities (e.g., XSS, Cross-site Request forgery, and Header Injection) that Nessus did not. WAVSEP and WIVET determined that Burp's results are more accurate than Nessus, and Burp generated fewer false positives. However, Nessus could scale more effectively than Burp in terms of scan completion time.

If an organization is scanning SCADA and scientific instrument devices for solely SCADA/OS related vulnerabilities, Nessus is the preferred tool. Nessus found vulnerabilities specific to SCADA devices and certain OS and was able to identify devices, something Burp could not achieve. Examples of vulnerabilities found by Nessus are Dropbear SSH vulnerabilities, DNS Server Recursive Query Cache Poisoning Weakness, PHP Unsupported Version Detection, and Unix OS Unsupported Version.

Nessus and Burp can also be used in concert with one another to obtain a complete vulnerability analysis of an organization's systems. Nessus can perform scans to classify devices, identify vulnerabilities apparent in specific OS, etc. while Burp can perform a more comprehensive web vulnerability assessment. Conversely, Burp can identify an issue at hand (e.g., unencrypted communications) and Nessus can identify the server.

There are several promising avenues for future research. First, we can expand our methodology to include other categories of emerging devices (e.g., medical, general IoT). Future research can aim to provide automated reports and mitigation strategies to the device owners to enhance overall security posture for their vulnerable devices. Each extension would provide a

deeper understanding into the performance of vulnerability assessment tools in various high-impact contexts.

6 ACKNOWLEDGEMENTS

This material is based upon work supported in part by the National Science Foundation (DUE-1303362). Also, I would like to thank Emma McMahon, Hsinchun Chen, Mark Patton, and Sagar Samtani for aiding me with my research.

7 REFERENCES

[1] P. Wood, B. Nahorney, “Internet security threat report”, *SYMANTEC*.

[2] R. Nigam, “SCADA security report 2016”, *Fortinet Blog*, Apr. 2016.

[3] S. Kanowitz, “Cyber threats for energy, transportation sectors on the rise”, *GCN*, Jun. 2016.

[4] L. Farrell, “Securing the scientific workflow”, *Science Node*, Mar. 2016.

[5] R. Krishnan, "NASA hacked! AnonSec tried to crash $222 million drone into pacific ocean", *The Hacker News*, Feb. 2016.

[6] D. Livingstone, "Opinion: we must defend against cyberattacks in space", *Newsweek*, Nov. 2016.

[7] A. Nikolich, "Cybersecurity innovation for cyberinfrastructure (CICI) ", *Nsf.gov*, Apr. 2016.

[8] N. Rathaus, "Vulnerability Assessment Whitepaper", *Beyond Security*.

[9] S. Drew, "Vulnerability Assessments versus Penetration Tests", *Secureworks.com*, Apr. 2015.

[10] M. Patton, E. Gross, R. Chinn, S. Forbis, L. Walker, H. Chen, “Uninvited connections: a study of vulnerable devices on the internet of things (ioT)”, *2014 IEEE Joint Intelligence and Security Informatics Conference*, 232-235.

[11] S. Samtani, S. Yu, H. Zhu, M. Patton, H. Chen, “Identifying scada vulnerabilities using passive and active vulnerability assessment techniques”, *The University of Arizona.*

[12] R. Angles, P. Boncz, J. Larriba, N. Martinez, B. Bishop, “Benchmark Principles and Methods”, *LDBC*, 2013.

[13] S. Chen, “The web application vulnerability scanners benchmark”, *Denim Group,* 2014.

[14] Y. Smeets, “Improving the adoption of dynamic web security vulnerability scanners”, *In Dei Nomine Feliciter,* 2015.

[15] D. Cornell, “Benchmarking web application scanners for your organization”, *Denim Group*, 2012.

[16] "sectooladdict/wavsep", *GitHub*, 2017. [Online].

[17] N. Sciberras, "How to Configure Acunetix WVS to Successfully Crawl WIVET - Acunetix", *Acunetix*, 2017. [Online].

8 APPENDIX A – Burp Automation