Identifying SCADA Systems and Their Vulnerabilities on the Internet of Things

A Text-Mining Approach

Sagar Samtani
University of Arizona

Shuo Yu
University of Arizona

Hongyi Zhu
University of Arizona

Mark Patton
University of Arizona

John Matherly
Shodan

Hsinchun Chen
University of Arizona

Supervisory Control and Data Acquisition (SCADA) systems allow operators to control critical infrastructure. Vendors are increasingly integrating Internet technology into these devices, making them more susceptible to cyberattacks. Identifying and assessing vulnerabilities of SCADA devices using Shodan, a search engine that contains records about publicly available Internet-connected devices, can help mitigate cyberattacks. The authors present a principled approach to systematically identify all SCADA devices on Shodan and then assess the vulnerabilities of the devices with a state-of-the-art tool.

Supervisory Control and Data Acquisition (SCADA) devices allow operators to supervise, maintain, control, and collect data from critical infrastructure such as power plants and transportation systems.[1] Today, many SCADA devices from popular manufacturers such as Siemens, Rockwell Automation, and Schneider Electric are connected to the Internet, meaning they can be operated remotely. Despite the significant benefits in convenience, this connectivity increases their susceptibility to cyberattacks such as malware, buffer overflow, and denial of service (DoS) attacks.[1] Such weaknesses have caused a 110-percent increase in SCADA-related cyberattacks from 2015 to 2016.[2]

Published by the IEEE Computer Society
1541-1672/18/$33.00 ©2018 IEEE

Shodan is a search engine that contains information about publicly available Internet of Things (IoT) devices—it scans, indexes, and provides port and textual banner data for more than 600 million devices on the IPv4 range. As a result, it can be used by malicious attackers to discover SCADA devices to exploit. However, Shodan can also be used to identify and assess vulnerabilities of SCADA devices in an attempt to mitigate cyberattacks.

Figure 1 illustrates how a user can leverage a SCADA query (in this case, a Rockwell Automation device) on Shodan's web interface to retrieve information about SCADA systems and access a SCADA system's unprotected web interface.

Academics and security professionals often manually identify SCADA devices because of Shodan's large data size and textual content. These manual analyses are slow, labor-intensive, not scalable, and provide an incomplete picture of the SCADA vulnerability landscape. To address some of these issues, our research aims to develop a principled approach that systematically analyzes device banner data to identify all SCADA devices on Shodan. We also assess the vulnerabilities of identified devices with Nessus, a state-of-the-art vulnerability assessment tool.

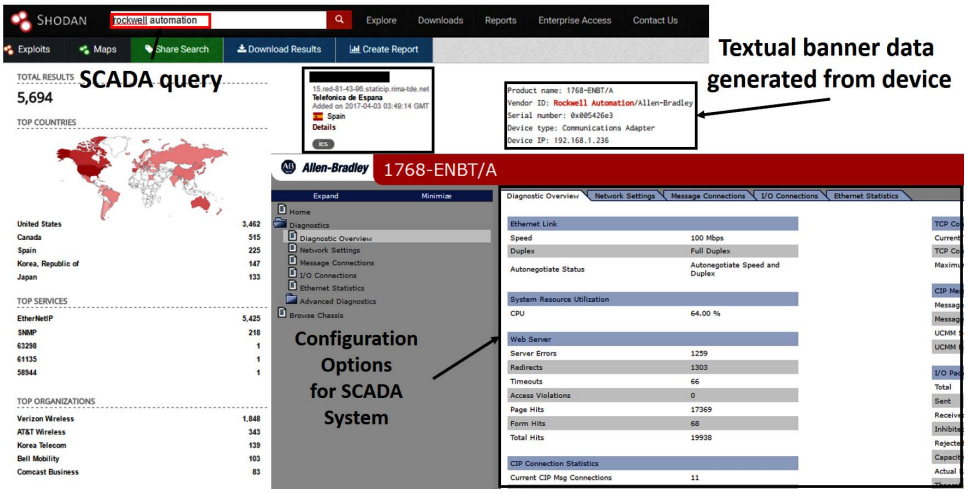

Figure 1. Shodan results for a Rockwell Automation Supervisory Control and Data Acquisition (SCADA) device. Shodan provides information about ports open on each device as well as the banner data generated from the ports.

RELATED WORK

Networks with SCADA systems (see Figure 2) have three major components: a corporate local area network (LAN), a SCADA-specific component, and a field device network.[1] The corporate LAN contains standard workstations and servers for day-to-day business operations. The SCADA network comprises human–machine interfaces (HMIs), SCADA servers, and historians.

SCADA operators utilize the HMIs in the SCADA network to control programmable logic controllers (PLCs), remote terminal units (RTUs), master terminal units (MTUs), and intelligent electronic devices (IEDs) at the field device layer. PLCs, RTUs, and IEDs receive commands from the HMIs to manage the infrastructure (for example, to maintain dam water levels). Such devices utilize proprietary communication protocols such as the Distributed Network Protocol (DNP3), the Common Industrial Protocol (CIP), and/or Modbus.[1]

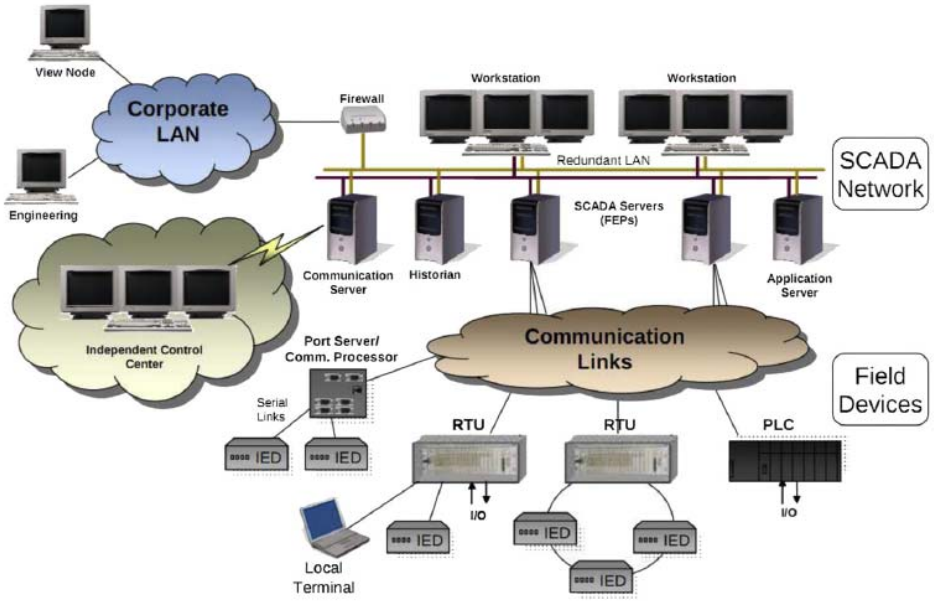

Figure 2. Layout of a network with SCADA systems. (Source: A. Nicholson et al., "SCADA Security in the Light of Cyber-Warfare," *Computers & Security*, vol. 31, no. 4, 2012, pp. 418–436.[1] Used with permission.)

SCADA security concerns have traditionally focused on mitigating local insider threats. However, the integration of TCP/IP protocols in the past 15 years has allowed attackers to conduct attacks both remotely and locally. This has significantly expanded the attack surface to include distributed denial of service (DDoS) attacks, buffer overflows, and memory exploits.[1] SCADA systems with web interfaces are susceptible to SQL injection, XSS attacks, password issues, and HTML injections. Openly accessible proprietary SCADA protocols such as DNP3, CIP, and Modbus also offer cyberattack avenues. These concerns are further exacerbated by the development of Shodan.

Despite well-documented SCADA security concerns, we only found two studies identifying vulnerabilities of SCADA devices on Shodan.[3,4] Co-author Mark Patton and his colleagues developed password-testing scripts to identify 45 out of 1,258 i.LON devices (3.5 percent) utilizing default telnet credentials.[3] Co-author Sagar Samtani and his colleagues performed Nessus scans to discover that 4,009 out of 20,461 devices (19.59 percent) from Rockwell Automation, Siemens, and Schneider Electric suffered from default credentials, outdated software, and Modbus exposure issues.[4] Despite their value, both studies analyzed only a sample of SCADA devices from Shodan rather than identifying and assessing all SCADA devices. As a result, these studies provide an incomplete view of the SCADA vulnerability landscape.

Nevertheless, both studies noted that data-mining and text-mining techniques can aid in identifying all SCADA devices in the Shodan database. In particular, Samtani and his colleagues observed that SCADA devices generate similar types of banner data (such as service information), and that carefully studying this data would provide valuable cues to identify SCADA devices that could be missed in a keyword search. However, processing textual banner data to identify devices in a multi-million record dataset is non-trivial and remains an open issue in ongoing SCADA analyses.[5]

Prevailing text analyses such as sentiment analysis, named entity recognition, and co-reference resolution assume natural language inputs. In these tasks, preprocessing steps often include removing stop words, stemming, and part-of-speech tagging. However, such steps are not generalizable to device-generated data (for example, Shodan banner data and computer log files). Device-generated data does not have the same characteristics of natural language, often containing version information and standardized response codes.[6,7] As a result, past data-mining and

text-mining work utilizing device-generated data tokenized text on white space to generate unigram and bigram representations. These representations were input into k-means and support vector machines (SVMs) to create domain-specific ontologies for server logs[6,7] and signature patterns of operating systems and other technologies,[8,9] or to identify general characteristics of log files.[10,11] Accuracy, precision, recall, and f-measure metrics were used to evaluate selected approaches. Such metrics are standard evaluations in data-mining literature.[12]

Utilizing text-mining and data-mining procedures can help efficiently and effectively process Shodan data to identify all SCADA devices. While this alone can provide a comprehensive look at the exposure of SCADA systems available on the open Internet, vulnerability assessment approaches are required to evaluate detected devices' cybersecurity posture.

Fundamentally, a vulnerability is "a flaw within a system, application, or service which allows an attacker to circumvent security controls and manipulate systems in ways the developer never intended."[13] Organizations utilize vulnerability assessments to identify and mitigate security issues within their technology. Today, there are three categories of vulnerability assessment tools: multi-purpose, web application, and specialty. Multi-purpose tools (such as Nessus and Qualys) focus on assessing the vulnerabilities of a broad range of devices, software, and services. Web application scanners such as Burp Suite detect issues only within web technologies. Specialty tools only find issues within certain technologies such as MySQL.

Among various options, professionals have cited Nessus as "one of the most comprehensive and widely deployed."[14] Nessus has more than 80,000 plugins to assess the vulnerability of a multitude of technologies on networks with thousands of devices. Nessus can identify web application flaws, discover outdated software, test default credentials, and find database issues. Unlike other tools, Nessus offers 286 plugins to identify SCADA vulnerabilities. Nessus categorizes vulnerabilities into risk thresholds of critical, high, medium, low, and information. Scores are based on the industry standard Common Vulnerability Scoring System (CVSS). CVSS factors in vulnerability age, known exploits for vulnerability, and device type in its calculations.

RESEARCH FRAMEWORK

Our review indicated that identifying and assessing the vulnerability of all SCADA devices on Shodan in a scalable, automated fashion is an open issue. Our research design (see Figure 3) aims to help address this gap with two major components: SCADA device classification and vulnerability assessment.

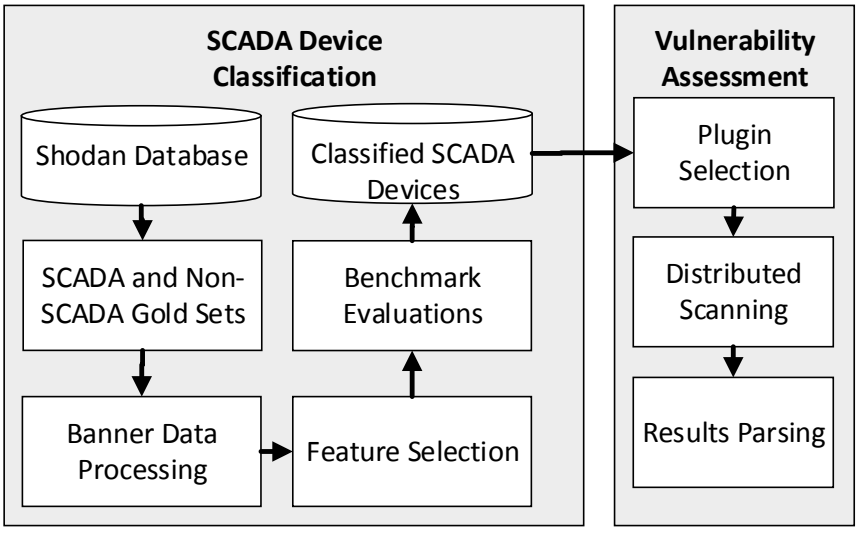

Figure 3. Research framework for identifying and assessing the vulnerabilities of SCADA devices from Shodan.

Shodan Device Classification

The SCADA device classification component of our framework utilizes text processing and data mining to classify whether a device found in Shodan is a SCADA device. Our research group partnered with Shodan's developer, John Matherly, to gain access to the entire Shodan dataset—about 627 million device records. Shodan offers three categories of attributes for each device: device, network, and location. Device-specific attributes include ports open on the machine, banner data, and device type. The network category includes IP address, host names, and Internet service provider (ISP). The location category includes country, latitude, and longitude. Although the device type field could help identify SCADA devices, only 2.03 percent of records were populated. Other attributes such as location or ISP do not aid in SCADA device identification. However, fields such as port and banner data are fully populated and contain valuable cues to identify SCADA devices. Table 1 illustrates examples of banner data for selected ports.

Table 1. Sample banner data from selected ports.

Port number	Protocol	Banner data
80 (Non-SCADA)	Hypertext Transfer Protocol (HTTP)	HTTP/1.1 200 OK Date: Thu, 22 Oct 2015 05:26:32 GMT Server: IPC@CHIP Content-Type: text/html Transfer-Encoding: chunked Last-Modified: Fri, 07 Mar 2014 05:13:30 GMT
21 (Non-SCADA)	File Transfer Protocol (FTP)	220 FTP server is ready. 230 Guest login ok, access restrictions apply. 214- The following commands are recognized (* =>'s unimplemented). ABOR… 500 Unknown command.
102 (SCADA)	Transport Service Access Point (ISO TSAP)	Copyright: Original Siemens Equipment PLC name: SIMATIC 300 Module type: CPU 313C-2 DP Unknown (129): Boot Loader A Module: 6ES7 313-6CG04-0AB0 v.0.2 Basic Firmware: v.3.3.7 Module name: CPU 313C-2 DP

Following standard classification procedures, we first created gold-standard sets of SCADA and non-SCADA devices. To do so, we identified 52,506 devices from major SCADA vendors such as Siemens and Rockwell Automation for our SCADA gold-standard set. The non-SCADA set included 66,536 device records on routers, printers, webcams, web servers, honeypots, databases, and more. A domain expert validated the selected records as being accurate and representative of the breadth of devices available on Shodan.

Similar to past studies examining device data, we tokenized banner data text based on white space to generate n-grams.[6–11] Alphanumeric and special characters are treated as components of tokens. Tokenizing in this manner retains valuable data such as device version and vendor names. After tokenization, we generated unigram and bigram lists for each device. From these

lists, we constructed a signature set containing all n-grams for all known SCADA devices. Specifically, we generated the n-gram signature set by taking the difference of the top 1,000 most frequent SCADA n-grams and the top 1,000 most frequent non-SCADA n-grams. Intuitively, a device with more n-grams in the SCADA signature set is more likely to be a SCADA device.

SCADA signature set unigram examples include "modbus" and "Schneider," while bigram examples include "rockwell automation" and "siemens s7." We also constructed signature sets for each port to capture a finer representation of each device's services. Similar to the general set, the port-specific signature sets are generated by taking the difference of the set of the top 100 most common n-grams by SCADA devices on that port and the set of the top 100 most common n-grams by non-SCADA devices. All of these operations were performed with Python, primarily with the regular expression package. Formally, our signature set creation process is denoted as:

SS_n as the n-gram signature set for SCADA devices over all ports;

SC_n as the set of top 1,000 common n-grams by SCADA devices over all ports;

NC_n as the set of top 1,000 common n-grams by non-SCADA devices over all ports;

$SS_{n,p}$ as the n-gram signature set for SCADA devices over a specific port p;

$SC_{n,p}$ as the set of top 100 common n-grams by SCADA devices over a specific port p;

$NC_{n,p}$ as the set of top 100 common n-grams by non-SCADA devices over a specific port p.

Then,

$$SS_n = SC_n - NC_n \text{ for } n \in \{1,2\};$$

$$SS_{n,p} = SC_{n,p} - NC_{n,p} \text{ for } n \in \{1,2\}; \text{ and } p \in \{\text{port number} \mid \text{all possible ports in dataset}\}.$$

Each signature set is one feature. Other features (see Table 2) include port number, banner length, and number of banner tokens.

Table 2. Feature list for representing devices.

Category	Feature name	Description
Port	Port	Port number the device is running. SCADA devices generally run port 502, 44818, and so on.
Banner—Structural	Length	Number of characters in banner data. SCADA devices will generally have the same length banner.
	num_token	Number of tokens in banner data. SCADA devices will generally have the same amount of tokens.
Banner—Signature sets	ss_1	Number of unigrams in banner data that belong to set SS_1
	ss_1_port	Number of unigrams in banner data that belong to set $SS_{1,p}$
	ss_2	Number of bigrams in banner data that belong to set SS_2
	ss_2_port	Number of bigrams in banner data that belong to set $SS_{2,p}$

After feature extraction, we trained multiple state-of-the-art classifier types including tree-based (random forest), geometric (logistic regression and SVM), probabilistic (naive Bayes), and neural-network based (artificial neural network and perceptron). Readers interested in each algorithm's technical details should see Ian H. Whitten's book on data mining.[12] Each algorithm was implemented in Weka, a suite of machine-learning software written in Java.

Hold-out validation was used to avoid overfitting. Two-thirds of the devices were randomly selected as the training set, with the remaining third as the test set. Consistent with data-mining principles, each algorithm was evaluated with precision, recall, and f-measure.[12] Each metric's formula is detailed below.

$$\text{Precision}(a) = \frac{\#\text{ of correctly classified instances for class } a}{\text{Total \# of instances identified as class } a}$$

$$\text{Recall}(a) = \frac{\#\text{ of correctly classified instances for class } a}{\text{Total \# of instances in class } a}$$

$$\text{F-measure}(a) = \frac{2 * \text{precision}(a) * \text{recall}(a)}{\text{precision}(a) + \text{recall}(a)}$$

Because f-measure calculates the harmonic mean of precision and recall, the algorithm with the highest f-measure is used. Training, evaluating, and selecting algorithms in this fashion is a commonly accepted practice in data-mining literature.[12] Algorithm 1 summarizes our entire data-mining procedure.

```
Input: tuples [port, banner_data] of all devices.
Output: all SCADA devices.
Procedure:
1. Generate signature sets based on banner_data of all devices.
2. For each tuple T = [port, banner_data]:
Features = port(T) + length(T) + num_tokens(T) + ss_1(T) + ss_1_port(T) + ss_2(T)
+ ss_2_port(T)
2. Separate tuples into training set and test set.
3. Train classifiers based on the training set.
4. Use the classifier C with highest f-measure to classify tuples in the test set
into two classes: is_SCADA = True and is_SCADA = False.
```

Algorithm 1. SCADA identification procedure.

SCADA Vulnerability Assessment

Our second framework component assesses the vulnerabilities of identified SCADA devices. We selected Nessus as our tool for several reasons. First, Nessus provides a rich set of plugins to assess many devices, including SCADA. Other tools lack such diversity. Second, Nessus was designed with scalability in mind, allowing users to scan an unlimited amount of devices simultaneously. Finally, Nessus provides risk rankings and descriptions for each vulnerability, ideal for post-scan analyses.

Consistent with penetration testing principles, we used a subset of Nessus scans.[13] We scanned for vulnerabilities commonly associated with SCADA devices, including SCADA-specific issues (for example, exposed Modbus coils), outdated technology, web application weaknesses, default credential issues, buffer overflow, DoS, and memory leaks.[1,4] Limiting our coverage reduces the risk of unintentionally harming systems and ensures timely assessment completion. This is particularly important for SCADA devices, as disruptions can damage the underlying critical infrastructure.

To complete scans quickly, we configured three slave machines with 200 Gbytes SSD, 32 CPUs, and 256 Gbytes RAM to each scan an equal number of devices. The master (1 Tbyte SSD, 24 Gbytes RAM, and 8 CPUs) monitored each slave. Scans completed in 96 hours.

CLASSIFICATION RESULTS

Random forest achieved the highest f-measure at 99.4 percent (see Table 3). Although this is not significantly higher than others, we selected random forest for application as it has a higher level of interpretability over classifiers.

Table 3. Classification results.

Algorithm	Precision	Recall	F-measure
Random forest	99.4%	99.4%	99.4%
Artificial neural network	99.2%	99.2%	99.2%
Support vector machine	99.0%	99.0%	99.0%
Naive Bayes	98.1%	98.1%	98.1%
Logistic regression	99.3%	99.3%	99.3%
Perceptron	98.9%	99.2%	99.0%

Our classifier identified 587,158 out of the 627 million devices (0.094 percent) as SCADA devices. Given that categorizing devices into finer subsets such as device type or manufacturer would require significant extensions, we save such work for future research and summarize current results. Overall, we identified thousands of BACnet controllers from Honeywell; communications adapters from Lantronix, Omron Corporation, and Veris Industries; and PLCs from Rockwell Automation, Siemens, and Schneider Electric.

VULNERABILITY ASSESSMENT RESULTS

Nessus found that out of 587,158 devices, 37,845 (6.45 percent) have critical, high, medium, and/or low vulnerabilities. Of these, 2,942 (7.77 percent) have critical issues, 4,241 (11.21 percent) have high vulnerabilities, 23,673 (62.55 percent) have medium vulnerabilities, and 6,989 (18.47 percent) have low vulnerabilities. Table 4 summarizes selected vulnerabilities. We discuss below the number of devices and vendors afflicted for each vulnerability.

Table 4. Vulnerabilities and affected vendors at critical, high, and medium risk levels.

Risk level	Number of devices	Vulnerability name	Selected affected vendors
Critical	1,207	Multiple issues: buffer overflow, man in the middle, denial of service (DoS) attacks	Rockwell Automation/ABB
	420	Outdated software or operating system	Rockwell Automation/ABB, Siemens, Schneider Electric, Honeywell
	218	Default credentials or no authentication	Rockwell Automation/ABB, Schneider Electric, Hewlett-Packard, RUGGEDCOM

High	2,211	Multiple PHP vulnerabilities (outdated, buffer overflow, DoS)	Siemens, Schneider Electric, Rockwell Automation/ABB
	1,564	SNMP agent default community name	Rockwell Automation/ABB, Siemens, Schneider Electric, Honeywell
Medium	2,618	Unencrypted telnet servers	Siemens, Honeywell, Power Measurement, Rockwell Automation/ABB, Schneider Electric, Acromag
	2,079	Modbus coil access	Schneider Electric, Lantronix, Power Measurement, Siemens, Modicon

The majority of critical vulnerabilities (1,207 out of 1,843) were due to Allen-Bradley MicroLogix 1400 PLC issues. Allen-Bradley notes that PLCs automate and monitor electromechanical processes including motors, hydraulics, and relays. PLCs are often used in industrial heating and cooling units and factory environments.[14] According to Nessus, attackers can use buffer overflow and man-in-the-middle attacks to harm the device and the infrastructure it controls.

Even more troublesome are the 218 PLCs from Rockwell Automation/ABB, Schneider Electric, Hewlett-Packard, and RUGGEDCOM that accept default credentials. Penetration testing literature states that taking advantage of password issues is often the easiest way to control a system, as attackers do not have to use sophisticated techniques to harm the device.[13] Unlike previous studies that also discovered credential issues,[3,4] we identified seven CODESYS PLCs for building automation, allowing users to start, stop, and reset the PLC with no authentication (see Figure 4). Given that these PLCs are used in buildings, exploiting them can severely interfere with a building's infrastructure operations.

```
------------------------------ snip ------------------------------
?              - show implemented commands
mem            - memorydump
memc           - memorydump relative to code-startaddress
memd           - memorydump relative to data-startaddress
reflect        - reflect current command (test!)
dpt            - get data-pointer-table
ppt            - get POU-table
pid            - get project ID
pinf           - get project info
tsk            - get IEC-task-list and IEC-task-infos
startprg       - start PLC program
stopprg        - stop PLC program
resetprg       - reset PLC program
resetprgcold   - reset PLC program cold
resetprgorg    - reset PLC program original
reload         - reload boot project
getprgprop     - program properties
getprgstat     - program status
filecopy       - copy files [from] [to]
filerename     - rename files [old] [new]
filedelete     - delete file [filename]
filedir        - directory list [start directory]
saveretain     - save retains in file [filename]
restoreretain  - restore retains from file [filename]
setpwd         - set online access password
delpwd         - delete online access password
io             - list info of plugged terminals
fds            - amount of free diskspace
format         - format file-system
extract        - extract files

------------------------------ snip ------------------------------
```

Figure 4. Programmable logic controller (PLC) configuration options for building automation. Attackers can start, stop, and reset PLCs, potentially causing severe damage to the building.

CONCLUSION AND FUTURE DIRECTIONS

This study aimed to identify all SCADA devices found in the Shodan database and assess their vulnerabilities using text mining, data mining, and vulnerability assessments. We found that more than 500,000 devices from major SCADA vendors can be accessed through Shodan and that thousands of these devices suffer from critical vulnerabilities such as default credential issues or outdated software.

There are several promising future directions. First, additional work can identify specific device owners and locations. Many IPs point to ISPs rather than to the owner—running traceroutes can help pinpoint owners. Second, future work can categorize devices at a finer grain to understand vulnerabilities afflicting specific subsets. Finally, researchers can employ assessments at regular time intervals to identify how SCADA vulnerabilities evolve. Each direction will help build robust cyber defenses for society's most critical assets.

ACKNOWLEDGMENTS

This material is based upon work supported in part by the National Science Foundation (DUE-1303362 and SES-1314631). We also thank Eric Gross and Samantha Forbis for their assistance.

REFERENCES

1. A. Nicholson et al., "SCADA Security in the Light of Cyber-Warfare," *Computers & Security*, vol. 31, no. 4, 2012, pp. 418–436.
2. P. Paganini, "The Number of ICS Attacks Continues to Increase Worldwide," *Security Affairs*, blog, 28 December 2016; http://securityaffairs.co/wordpress/54792/security/ics-attacks-2016.html.
3. M. Patton et al., "Uninvited Connections: A Study of Vulnerable Devices on the Internet of Things (IoT)," *2014 IEEE Joint Intelligence and Security Informatics Conf.* (JISIC), 2014, pp. 232–235.
4. S. Samtani et al., "Identifying SCADA Vulnerabilities Using Passive and Active Vulnerability Assessment Techniques," *2016 IEEE Conf. Intelligence and Security Informatics* (ISI), 2016, pp. 25–30.
5. J. Matherly, *Complete Guide to Shodan: Collect. Analyze. Visualize. Make Internet Intelligence Work for You.*, Leanpub, 2016; https://leanpub.com/shodan.
6. A.V. Deokar and J. Tao, "Semantics-Based Event Log Aggregation for Process Mining and Analytics," *Information Systems Frontiers*, vol. 17, no. 6, 2015, pp. 1209–1226.
7. M. Balduccini, S. Kushner, and J. Speck, "Ontology-Driven Data Semantics Discovery for Cyber-Security," *Int'l Symp. Practical Aspects of Declarative Languages* (PADL), 2015; doi.org/10.1007/978-3-319-19686-2_1.
8. B. Joshi, U. Bista, and M. Ghimire, "Intelligent Clustering Scheme for Log Data Streams," *Int'l Conf. Intelligent Text Processing and Computational Linguistics* (CICLing), 2014, pp. 454–465.
9. I. Weber et al., "Discovering and Visualizing Operations Processes with POD-Discovery and POD-Viz," *45th Ann. IEEE/IFIP Int'l Conf. Dependable Systems and Networks* (DSN), 2015, pp. 537–544.
10. H. Saneifar et al., "From Terminology Extraction to Terminology Validation: An Approach Adapted to Log Files," *J. Universal Computer Science*, vol. 21, no. 4, 2015, pp. 604–636.
11. J. Jayadeep, "Silhouette Threshold Based Text Clustering for Log Analysis," *Int'l J. Data Mining Techniques and Applications*, vol. 6, no. 1, 2017, pp. 17–25.
12. I.H. Witten et al., *Data Mining: Practical Machine Learning Tools and Techniques, 4th ed.*, Morgan Kaufman, 2016.

13. G. Weidman, *Penetration Testing: A Hands-On Introduction to Hacking, 1st ed.*, No Starch Press, 2014.
14. P. Stephenson, *Tenable Network Security Nessus*, blog, SC Media, 2 March 2015; www.scmagazine.com/tenable-network-security-nessus/review/6977.
15. *User Manual: MicroLogix 1400 Programmable Controllers*, Rockwell Automation, 2015; http://literature.rockwellautomation.com/idc/groups/literature/documents/um/1766-um001_-en-p.pdf.

ABOUT THE AUTHORS

Sagar Samtani is a doctoral student in the Department of Management Information Systems (MIS) and a research associate in the Artificial Intelligence (AI) Lab at the University of Arizona (UA). His research interests include cyberthreat intelligence, vulnerability assessment, machine learning, and text mining. Contact him at sagars@email.arizona.edu.

Shuo Yu is a doctoral student in the Department of MIS and a research associate in the AI Lab at UA. His research interests include mobile health, machine learning, and text mining. Contact him at shuoyu@email.arizona.edu.

Hongyi Zhu is a doctoral student in the Department of MIS and a research associate in the AI Lab at UA. His research interests include mobile health, machine learning, and visualization. Contact him at zhuhy@email.arizona.edu.

Mark Patton is a lecturer in the Department of MIS at UA and the program administrator for the NSF-funded AZSecure Cybersecurity Fellowship Program. His research interests include security informatics and vulnerability assessment. Patton received a PhD in MIS from UA. Contact him at mpatton@email.arizona.edu.

John Matherly is the founder of Shodan, the world's first search engine for Internet-connected devices. His work has been featured on CNBC, Bloomberg, and others. Contact him at jmath@shodan.io.

Hsinchun Chen is a Regents' Professor and the Thomas R. Brown Chair in Management and Technology in the Department of MIS, as well as the founder and director of the AI Lab, at UA. His research interests include security informatics and data/text/web mining. Chen received a PhD in information systems from New York University. He is a Fellow of the IEEE, ACM, and AAAS. Contact him at hchen@eller.arizona.edu.