Integration of ZMap with Shodan for Comprehensive Internet of Things Research

By

Samantha L. Forbis

A Thesis Submitted to the Faculty of the

Department of Management Information Systems

In Partial Fulfillment of the Requirements

For the Degree of

MASTER OF SCIENCE

In the Graduate College

THE UNIVERSITY OF ARIZONA

2015

STATEMENT BY AUTHOR

This thesis has been submitted in partial fulfillment of requirements for an advanced degree at the University of Arizona and is deposited in the University Library to be made available to borrowers under rules of the library.

Brief quotations from this thesis are allowable without special permission, provided that an accurate acknowledgement of the source is made. Requests for permission for 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 judgement the proposed use of the material is in the interests of scholarship. In all other instances, however, permission must be obtained from the author.

The view expressed in this thesis are those of the author and do not reflect the official policy or position of the University of Arizona, or the National Science Foundation. This material is based upon work supported by the National Science Foundation under Grants No. DUE-1303362 and No. SES-131463

SIGNED: Samantha L Forbis

APPROVAL BY THESIS DIRECTOR

This thesis has been approved on the date shown below:

Dr. Hsinchun Chen UA Regents' Professor of MIS	Date

ACKNOWLEDGEMENTS

I would like to extend my thanks and appreciation to Dr. Hsinchun Chen for his leadership and guidance and over the past 2 years.

I would also like to thank my cohort colleagues, Ryan Chinn, Eric Gross, and Leon Walker, for their continued support and motivation.

Finally, I would like to thank Dr. Mark Patton for taking the time to mentor me and help me become a better researcher.

DEDICATION

I dedicate this work to my husband, George, and my son, Damion, for their endless motivation and patience over the course of this research. Without their support, none of this would have been possible.

ABSTRACT

The perpetuation of devices that populate the Internet of Things (IoT) continues to increase at a furious pace. The state of the security of these devices has not followed suit. This situation is continuously overlooked by manufacturers, to whom the bottom line is most important, and by consumers, to whom convenience and device features are most important. The dual neglect has led to an increasingly dubious state of insecurity amongst all types of Internet-facing devices. From consumer devices to industrial control devices, security and convenience continue to clash.

Tools have emerged to locate these highly visible Internet-facing devices and highlight the depth to which the security problem goes. Academic research aims to identify these vulnerable devices to aid in the mitigation and remediation of this issue.

Through this paper, two popular tools are reviewed: Shodan and ZMap. The review includes a highlight of past papers and projects employing these tools, identification of the advantages and disadvantages of each, a verification of these aspects, and a discussion of the benefit the combination of these tools have, when used together, for enriching a comprehensive Internet of Things academic research environment.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

With the evolution of technology, the size of components within electronic devices is decreasing rapidly, along with the size of the electronic devices themselves. Additionally, growth and improvement of the nation's broadband, wireless, and Wi-Fi capabilities has allowed for more reliable wireless integration with electronic devices, commonly referred to as digitization (Press 2014). These technological innovations have also enabled the creation of devices that were historically thought to be impossible and adding previously unimagined convenience and efficiency to our lives. One just need witness the amazing abilities of the state-of-the-art home automation system to know that current and new technology are potentially ushering in a second industrial revolution, or more appropriately *technological revolution*.

The Internet of Things (IoT) has been difficult to define. Internet connected devices are found in every facet of life including: consumer products, healthcare, industry, transportation, retail, smart infrastructure, security and surveillance, and many other areas. In fact, it is estimated that there will be approximately **50 billion** devices connected to the Internet by 2020 which equates to about 8 devices per person. It is estimated that there are currently 15 billion devices connected to the Internet today (Evans 2011). One definition that was appealing is from a Morgan Stanley blue paper on the IoT: "An army of tens of billions of tiny robots making our lives easier" (Meunier, Wood et al. 2014). But the most concise and still comprehensive definition found, also from Morgan Stanley, is: "The next generation of personal computing, whereby objects interact, potentially independently, with each other and their environment" (Meunier, Wood et al. 2014) .

The IoT has some amazing devices that have allowed some amazing collaboration and improved efficiency. However the security of these devices, from the production standpoint, has always been

on the back burner. The public demand for these devices is very high. This leads manufacturers to get the device produced and into the hands of the public as quickly as possible, often sacrificing the security aspect in the process. Consumers assume the manufacturer has their best interest in mind, leading them to overly trust the manufacturers and not give a passing thought to potential implications of their usage of a grossly insecure device. This is more than evident in public's willingness to handle financial transactions through the devices as well as allow for individual location tracking, for example the convenience of knowing where your jog took you and how far you went (MapMyFitness 2014). The security of these devices has become big news in the past couple years. An article published on April 27, 2015 states: "less than half [of organizations] focus on securing their IoT products at the beginning of the product development phase, and 47 per cent do not provide any privacy-related information on their IoT products" (Wilcox 2015).

At this point in the game, society would almost definitely be unwilling to give up all of these conveniences. So the next logical step is to determine what we can do to protect these devices and, more importantly, our data and infrastructure. How do we determine what devices are vulnerable, how they can be seen, and who can see them?

One such tool for this type of exploration and discovery is Shodan, known as "The search engine for the Internet of Things." (Matherly 2015). Shodan was developed by John Matherly in 2009 as a way for companies to see who was using their devices and how. Since then, Shodan has grown to become a massive index of stored banner and port data. A banner is the metadata (data that describes data) a device returns to identify itself to the querying machine. A port is "a number that identifies one side of a connection between two computers…to determine to which process or application a message should be delivered" (Unknown 2013b). Shodan employs in-house developed, proprietary, port-scanning algorithms to acquire this data and currently runs scans on

approximately 155 ports (Appendix E). The site runs on any system and also provides a convenient graphical user interface (GUI), which presents all of the data in an easy to understand format.

The amount of internet-facing devices that are found on Shodan is staggering. A person can find everything from home automation devices, security cameras, and car washes to traffic lights and hydroelectric plant control systems (Goldman 2013). The implications of the insecurity of these Internet-facing devices warrants increased attention. For example, someone with malicious intent could gain access to an ICS (Industrial Control System) such as a hydroelectric plant, and use that access to change vital settings where "the rotor speed could exceed supported [levels], which could cause an explosion in the generator, damaging the pipes and cause a large dam leakage." (H.S.Peleaz 2013). This type of damage to a vital electricity generating facility could lead to widespread blackouts that could last a very long time. Even worse, if a nuclear power plant were to experience the same events, the results could be catastrophic (H.S.Peleaz 2013).

However, when used for good, the data and devices found on Shodan can be used to ensure systems we believe to be secure and not openly accessible via the Internet, are *really* secure and not openly accessible via the Internet. Thinking something is secure and knowing it is secure are two very different things. Therefore, Shodan should be viewed as an opportunity to find insecure devices, determine who can see them and access them, then take the necessary steps to remediate any findings that are contrary to the organizations security standards.

Another tool that runs similar scans is ZMap. Developed at the University of Michigan and released in 2013, ZMap is an open-source tool that allows a person to scan the entire IPv4 range on a particular port in 45 minutes (Durumeric, Wustrow et al. 2013) ZMap's goal has been to "elevate Internet-wide scanning from an expensive and time-consuming endeavor to a routine

methodology for future security research" (Durumeric, Wustrow et al. 2013). In contrast to Shodan, ZMap is a Linux-based, command line tool without a GUI.

While both Shodan and ZMap are excellent tools, the differences and similarities must be noted. These similarities and differences are what make combining these two tools ideal.

	Ports	Interface	OS	Code Type	Per Connection State
ZMap	65535	Command Line	Linux	Open-Source Modular	No
Shodan	155	API, GUI	Any	Proprietary	Yes

Table 1 - ZMap and Shodan Comparison

It is because of the proliferation of the devices that comprise the Internet of Things, and the acutely insecure state of these devices, that this research was undertaken. The primary motivation for this research was to determine how to cohesively combine Shodan and ZMap into a vulnerability assessment framework. With this framework, the goal is to assist in the development of an automated device data collection, identification, and verification system to further enhance cybersecurity research and aid in the efforts to mitigate the security nightmare we are currently facing.

2 LITERATURE REVIEW AND PREVIOUS WORK

2.1 Literature Review

To establish domain knowledge and determine prior research, two primary areas of literature are reviewed: Shodan and ZMap. It was necessary to determine how the tools are being used and for what purpose people are using them. Through this review a comprehensive assessment of the literature, as well as a critique to highlight areas that indicate research gaps, will be provided. These steps will identify why this research is relevant and provide a means for comparison.

The literature has been broken down into 6 relevant attributes for more comprehensive comparison to allow for quick identification of the strengths and weaknesses of each piece of literature as well as what gaps may exist.

The review will begin with Shodan and proceed to ZMap. Next is a highlight of some of the projects that have been undertaken using one or both of these tools. Finally, a brief overview and discussion pertaining to the ethics and legality of these methods, is presented.

2.1.1 Shodan

2.1.1.1 Literature

Paper	Focus	Methods	Data Source(s)	Results
Bodenheim (2014)	Shodan indexing function	Deployed unsolicited, internet-facing ICS honeypots	Shodan DB Nmap TCPdump Wireshark	Indexing completed between 3 and 13 days
Bodenheim, et al. (2014)	Shodan functionality	Deployed 4 Allen-Bradley PLCs	Shodan DB Shodanhq	Devices identifiable within 19 days
Patton, et al. (2014)	Vulnerability discovery	Manual processing	Shodan DB Password DB	Vulnerability rate:.44% to 40%
Radvanovsky (2014)	SCADA/ICS, Project SHINE	Use Shodan to extract and store SCADA/ICS data into database	ShodanDB	More than 2.2 million devices
Williams (2014)	Distinguish Internet-facing ICS devices indexed by Shodan	Collect and compare PLC code	ShodanDB SMEs	540 Target IPs 3608 devices on port 44818

Figure 1 - Shodan Literature Taxonomy

DB	Database
ICS	Industrial Control System
IP	Internet Protocol
PLC	Programmable Logic Controller
SCADA	Supervisory Control And Data Acquisition
SME	Subject Matter Expert

Table 2 - Shodan Literature Acronyms

2.1.1.2 Key Findings

After reviewing the above literature, as well as many other articles and papers not listed, there are a number of attributes that stand out. First, the primary motivation in a majority of papers is in determining how Shodan performs it indexing and how that indexing has an effect on their network and devices. Second, the methods have been primarily manual in nature.

In Williams (2014), separate panels of PLC code experts and ICS Engineers each individually reviewed PLC code collected to determine its legitimacy as an ICS device and produce a list of keywords to back up these determinations. While producing the keywords can lead to an automated process, there is no indication from the author that any sort of automated procedure was part of future endeavors.

In Radvanovsky (2014), *Project SHINE* (SHodan INtelligence Extraction) was a limited run project that used only the Shodan database "with the intention of defining a searchable term criteria set metadata database that could be modified easily to establish a census baseline of all SCADA/control systems discovered through the SHODAN" (Radvanovsky 2014). Due to the sheer magnitude of devices they found on Shodan, as well as the new ones added daily, an actual baseline could not be established. However, this report does illustrate their findings as of the end of the project on January 31, 2014. These results include top manufacturers in several areas, as well as top countries identified by a predetermined list of 5 ports, and the top countries identified through Shodan.

A few shortcomings stand out. The actual methodology, search terms, and results are not published and are not available. This makes duplicating the process almost impossible – reducing the effectiveness of the scientific method. Also, they did not perform any sort of de-duplication,

potentially skewing the results significantly. Lastly, since this was a limited-run project, the actual results loose validity as time passes and the state of the devices changes.

The Bodenheim (2014) and Bodenheim, et al. (2014) papers both employed use of honeypots to "determine Shodan's scanning routine, scanning frequency, and web database identification timelines."(Bodenheim 2014). This approach helps to control the environment and provide an accurate representation of what they were looking for. However, a controlled environment doesn't always translate directly to real-world situations.

Lastly in Patton, et al. (2014) a good portion of the results and responses involved manual verification and processing. Even so, the paper illustrated a clear quantification of vulnerabilities located on Shodan. However, like Williams (2014), no indication was made of automating these processes in the future.

While Shodan has proven to be proficient in finding and indexing devices on the IoT, at the time of this writing the number of ports scanned is limited to 155 ports, whereas ZMap is not constrained by this limitation. To explore ZMap's functionality, and the projects that have employed its use, the following literature was reviewed.

2.1.2 ZMap

2.1.2.1 Literature

Paper	Focus	Methods	Data Source	Results
Adrian, et al. (2014)	ZMap optimization	Introduce optimized address constraints	ZMap scan results	Complete scan of the public IPv4 space in 4m29s
Durumeric, et al. (2013)	Internet-wide network scanning	Optimized probing	Scan Results	Complete scan of public IPv4 space in 44 minutes
Durumeric, et al. (2014)	Identify broad scanning pattern behavior	Analyze scan traffic from the past year	dark net by Merit Network	10.8M scans from 1.76M hosts
Lawshae (2014)	Host reporting of botnet infection	Write a payload Scan with ZMap	Results from Scans	10500 Unique hosts across 114 countries
Pujol, et al. (2014)	Server-to-server Web traffic over the public Internet	Identify back-office traffic	Exchanges ISP Traces CDN Traces ZMap Datasets	11% of web servers also act as web clients
Schloesser (2013) Rapid7-Sonar (2015)	Internet-wide scanning	Scan IPv4 space with ZMap	Results from scans	Scans.io Critical.io

Figure 2 - ZMap Literature Taxonomy

CDN	Content Delivery Network
ISP	Internet Service Provider
IPv4	Internet Protocol version 4
XML	Extensible Markup Language

Table 3 - ZMap Literature Acronyms

2.1.2.2 Key Findings

Upon completion of reviewing the above literature, a few items have become evident. First, the majority of the studies and papers about ZMap are written by the creators themselves. These include the papers that are also available on the ZMap website. While these papers are informative and well written, papers by a wider variety of researchers would provide a more comprehensive, objective analysis of the functionality of ZMap as a whole. Research using ZMap as an active scanning tool is very limited. The primary tool for Internet-wide scanning appears to remain dominated by Nmap.

Durumeric, et al. (2013) is the paper that introduces ZMap, the Internet-wide scanning tool. As one of the first fast Internet-wide scanning tools, ZMap was a more time efficient solution for enabling security researchers the ability to expand and improve security research.

Adrian, et al. (2014) reveals the updated, faster version of ZMap that introduces various optimizations to reduce the overall processing cost and reduce bottlenecks caused by the blacklist and whitelist features. These are necessary to accommodate those who ask to be removed from future scans and avoid scanning IPs that are private. The improvements also reduced the time to scan the entire public IPv4 address space from approximately 44 minutes down to approximately 4 minutes. This is a drastic improvement. However, to achieve these scan rates the network the scan is being performed from needs to operate at near maximum, optimal bandwidth of 10Gbps. (Gigabits per second). Unfortunately, these speeds are difficult to achieve and maintain in most academic settings.

Durumeric, et al. (2014) is an actual project undertaken by the ZMap team at the University of Michigan. This project aimed to document and analyze the changes in scanning patterns since

prior, older studies. The team uncovered a multitude of interesting statistics including: top targeted ports, top scanning countries, top scans by software, differences in targeted ports from 2004, 2010 and 2014, commonly targeted services, and many more. They also analyzed several case studies including: the Heartbleed Vulnerability, the NTP DDoS Attacks, and the Linksys Backdoor. The case studies looked into how attackers were using tools like ZMap and Masscan and how quickly they began scanning after announcement of the vulnerability. These studies provided very interesting results, but lacked the real-time component. But this is not something identified as a goal in the project.

Pujol, et al. (2014) takes an interesting approach in that their goal was to identify and measure server-to-server web traffic known as back-office traffic. The server-to-server traffic is not visible to front-office, or Web server/client, architecture. The authors state that this is the first paper to take on this subject as it is usually overlooked for studies focused on Web server/Web browser traffic, which is more directly related to the end-user experience. This project uses ZMap, but not for active scanning. Instead they use ZMap data sets to classify IPs and to put other data into perspective.

Lawshae (2014) is the most creative use for ZMap in this list. Lawshae chose to take an active approach at using ZMap to identify malware infected botnet hosts. The Zero Access malware employs a feature where hosts can communicate with one another to identify and maintain a list of infected hosts. This allowed Lawshae to use ZMap to deploy a mimicked version of the command used for hosts-to-host communication to receive feedback and confirmation of infected hosts. Lawshae identified 10,500 unique hosts in 114 countries as well as the top 10 most affected ISPs. The unconventional use of ZMap to deliver a payload was creative and opens the door for other researchers to explore similar functionality.

The Schloesser (2013) and Rapid7-Sonar (2014) papers introduce Project Sonar. This project is an open-source, community based, collaborative port-scanning effort created by Rapid7. Project Sonar performs scans using ZMap, DAP for data processing, and Recog for fingerprinting the processed data. DAP and Recog are Rapid7 developed tools. The scans performed by the community are collected by Rapid7 and uploaded to scans.io for public access. The fact that this project is open-source and the tools are easy to obtain makes this a progressive, positive approach for enhancing security research. Like ZMap, DAP and Recog are command-line, Linux based programs. One disappointment across both ZMap and Project Sonar is the lack of adequate documentation. For users not completely familiar with Linux environments, this can present a barrier to effective use. ZMap documentation provides a high level, competent overview of basic functionality, however when looking into the more useful, advanced features, the explanations become vague. Finding information on Project Sonar, aside from the main web page, is difficult. The main page gives a broad overview of the project and provides links to GitHub pages for the project as well as DAP and Recog.

2.1.3 Port-scanning Ethics and Legality

A discussion of Shodan and ZMap cannot be had without attention paid to the controversial nature of port-scanning in general. Kenneally (2015) discusses the existing disconnect between indicators of legal and ethical risk, and the current advancement of technology - including the increasing availability of data online. This disconnect makes determining where the grey areas are, or what the responsibility of the researcher is, in terms of data that may be of personal, sensitive, or illegal nature but available from a *public* online source.

The makers of ZMap have identified what they are calling “Scanning Best Practices” (Wustrow, Durumeric et al. 2014). These best practices are as follows:

1. Coordinate closely with local network administrators to reduce risks and handle inquiries
2. Verify that scans will not overwhelm the local network or upstream provider
3. Signal the benign nature of the scans in web pages and DNS entries of the source addresses
4. Clearly explain the purpose and scope of the scans in all communities
5. Provide a simple means of opting out and honor requests promptly
6. Conduct scans no larger or more frequent than is necessary for research objectives
7. Spread scan traffic over time or source addresses when feasible.

They go on to state that researchers need to be aware of any local legalities. Even more specific, they inform researchers to not exploit found or known vulnerabilities as well as not to access obviously protected systems.

2.1.3.1 Internet Census 2012

The Internet Census 2012 is an attempt at Internet-wide scanning to return the ports, software, and devices used throughout the IPv4 address space. It was with good intentions that the researchers approached this limited run project which employed a botnet, called the Carna Botnet. This botnet was programmed in the “least invasive way possible and with the maximum respect to the privacy of the regular device users” (Unknown 2013a). The project produced a vast, valuable amount of information. However, the use of this botnet is, in fact, illegal. With the dubious state between what is legal and what is not legal in computer and online research, it is highly recommended to avoid this approach. Determining the best way to undertake online research will continue to be an area of contention and confusion.

3 RESEARCH GAPS AND QUESTIONS

3.1 Research Gaps and Discussion

3.1.1 Shodan Research Gaps

Research as it pertains to Shodan, has primarily centered on determining how Shodan does its scanning and indexing (Bodenheim 2014, Bodenheim, Butts et al. 2014). Other research involves regularly accessing Shodan databases and saving the results in their own databases (Radvanovsky 2014). Notably, the primary subject matter for the majority of Shodan research revolves around SCADA/ICS devices (Bodenheim, Butts et al 2014, Radvanovsky 2014, Williams 2014).

3.1.2 ZMap Research Gaps

Research as it pertains to ZMap, has primarily used ZMap as a *supplemental tool* rather than an active scanning component. In Durumeric, et al. (2014) and Pujol, et al. (2014), ZMap wasn't used for scanning at all; rather prior scans performed and stored at scans.io were used.

3.2 Research Questions

Based on the research gaps identified through the literature review, and identified in the previous section, the following research questions are proposed.

3.2.1 Research Question 1 (RQ1)

What are the similarities and differences in the functionality and usability of Shodan and ZMap?

3.2.2 Research Question 2 (RQ2)

Can ZMap provide a confirmation mechanism for results obtained through queries performed on Shodan? Also, how can ZMap potentially expand on those results?

4 RESEARCH DESIGN, TESTBED AND DISCUSSION

4.1 Research Design Intro

To answer these questions, research as broken up into manageable steps. These steps are illustrated in the research design diagram presented below.

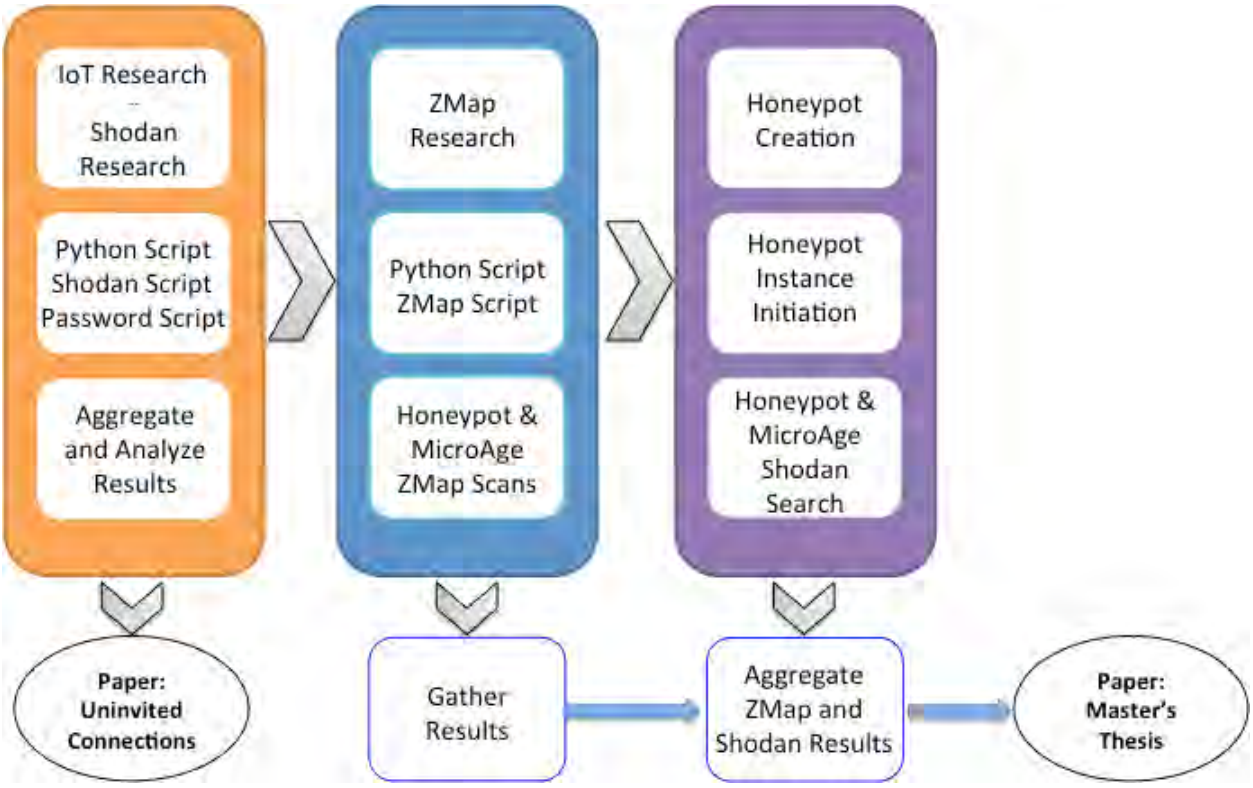


Figure 3 - Research Design Diagram

4.1.1 Shodan Exploration

Shodan Exploration began with research into the Internet of Things (IoT). This was necessary to establish domain knowledge for performing competent keyword searches through Shodan. While general IoT research was useful, to gather keywords it was more productive to select devices (or types of devices) of interest and search the manufacturer's website, or user manuals, for device specific data. Following IoT research, exploration of the interface, commands, API usage, and JSON output of Shodan was undertaken.

Similar to a web page search engine such as Google, to search for IoT devices on Shodan, keywords are vital. A search using broad keywords such as "webcam" will return many devices. However, some webcams do not have the word "webcam" in their banner and those may not be part of the search results. If a specific device is of interest, more customized keywords such as "AXIS 212" will yield more relevant results.

Shodan is also able to perform searches using IP addresses and CIDR notation. These are extremely narrow keywords and a very limited number of results should be expected. Search results on the webpage are returned in an easy to read format. This example is of a port 80, HTTP banner:

4.1.2 ZMap Exploration

ZMap exploration began with research into port-scanning in general. Understanding the different port-scanning methods is important to achieve a competent level of comprehension into the functionality of ZMap. There are many different types of port-scanning methods. By default, ZMap employs three (3) methods: TCP SYN Scan, ICMP Echo Scan, and UDP Datagram Scan.

The TCP SYN scan is known as a *half-open scan* where the source machine initiates the three way handshake process with the destination machine by sending a SYN, or synchronize, command. The destination machine will respond in one of three ways: No Response – indicating a probable closed or filtered port, RST (rest) – indicating a closed port, or SYN ACK (synchronize acknowledge) – indicating the port is open and the destination machine is willing to communicate. If there is no response or a RST is received, the source machine moves on. If SYN ACK is received, the source machine will send RST to discontinue the communication and avoid causing a denial of service situation. With this type of scan, no full connection is made and no data is exchanged.

UDP Datagram Scan sends a UDP packet to a port and if that port returns an ICMP unreachable type of response, that port is assumed to be closed. This can be problematic and lead to false negative and false positives. If a port is blocked in any way, by a firewall for example, it won't return a response leading the scan to identify it as open. Therefore it is never really certain if a port is actually open or not. However, this is a good way to get an idea of where to begin and move to another type of scan after narrowing down the candidate field.

The ICMP Echo Scan is a bit different that the TCP SYN and the UDP Datagram scans in that ICMP doesn't have port numbers. A request, ping for example, is sent to a destination machine and the machine will reply with an ICMP packet if received.

ZMap is equipped with the ability to compose custom probe and output modules. Output from the scans can be formatted in 3 forms: CSV, JSON, and Redis. CSV is default with functionality for JSON and Redis involving additional installation and implementation procedures.

With approximately 31 command line options (Appendix A) and 18 output field options (Appendix B), it is extremely beneficial that ZMap allows for the use of config (configuration) files. A custom config file can be created or the user can modify the default 'zmap.config' file. Equally as indispensable are ZMap's blacklist and whitelist features. The blacklist is editable and comes preconfigured with IANA IPv4 Special-Purpose Address Registry (Appendix C) restricted IPs.

For the purposes of this research, ZMap was installed on an Ubuntu 14.04 VirtualBox virtual machine. Since ZMap is a command-line, Linux only tool, the virtual machine was necessary. Scripts were created to automate and expedite the scanning procedures.

4.1.3 Honeypot Deployment

Three (3) Amazon Elastic Compute Cloud (EC2) instances were created with each in different EC2 geographic regions in the United States: California, Oregon, and Virginia. The honeypots were created using Dionaea and p0f and, upon creation, were instantiated and left running.

4.2 Research Testbed

To provide proof on concept. ZMap and Shodan were tested on two sets of IPs. The first set is live, actual IPs, with the second being the honeypots. It was necessary to explore both of these options because, by nature, honeypots will tend to have more open ports than a live IP. (List of IPs and ports in Appendix D)

4.2.1 MicroAge Lab Testbed

To scan the University of Arizona MicroAge lab with ZMap, public IPs were scanned using their CIDR notation on 2510 ports. Out of these, 10 ports returned open on two individual IPs. Shodan searches were performed manually on the IPs that ZMap returned as open. A test set of those that ZMap did not indicate as open were also scanned. The results are as follows:

IP Addresses	Ports									
	21	22	42	80	427	443	902	1433	3306	8000
H_VA				Shodan						
H_VA	ZMap			ZMap		ZMap				
H_OR		Shodan		Shodan		Shodan				
H_OR		ZMap		ZMap	ZMap	ZMap	ZMap			ZMap

Table 4 - MicroAge Lab Scan Results

With ZMap, not every IP in the CIDR range registered as open, which was expected. Out of 312 IP addresses, only 2 registered as open. Additionally, on Shodan the CIDR notation returned nothing. However, Shodan was able to find them using the individual IPs that had registered as open with ZMap.

4.2.2 Honeypot Testbed

ZMap scans were performed on 3 individual honeypots. The honeypot IP addresses are not in CIDR notation. This simplified the Shodan searches. As these were the only IP addresses for the individual honeypots, no additional IPs were scanned. The results are as follows:

IP Addresses	Ports									
	21	22	42	80	427	443	902	1433	3306	8000
H_VA										
H_VA										
H_OR										
H_OR										
H_CA										
H_CA										

ZMap

Shodan

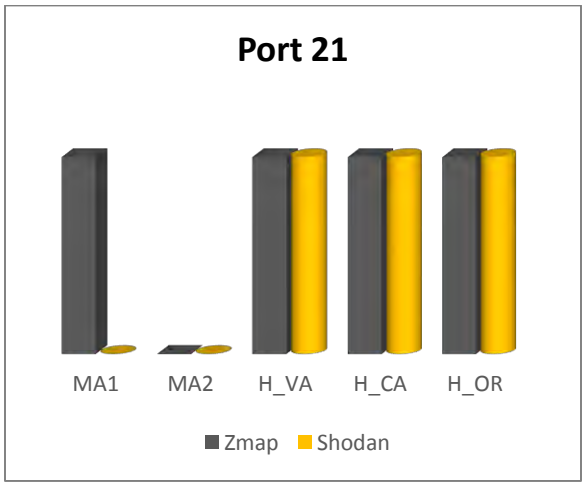
Table 5 - Honeypot Scan Results

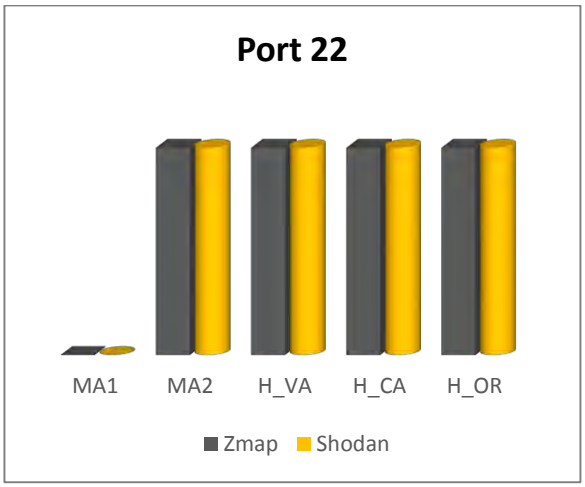
Every IP was able to be scanned and returned some result on either ZMap or Shodan, or both.

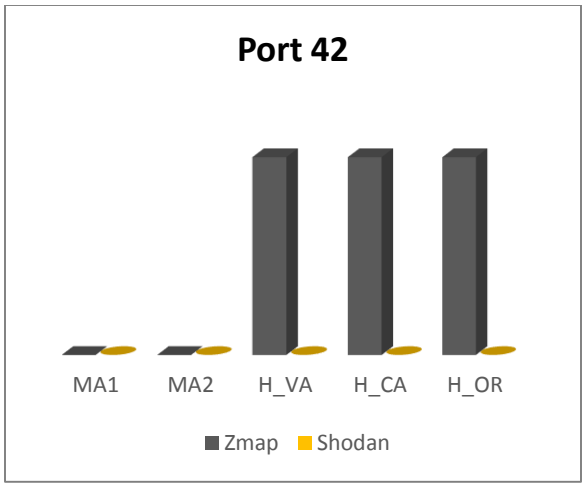
4.3 Results

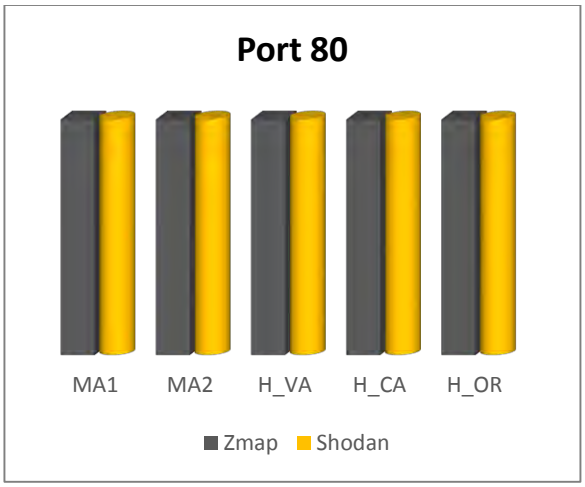
4.3.1 Detailed Results Figures

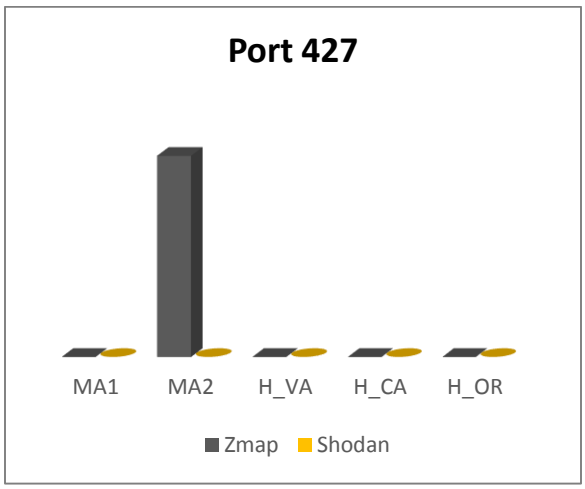
Below are figures that show the similarities and differences between the Shodan and ZMap scans, by port. These results are binary (identified or not identified), with a column indicating *identified* and a flat circle and/or square indicating *not identified*.

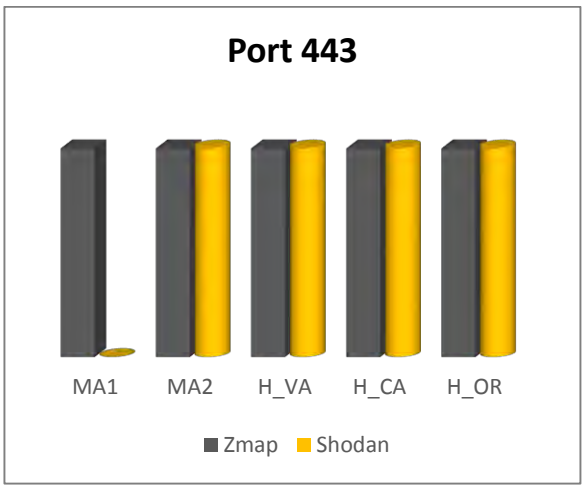


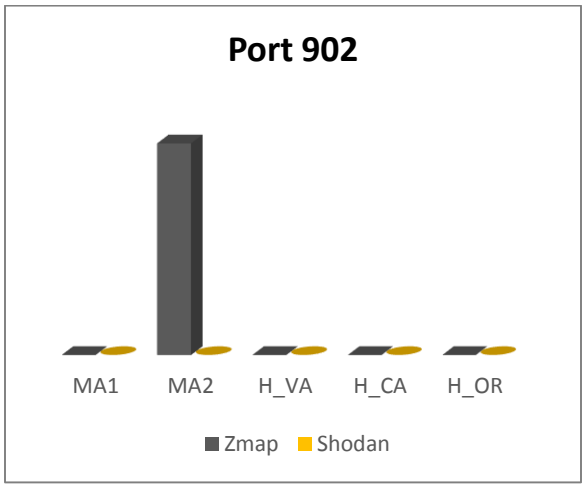


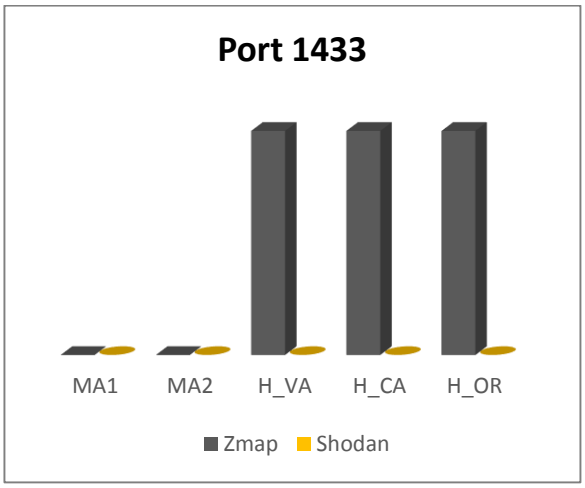

Port 42
MA1
MA2
H_VA
H_CA
H_OR
Zmap
Shodan

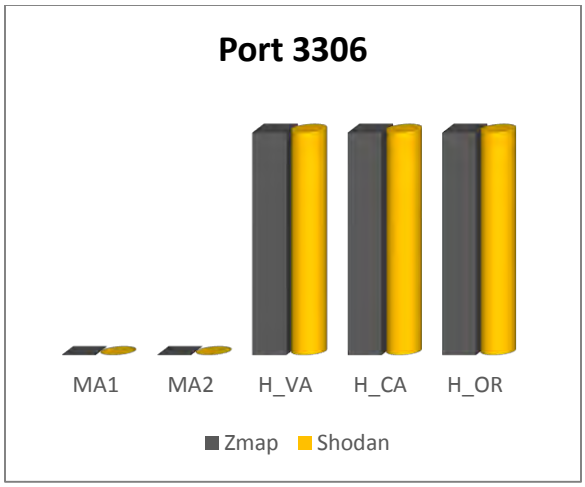

Port 80
MA1
MA2
H_VA
H_CA
H_OR
Zmap
Shodan

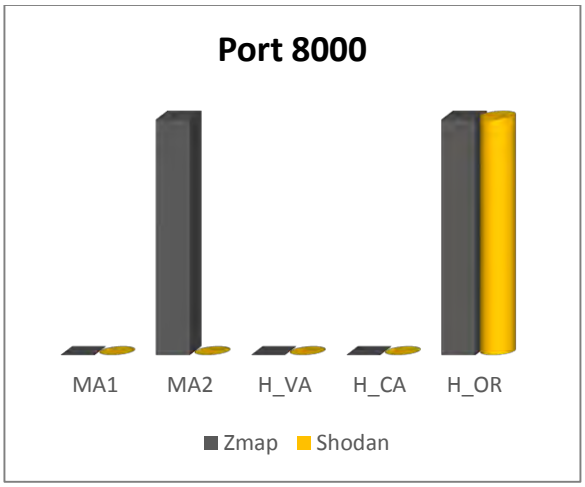

Port 427
MA1
MA2
H_VA
H_CA
H_OR
Zmap
Shodan


Port 443
MA1
MA2
H_VA
H_CA
H_OR
Zmap
Shodan


Port 902
MA1
MA2
H_VA
H_CA
H_OR
Zmap
Shodan


Port 1433
MA1
MA2
H_VA
H_CA
H_OR
Zmap
Shodan





4.4 Discussion of Using Testbed

4.4.1 Discussion of similarities between Shodan results and ZMap results

Shodan and ZMap have similar functionality. Port-scanning is central to Shodan's success and is the primary functionality of ZMap. Several ports are used much more often than others. Port 80 is a highly popular port since it handles the HTTP protocol for web browsing. Shodan was able to find the majority of popular ports such as port 80, 22, and 443. ZMap also displayed similar success with these ports

4.4.2 Discussion of differences in Shodan and ZMap results

One of the primary limitations of Shodan is the limited range of ports that are scanned. With ZMap any port is capable of being scanned. Private IPs, such as on a personal network, can be scanned with editing or disabling of the blacklist function.

With Shodan, search results of a particular IP displays all of the ports that are indicated as open for that IP. ZMap is limited as far as this functionality is concerned. Scanning a range of ports is not currently supported so scanning one IP will return if that one port is open or not. In order to scan multiple ports without typing out the command line text each time, it is necessary to have those ports saved in a TXT or CSV file. To access this file, custom Python scripts need to be created. Finally, the banner grab performed, and returned, by Shodan is not part of the default scanning configuration for ZMap. In order to accomplish this task, a separate add on function is necessary. This function is set up completely separately from ZMap, but needs ZMap to work. In regards to the actual scans, ZMap was able to retrieve results from more ports than Shodan for the same IP addresses. In fact, there was never a time that Shodan reported a port that ZMap didn't find.

The chart below shows the ports and IPs scanned, the total positive results from each tool, the totals of those results and the percentage of ports that Shodan found when compared with ZMap.

Port	Shodan	ZMap	Percent
21	3	4	75%
22	4	4	100%
42	0	3	0%
80	5	5	100%
427	0	1	0%
443	4	5	80%
902	0	1	0%
1433	0	3	0%
3306	3	3	100%
8000	1	2	50%
Total	20	31	65%

Table 6 - Port Results

These results indicates a potential improvement for the Shodan framework. Of all of the ports scanned and results recorded, Shodan only returned 65% of the records that were returned by ZMap.

5 CONCLUSION

5.1 Conclusions

From the results presented above, it is clear that Shodan is a competent port-scanning resource. It is also clear that Shodan isn't presenting a complete picture. ZMap has proven through these results that, for port-scanning, it can match and, at times, exceed the results from Shodan.

Given these facts, with the expanded port options, ZMap is an ideal tool to use in conjunction with Shodan. With the difference in ports, further exploration can be identified and acted upon. It is always beneficial to have multiple tools to increase the confidence in reported results.

6 FUTURE WORK

6.1 Future Direction

6.1.1 Expansion and Integration of ZMap with Shodan Projects

Expanding the use of ZMap in conjunction with Shodan projects will include the following:

- Exploration of ZMap's UDP Datagram Scan and ICMP Echo Scan to determine whether the integration of these functions will enhance ZMap's performance.

- Expanding ZMap to include the banner grab module will greatly expand the functionality and usability of ZMap.

With the inclusion of the above function and performance improvements, it will become increasingly more likely that the development of a comprehensive device scanning and identification system will be achievable.

7 REFERENCES

ADRIAN, D., DURUMERIC, Z., SINGH, G. and HALDERMAN, J.A., 2014. Zippier ZMap: Internet-Wide Scanning at 10 Gbps, *Teh 8th USENIX Workshop of Offensive Technologies (WOOT'14)* 2014.

BODENHEIM, R., 2014. *Impact of the Shodan Computer Search Engine on Internet-facing Industrial Control System Devices*, Air Force Institute of Technology Wright Patterson AFB OH Graduate School of Engineering and Management.

BODENHEIM, R., BUTTS, J., DUNLAP, S. and MULLINS, B., 2014. Evaluation of the ability of the Shodan search engine to identify Internet-facing industrial control devices. *International Journal of Critical Infrastructure Protection,* **7**(2), pp. 114-123.

DURUMERIC, Z., BAILEY, M. and HALDERMAN, J.A., An Internet-Wide View of Internet-Wide Scanning, *23rd USENIX Security Symposium*.

DURUMERIC, Z., WUSTROW, E. and HALDERMAN, J.A., 2013. ZMap: Fast Internet-Wide Scanning and its Security Applications, *22nd USENIX Security Symposium* 2013.

EVANS, D., 2011. *The Internet of Things: How the Next Evolution of the Internet is Changing Everything.* cisco.com: Cisco Internet Business Solutions Group.

GOLDMAN, D., 2013-last update, Shodan: The Scariest Search Engine on the Internet. Available: http://money.cnn.com/2013/04/08/technology/security/shodan/. [February, 2014].

H.S.PELEAZ, M., 2013. *What Can Happen within a Cyberterrorist Attack to the Electrical Grid of a Country?* .

LAWSHAE, R., 2014. *Hunting Botnets with ZMap.*

MAPMYFITNESS, 2014. *Map My Run - GPS Running and Workout Tracking with Caloire Counting.* 5.10.2 edn. Apple iTunes: Apple.

MATHERLY, J., 2015-last update, Shodan. Available: www.shodan.io.

MEUNIER, F., WOOD, A., WEISS, K., HUBERTY, K., FLANNERY, S., MOORE, J., HETTENBACH, C. and LU, B., 2014-last update, The 'Internet of Things' Is Now: Connecting the Real Economy. Available: www.morganstanley.com/what-we-do/research [April 10, 2015].

PATTON, M., GROSS, E., CHINN, R., FORBIS, S., WALKER, L. and CHEN, H., 2014. Uninvited Connections: A Study of Vulnerable Devices on the Internet of Things (IoT), *2014 IEEE Joint Intelligence and Security Informatics Conference*, September 24-26, 2014 2014, pp. 232-235.

PRESS, G., 2014. *The Two Forces Driving the Internet of Things.* Forbes.

PUJOL, E., RICHTER, P., CHANDRASEKARAN, B., SMARAGDAKIS, G., FELDMAN, A., MACDOWELL MAGGS, B. and NG, K.C., 2014. Back-Office Web Traffic on The Internet, *Proceedings of the 2014 Conference on Internet Measurement* 2014, pp. 257-270.

RADVANOVSKY, B., 2014. *Project SHINE (SHodan INtelligence Extraction) Findings Report.* Infracritical.

RAPID7-SONAR, 2015-last update, Project Sonar by Rapid7. Available: https://sonar.labs.rapid7.com/ [04/18, 2015].

SCHLOESSER, M., 2013. *Scanning All The Things.*

UNKNOWN, 2013a. *Internet Census 2012: Port Scanning /0 using Insecure Embedded Devices, Carna Botnet.* http://internetcensus2012.bitbucket.org/paper.html edn.

UNKNOWN, 2013b-last update, Network Port. Available: http://simple.wikipedia.org/wiki/Network_port2015].

WILCOX, R., 2015-last update, When THINGS attack! Defending data centres from IoT device-krieg. Available: http://www.theregister.co.uk/2015/04/27/when_fridges_attack/?page=3 [05/01, 2015].

WILLIAMS, P.M., 2014. *Distinguishing Internet-facing ICS Devices Using PLC Programming Information*, Air Force Institute of Technology Wright-Patterson AFB OH Graduate School of Engineering and Management.

WUSTROW, E., DURUMERIC, Z. and HALDERMAN, J.A., 2014-last update, ZMap Documentation. Available: https://zmap.io/documentation.html.

APPENDIX A

ZMap Command Line Flags

COMMON OPTIONS	
-p, --target-port=port	TCP port number to scan (e.g., 443)
-o, --output-file=name	Write results to this file
-b, --blacklist-file=path	File of subnets to exclude, in CIDR notation, one per line
SCAN OPTIONS	
--n, --max-targets=n	Cap the number of targets to probe.
-N, --max-results=n	Exit after receiving this many results
-t, --max-runtime=secs	Cap the length of time for sending packets
-r, --rate=pps	Set the send rate in packets/sec
-B, -bandwidth=bps	Set the send rate in bits/sec. This overrides the –rate flag
-c, --cooldown-time=secs	How long to continue receiving after sending has completed
-e, --seed=n	Seed used to select address permutation
--shards=n	Split up scan into N shards/partitions among different instances of Zmap. When sharding –seed required
--shard=n	Set which shard to scan. Shards indexed in the range [0,N). When sharding –seed required
-T, --sender-threads=n	Threads used to send packets
-P, --probes=n	Number of probes to send to each IP
-d, --dryrun	Print each packet instead of sending

NETWORK OPTIONS	
-s, --source-port=port|range	Source port(s) to send packets from
-S, --source-ip=ip|range	Source address(es) to send packets from
-G, --gateway-mac=addr	Gateway MAC address to send packets to
-I, --interface=name	Network interface to use
PROBE OPTIONS	
--list-probe-modules	List available probe modules
-M, --probe-module=name	Select probe module
-probe-args=args	Arguments to pass to probe module
--list-ouput-fileds	List fileds module can send to output module
OUTPUT OPTIONS	
--list-output-modules	List available output modules
-O, --output-module=name	Select output module
--probe-arg=args	Arguments to pass to output module
-f, --output-fields=fields	Comma-separated list of fields to output
--output-filter	Specify filter over fields defined by probe module

ADDITIONAL OPTIONS	
-C, --config=filename	Read configuration file
-q, --quiet	Do not print status updates once per second
-g, --summary	Print config and results summary at end of scan
-v, --verbosity=n	Level of log detail (0-5, default = 3)
-h, --help	Print help and exit
-V, --version	Print version and exit
TCP SYN SCANS	
-p, --target-port=port	TCP port number to scan
-s, --source-port=port|range	Source port(s) for scan packets

RESULTS OUTPUT	
-o, --output-file=p	File to write output to
-O, --output-module=p	Invoke a custom output module
-f, --output-fields=p	Comma-separated list of fields to output
--output-filter=filter	Specify output filter over fields for given probe
--list-output-modules	List available output modules
--list-output-fields	List available output fields for a given probe

BLACKLISTING AND WHITELISTING	
-b, --blacklist-file=path	File of subnets to blacklist in CIDR notation
-w, --whitelist-file=path	File of subnets to limit scan to in CIDR notation
RATE LIMITING AND SAMPLING	
-r, --rate=pps	Set maximum send rate in packets/second
-B, --bandwidth=bps	Set send rate in bits/second. Overrides the –rate flag
-n, --max-targets=n	Cap number of targets to probe
-N, --max-results=n	Cap number of results
-t, --max-run-time=seconds	Cap length of time for sending packets
-s, --seed=n	Used to select address permutation.
SENDING MULTIPLE PACKETS	
-p, --probes=n	Number of unique probes to send to each IP

WRITING PROBE & OUTPUT MODULES	
--list-probe-modules	List installed probe modules
--list-output-modules	List installed output modules

APPENDIX B

ZMap Output Fields

FIELD TITLE	TYPE	DESCRIPTION
saddr	string	Source IP address of response
saddr-raw	int	Network order integer form of source IP address
daddr	string	Destination IP address of response
daddr-raw	int	Network order integer form of destination IP address
ipid	int	IP identification number of response
ttl	int	Time-to-live of response packet
sport	int	TCP source port
dport	int	TCP destination port
seqnum	int	TCP sequence number
acknum	int	TCP acknowledgement number
window	int	TCP window
classification	string	Packet classification
success	int	Is response considered a success?
repeat	int	Is response a repeat response from host?
cooldown	int	Was response received during the cooldown period?
timestamp-str	string	Timestamp of when response arrived in ISO8601 format
timestamp-ts	int	Timestamp when response arrived (secs) since epoch
timestamp-us	int	Microseconds since timestamp-ts

APPENDIX C

IANA Special Purpose Address Registry

Address Block	Name
0.0.0.0/8	"This host on this network"
10.0.0.0/8	Private-Use
100.64.0.0/10	Shared Address Space
127.0.0.0/8	Loopback
169.254.0.0/16	Link Local
172.16.0.0/12	Private-Use
192.0.0.0/24[2]	IETF Protocol Assignments
192.0.0.0/29	IPv4 Service Continuity Prefix
192.0.0.8/32	IPv4 dummy address
192.0.0.170/32, 192.0.0.171/32	NAT64/DNS64 Discovery
192.0.2.0/24	Documentation (TEST-NET-1)
192.31.196.0/24	AS112-v4
192.52.193.0/24	AMT
192.88.99.0/24	Deprecated (6to4 Relay Anycast)
192.168.0.0/16	Private-Use
192.175.48.0/24	Direct Delegation AS112 Service
198.18.0.0/15	Benchmarking
198.51.100.0/24	Documentation (TEST-NET-2)
203.0.113.0/24	Documentation (TEST-NET-3)
240.0.0.0/4	Reserved
255.255.255.255/32	Limited Broadcast

APPENDIX D

Project Ports and IP Addresses

Ports Scanned

1 – 2056	2323	3306	8000	8080

Ports with Results

Port	Name
21	FTP
22	SSH
80	HTTP
443	HTTPS
3306	MYSQL
8000	QCONN WSGLSERVER

IP Addresses

Alias	IP
MA_CIDR1	128.169.27.128/25
MA_CIDR2	128.196.146.98/27
MA1	128.196.146.120
MA2	128.196.146.106
H_VA	54.173.35.215
H_CA	54.191.186.152
H_OR	54.183.248.109

APPENDIX E

Shodan Ports

Port	Name	Port	Name
7	Echo	443	HTTPS
11	Systat	445	SMB
13	Daytime	465	SMTP + SSL
15	Netstat	500	IKE
17	Quote of the day	502	Modbus
19	Character Generator	515	Line Printer Daemon
21	FTP	523	IBM DB2
22	SSH	623	IPMI
23	Telnet	626	serialnumbered
25	SMTP	631	CUPS
37	rdate	771	RealPort
53	DNS	789	Red Lion
67	DHCP	992	Telnet + SSL
79	Finger	993	IMAP + SSL
80	HTTP	995	POP3 + SSL
81	HTTP (81)	1023	Telnet (1023)
82	HTTP (82)	1200	Codesys
83	HTTP (83)	1234	Udpxy
84	HTTP (84)	1434	MS-SQL Monitor
88	Kerberos	1471	Hak5 Pineapple
102	Siemens S7	1604	Citrix
110	POP3	1723	PPTP
111	Portmap	1900	UPnP
119	NNTP	1911	Tridium Fox
123	NTP	1962	PCWorx
129	Password generator protocol	2067	DLSW
137	NetBIOS	2082	cPanel
143	IMAP	2083	cPanel + SSL
161	SNMP	2086	WHM

Port	Name	Port	Name
389	LDAP	2087	WHM + SSL
2123	GPRS Tunneling Protocol	5094	HART-IP
2152	GPRS Tunneling Protocol	5222	XMPP
2323	Telnet (2323)	5353	mDNS
2375	Docker	5357	Microsoft-HTTPAPI/2.0
2376	Docker + SSL	5432	PostgreSQL
2404	IEC-104	5560	Oracle HTTP
2455	Codesys	5632	PC Anywhere
2628	Dictionary	5900	VNC
3000	ntop	5901	VNC (5901)
3128	Squid Proxy	5985	WinRM 2.0
3306	MySQL	5986	WinRM 2.0 + SSL
3386	GPRS Tunneling Protocol	6000	X Windows
3388	RDP (3388)	6379	Redis
3389	RDP	6666	Voldemort
3479	2-Wire RPC	7071	Zimbra HTTP
3780	Nexpose	7547	Modem Web Interface
3790	Metasploit	7657	HTTP (7657)
4022	Udpxy	7777	Oracle
4040	Chef	8000	Qconn
4369	Erlang Port Mapper Daemon	8069	OpenERP
4443	Symantec Data Center Security	8080	HTTP (8080)
4500	IKE-NAT-T	8087	Riak Protobuf
4911	Tridium Fox + SSL	8089	Splunk
4949	Munin	8090	Insteon Hub
5000	Synology	8098	Riak Web Interface
5001	Synology	8129	Snapstream
5006	Mitsubishi MELSEC-Q	8139	Puppet Agent
5007	Mitsubishi MELSEC-Q	8140	Puppet Master
5008	NetMobility	8181	GlassFish Server
5060	SIP	8333	Bitcoin

Port	Name	Port	Name
8443	HTTPS (8443)	16010	Hbase
8834	Nessus	18245	General Electric SRTP
8888	AndroMouse	18246	General Electric SRTP
9000	NAS Web Interfaces	20000	DNP3
9051	Tor control port	20547	ProConOS
9100	Printer Job Language	25565	Minecraft
9151	Tor control port	27017	MongoDB
9160	Cassandra	28017	MongoDB Web Interface
9200	ElasticSearch	32764	Router backdoor
9600	OMRON FINS	44818	EtherNetIP
9943	Pipeline Pilot + SSL	47808	BACnet
9944	Pipeline Pilot	49152	Supermicro Web Interface
9981	HTS/ tvheadend	50100	Telnet
9999	Telnet (Lantronix)	55553	Metasploit (55553)
10000	Webmin	55554	Metasploit (55554)
10001	Automated Tank Gauge	62078	iPhone
10243	Microsoft-HTTPAPI/2.0	64738	Mumble server
11211	MemCache		

APPENDIX F

Acronyms

Acronym	Definition
ACK	Acknowledge
API	Application Program Interface
CDN	Content Delivery Network
CIDR	Classless Inter-Domain Routing
CSV	Comma-Separated Values
DAP	Data Analysis Pipeline
DB	Database
FTP	File Transfer Protocol
DDoS	Distributed Denial of Service
GUI	Graphical User Interface
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
IANA	Internet Assigned Numbers Authority
ICMP	Internet Control Message Protocol
ICS	Industrial Control System
IOT	Internet of Things
IP	Internet Protocol
IPv4	Internet Protocol Version 4
ISP	Internet Service Provider
JSON	JavaScript Object Notation
OS	Operating System
NTP	Network Time Protocol
PLC	Programmable Logistic Controller
SCADA	Supervisory Control And Data Acquisition
SHINE	SHondan INtelligence Extraction
SME	Subject Matter Expert
SSH	Secure Shell
SYN	Synchronize
TCP	Transmission Control Protocol
TXT	Text, as in text file
UDP	User Data Protocol