Identification of Visible Industrial Control Devices at Internet Scale

Xuan Feng*, Qiang Li†, Qi Han‡, Hongsong Zhu*, Yan Liu §, Limin Sun*
* Institute of Information Engineering, Chinese Academy of Sciences, China
†School of Computer and Information Technology, Beijing Jiaotong University, China
§ College of Software and Microelectronics, Peking University, China
‡Department of EECS, Colorado School of Mines, Golden, CO USA

***Abstract*—Nowadays industrial control devices are crucial for infrastructure-critical systems such as factories, power plants, and water treatment facilities. Devices with IP addresses are visible on the Internet and they connect cyber space and physical world. The first step in protecting devices from attackers is a deep understanding of the devices' characteristics in the cyber space. In this paper, we take a first step in this direction by investigating physical devices running one of the two specific protocols that are widely adopted in industrial control systems. In order to detect these devices in real-time, we propose a two-stage discovery mechanism: first filtering out unqualified hosts from 4 billion remote hosts and then identifying physical devices from qualified candidates. We have conducted a real-world experiment to verify the mechanism and identified dozens of thousands of physical devices from the entire Internet. Results show that our method discovers all devices in 20 hours with 89.5% precision and 79.3% recall.**

I. Introduction

The number of industrial control equipments with computing and communication capabilities is rising dramatically and these devices are visible on the Internet via IP addresses. Supervisory control and data acquisition (SCADA) [1] is often used to control remote equipments with coded signals, and these equipments are typically computer-based systems with access to the Internet. They are crucial for infrastructure-critical systems such as power, oil, and gas pipelines, and water distribution and wastewater collection systems. Protecting these systems' security is particularly important because they bridge cyber space and physical world. The first step is to find these devices and understand their usage characteristics in the cyber space. In this paper, we take a first step in addressing this knowledge gap.

There are several previous research works focusing on exploiting the cyber space from live host detection [2] and cyber scan [3] to security event detection [4] and abnormality analysis [5]. In contrast, we would like to explore the industrial control devices at Internet scale. The findings will help ensure security of infrastructure-critical systems in cyber space. This is because Internet has made visible devices more vulnerable to attacks, leading to security issues in critical infrastructures. For instance, the event of a cyber-attack on an industrial system of a German steel mill in 2014 resulted in massive damages [6]. Discovering these industrial control equipments help employ preventive measures.

Directly detecting physical devices is faced with two major challenges. Most physical devices have application protocols built over TCP/IP that are different from commonly used application protocols such as HTTP or FTP. For example, the Modbus protocol [7] operates over TCP/IP, but it uses its own signal code as application communication protocol to exchange data between devices and access data from industrial control systems. In addition, there are 4 billion IPv4 addresses, so detecting all physical devices is time-consuming. However, identification of device usage and their distribution needs to be done in a timely manner.

To address these challenges, we investigate two industrial protocols that are widely adopted for running physical devices in industrial control systems. In particular, we present an analysis of the Modbus and S7 protocols [7] and propose a mechanism for discovering the physical devices running either of these two protocols. Although only two device protocols are studied, our approach can be easily extended to other types of industrial devices with minor modifications. At the first stage, we scan all IP addresses with the particular port to filter out equipments that definitely do not run any of these two protocols and keep the candidates that have a high probability of running Modbus and S7 protocols. At the second stage, we prune the number of application packets of Modbus and S7 protocols into merely one packet probe, as the speed of discovering physical devices can be improved by reducing the count of packet sending.

To verify our discovery mechanism of physical devices, we implemented it using go scripts [8], which send Modbus and S7 physical device discovery packets. We ran our system on Cloud computing server, Amazon EC2 [9] to investigate the entire IP addresses to discover these physical devices. The results show that our approach can discover all devices in 20 hours with 89.5% precision and 79.3% recall.

In brief, we have made the following two contributions:

- We have investigated two industrial protocols running on physical devices and proposed a two-stage discovery mechanism for discovering all industrial control devices from the large number of IP addresses in real-time.
- We have implemented the discovery mechanism with go

† Hongsong Zhu is corresponding author.

978-1-4799-6664-6/16/$31.00 ©2016 IEEE

scripts and run it on Amazon EC2. We have probed the entire Internet space three times and the exploration took 20 hours each time.

The rest of the paper is structured as follows. We describe the related work in Section II. We then propose our two-stage discovery mechanism in Section III. In Section IV, the real world experiments are conducted. Finally, we make final remarks in Section V,VI.

II. Related Work

Detecting remote industrial control device is a cross-cutting research topic related to both network measurement and security of cyber-physical systems.

Network measurement or network reconnaissance is a primary stage of discovering and remotely locating vulnerable systems. Previous work [10] [11] can discover lots of device information by sending a combination of complicated packets to every host. This process is time consuming, so it is only suitable for a small-scale network space. In contrast, our work aims to discover as much device information as possible in large-scale network space. The network reconnaissance research [12]–[17] mainly focuses on speeding up the measurement. Over time it does get progressively faster - from 4 months down to 30 days, one day, and more recently 45 minutes. For instance, Xie et.al [12] proposed the UDmap algorithm to identify and analyze hosts across the entire IP address space. Heidemann et.al [11] explored the visible Internet to characterize the edge hosts and evaluate their usages via active scanner within 30 days of latency. Hong et.al [15] searched the entire Internet and then classified IP addresses as popular or unpopular. Leonard et.al [13] implemented IRLscanner as an Internet-wide detection mechanism within 24 hours. It focused on typical application layer protocols such as HTTP, SMTP. The difference of our work is that we focus on physical device detection. Durumeric et.al [17] proposed Zmap for Internet-wide host detection within 45 minutes. Its time is a theoretical upper bound and cannot be achieved in practice due to network congestion. Zmap only sends one packet to every IP address to find live host but our approach aims to extract device information as much as possible.

Research on security of cyber-physical systems (CPS) [18]–[21] is an emerging field. An industrial control system is a classic case of CPS. For remote access, industrial control devices should be access-controlled by a firewall or a VPN. However, SCADA-enabled devices expose themselves in the public Internet, enabling attackers to remotely locate and subsequently exploit vulnerable parts. Igure et.al [22] presented serious security problems about SCADA-enable devices that network reconnaissance could discover devices and exploit their vulnerabilities. Our work is a first step to identify industrial devices and characteristics for preserving devices' security in the cyber space. Durumeric et.al [4] proposed to detect a security event (a OpenSSL flaw "Heartbleed") at Internet scale. If similar problems arise in an industrial control system, a real-time discovery mechanism is necessary. Ten et.al [23] presented the vulnerability assessment of industrial

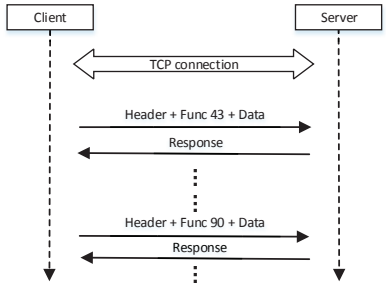

Fig. 1: Interactions between server and device running the Modbus protocol.

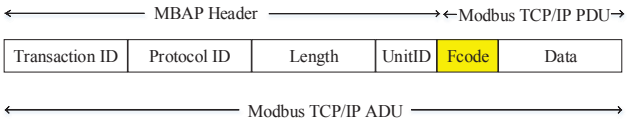

Fig. 2: Modbus Protocol Header

control systems. It is only suitable at a small scale, but our work is aimed at Internet scale. Yang et.al [5] provided anomaly-based intrusion detection for SCADA-enable devices. Our work can help it to enumerate physical devices hosts and shed light on compromised devices. We can also help system administrators do security auditing, discovering and patching servers with critical security vulnerabilities [24]. In brief, we probe large subsets of the public IP address space and discover industrial control devices at Internet scale.

III. Physical Device Discovery Mechanism

In this section, we present the details on how to discover industrial control devices in the Internet. We firstly analyze two industrial protocols commonly used in industrial control systems and propose how to use them to identify those physical devices among various hosts. Then, we propose a two-stage detection mechanism for discovering and recognizing all the physical devices within certain time constraints.

A. Industrial Protocols for Identifying Devices

Different physical devices typically use different application protocols. These protocols have a header to indicate the type of the device. However, it is prohibitively expensive to enumerate every protocol to identify device types. Here, we study two typical industrial protocols that are widely adopted in industrial control systems, S7 protocol - SIMATIC Programmable Logic Controller- and Modbus protocol - Schneider Electric Modicon Programmable Logic Controller. S7 and Modbus are used as application protocols to send configurations and control commands to industrial equipments. Although only two particular protocols are studied, our approach can be extended to other types of industrial devices with minor modifications.

Modbus is an industrial protocol that operates over TCP/IP for standardized and recognized communication protocols.

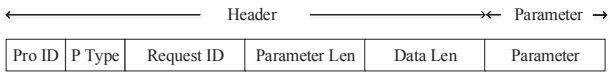

Fig. 3: S7 Protocol Header

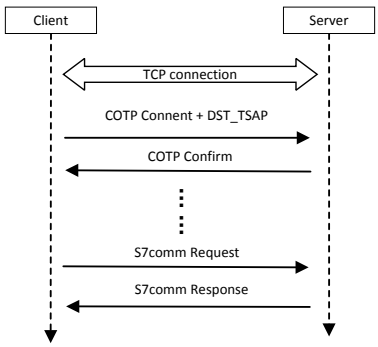

Fig. 4: Interactions between server and device running the S7 protocol.

It can send signal code to configure and control industrial equipments remotely. As shown in Figure 1, devices running Modbus first need to build the TCP connection, before exchanging the data with each other. Every time devices transmit data, it carries a function code, labeled as "Fcode". Figure 2 shows its protocol structure, where "Fcode" occupies 1 byte, specifying a particular function to regulate the purpose of the data, such as 04 (read input register), 03 (Read Holding Registers). Combining these two figures, if we send a specific packet satisfying the Modbus protocol standard, we can identify whether the host is the physical device running Modbus or not from the response. Therefore, we send a probe packet with payload encapsulating the function code. If the host responds to the function code request, we label the host as a physical devices running Modbus, otherwise not.

As Modbus, S7 protocol is also an industrial protocol encapsulating TCP/IP, sending signal command and transmitting data. It is a Siemens proprietary protocol belonging to the Siemens S7-300/400 family. Different from Modbus, S7 has no fixed function code like Modus "Fcode" to identify whether the device is running it or not. As shown in Figure 3, S7 protocol structure regulates each byte usage in its header and the particular header field "type" reveals the particular device running it. What makes the detection of these devices tricky is that before receiving a S7 comm response packet there is another connection process on ISO layer with Connection-Oriented Transport Protocol (COTP) [25]. As shown in Figure 4, after the common TCP connection is established, the device first sends a COTP packet to verify whether the industrial equipment is running S7 comm or not. If the device passes the verification, we process the next stage for the S7 comm protocol to control and transmit data. It is vital for passing the COTP confirmation that the signal "DST_TSAP" must be the same value as the physical

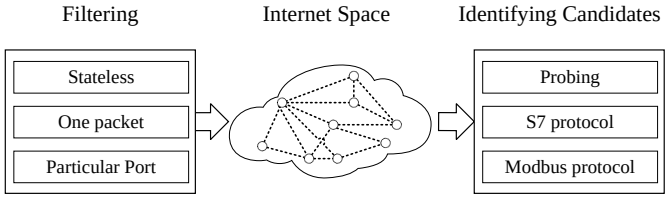

Fig. 5: A two-stage mechanism for real-time detection of physical devices running S7 or Modbus

industrial equipment pre-labeled, otherwise this process can not be permitted. The "DST_TSAP" filed has two bytes, where the first byte indicates the communication type (1=PG, 2=OP) and the second byte is the rack and slot number to present the position of the industrial equipments' CPU. The second byte of "DST_TSAP" filed must match with the position of equipments' CPU. The position is labeled as two fields with the rack and slot number, and the second byte has lower Bits (0-4) to match slot number and the Bits (5-7) to match the rack number.

By analyzing these two industrial protocols, we can identify which devices are running S7 protocol or Modbus protocol only by sending packet to an IP address with their protocol standard regulation. In general, these industrial protocols always run in industrial control device, so we can infer that the place where S7 or Modbus appear has a high probability that industrial control devices are present.

B. Two-stage Detection Mechanism

As aforementioned, we can utilize the standard application protocols to discover physical devices running S7 or Modbus. While the idea seems simple, a challenge arises in practice when the discovery needs to be done in real-time. In IPv4 space, there are nearly 4 billion addresses. As shown in Fig. 1 and Fig. 4, TCP connection is first built which introduces certain latency. Moreover, Modbus needs extra packets to traverse each type "Fcode" to dig out device information and S7 requires a COTP confirmation which requires correct value of "DST_TSAP" field in its header. To gather this information, each IP address needs more than a dozen probing packets in order to collect enough information to determine whether this host is a physical device or not. Assuming each address takes several seconds of latency, the time spent over 4 billion addresses will be several months, which is infeasible in practice.

To deal with this practical challenge, we propose a two-stage mechanism for physical devices detection in real time. As shown in Figure 5, we first use existing techniques to filter out those devices that have a low probability of running S7 or Modbus from 4 billion hosts, then we identify qualified devices by sending standard S7 and Modbus protocol probing packets.

At the first stage, to speed up discovery, we apply the following three existing techniques to filter out unqualified hosts.

TABLE I: Physical devices discovery at Internet scale

Begin Time	IP Space	Protocol	Candidates Amount	Physical Devices
2015-08-31	3.7 billion	Modbus/S7	2.18 million	23231
2015-09-02	3.7 billion	Modbus/S7	2.12 million	23346
2015-09-05	3.7 billion	Modbus/S7	2.27 million	23207

TABLE II: Resource usage of our two-stage discovery mechanism

CPU usage	Network usage (in)	Network usage (out)
53%	0.5Mbps	50Mbps

- Stateless. Before probing application protocol, building TCP connection needs to store every state which results in longer time delay. Recent tool Masscan [26] suggested that stateless connection makes probing fast while guaranteeing nearly 95% accuracy. We have also used stateless connection to speed up the filtering process.
- One packet. More packets used to discover physical devices needs more time. Recent research Zmap [17] only uses one packet with random algorithm and can cover nearly 90% IP address. We adopt this approach by only sending out one packet to a random address each time to discover physical devices.
- Particular Port Selection. We adopt a particular port to filter out those devices that are less likely to be running S7 or Modbus. We select 502 port and 102 port as the filtering indication, when devices running the Modbus or S7 protocols, they have a high probability to open these two ports.

At the second stage, we verify the candidates to identify whether it is a physical device running S7 or Modbus protocols. After the first stage of filtering, the number of candidates is greatly reduced to the order of millions compared with the initial 4 billions. For each candidate, we first conduct three-way handshakes using TCP to build state connection. To identify Modbus protocol, we send a probing packet encapsulating its standard header and data payload. To get more information, we set up "Fcode" field to extract more information of devices. To identify S7 protocol, we traverse packets for finding correct value of "DST_TSAP" field and pass through the COTP connection. We then build S7 communication connection and gain more information of devices.

IV. Performance Evaluation

In this section, we first describe the implementation details and how we perform our scans, then present performance evaluation of our proposed two-stage mechanism.

A. Implementation

We implemented the proposed two-stage discovery mechanism. In the first stage, we define a single TCP packet that has "SYN" filed in header and "NULL" data payload. By reusing the stateless connection and IP address randomization

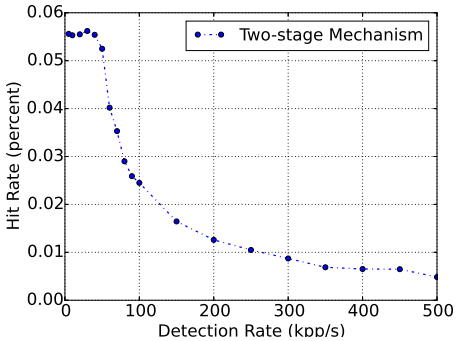

Fig. 6: The hit rate and detection rate for internet-wide physical device discovery

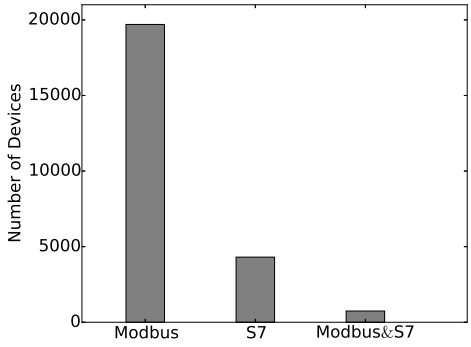

Fig. 7: Physical devices running S7 or Modbus

algorithms of the network scanner tool Zmap [17], we send the probe to every IP address with port "502" and port "102". If the host responds with "SYN-ACK" in the header, we put this host into the set of candidates, otherwise throw it away. The candidates are stored as a JSON file to be handled at next phase. In the second stage, we implement the S7 and Modbus protocols parsing scripts using the go language [8]. For each candidate, we send a "DST_TSAP" field request to pass COTP connection and communicate by S7comm to get information, and send a "Fcode" field request to get Modbus information. Finally, the detailed information about physical devices is stored in a plain text file.

We deployed our two-stage mechanism program on Cloud computing server, Amazon EC2 [9]. It is Ubuntu 14.04.2 LTS (GNU/Linux 3.13.0-48-generic x86_64) with 2 vCPU and 8 GB of memory, 450Mbps bandwidth. After probing the entire IP space, we fetch the date set from the server and use Python to preprocess and extract useful information (such as IP, timestamp, response information).

B. Experimental Results

Using the prototype system implemented by us, we conducted the experiments three times from August to September 2015 and the details are shown in Table I. Each time we have exhaustively searched the entire IP address space (close to 3.7 billion addresses). Every time we excluded both reserved/unallocated IP space from IANA [27] and IPs which

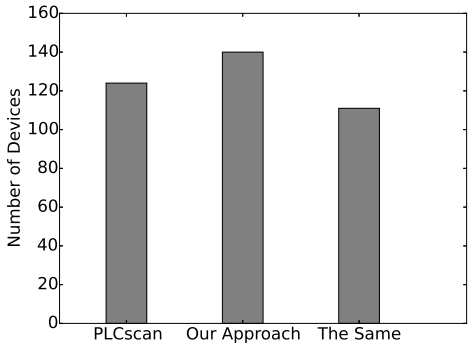

Fig. 8: Comparison of our approach with PLCscan

TABLE III: The precision and recall of our approach

	Condition positive	Condition negative
Test outcome positive	111	13
Test outcome negative	29	3972

send some emails to complain our scan activity. Altogether we have added about 610 million IP address into our blacklist to exclude them from the Internet-wide range. We have measured the CPU, memory usage and bandwidth usage for the two-stage detection mechanism prototype system. As shown in Table II, the average CPU usage is 53%, network usage (in) is 0.5Mbps, and network usage (out) is 50Mbps, about 10% of our bandwidth. It is acceptable in practice. Each time, we were able to discover physical devices running S7 or Modbus. In the first stage, we discovered nearly 2.2 million candidates and we further narrowed down the list to more than 23,000 physical devices at the second stage.

We have conducted an experiment to measure the latency of our two-stage mechanism. As shown in Figure 6, there is a tradeoff between the hit rate and detection rate. The hit rate is equal to $N_{candidates}/N_{total}$, where N_{total} is the total number of IP addresses and $N_{candidates}$ is the number of responding hosts. The higher the hit rate, the more candidates we can include. Detection rate is the speed of discovering physical devices, i.e. the latency incurred in our Internet-wide discovery. When the detection rate is 50,000 packets per second, we can get a stable hit rate for discovering physical devices. Although Zmap [17] claims that it can finish the entire IP space detection under 45 minutes, the time is only a theoretical bound and it only includes host discovery. Our approach collects more information about physical devices than survival scan by Zmap [17]. A practical issue is that network congestion reduces the hit rate. We adopt the most stable detection rate while keeping the hit rate high. In Table I, for each data collection we use 50,000 packets per second, and each experiment is finished in about 20 hours. Our approach is able to discover the largest amount of candidates and physical devices, and its time latency is acceptable in practice.

Furthermore, we verify the precision of our two-stage discovery mechanism. Because device detection occurs in the remote Internet space, there is no ground truth about the device except for its IP address. To gain the ground truth, we use the traditional tool PLCscan [28] to find out physical devices running S7 or Modbus protocol. It is a utility that was released to identify SCADA devices [1] on the network. We choose a $/20$ subnet with 4096 IP address to detect physical devices. As shown in Figure 8, our two-stage mechanism have found 140 physical devices running S7 or Modbus protocols and PLCscan merely found 124. We further compare the coverage range of two approaches, and 111 devices are the same in both scans. Using PLCscan results as the ground truth, our precision is 89.5%, and recall is 79.3% as shown in Table III. However, our approach only takes 5s , while PLCscan needs more than 1 hour. This is because that PLCscan needs to traverse every IP address with state connection and complicated packets probing process. In contrast, we use one packet and stateless measurement at first and then do complicated S7 and Modbus identification.

Our two-stage detection mechanism can identify physical devices running Modbus or S7 protocol. Based on the collected data set (Table I), we further determine which protocol the device is using. As shown in Figure 7, there are almost 4,000 physical devices running only S7 protocol, 19,000 running only Modbus, nearly 800 running both protocols.

V. Discussion

Our work can be improved in three different areas.

Firewall and VPN. For remote access, industrial control devices should be access-controlled by a firewall or a VPN. However, we have collected information devices running Modbus and S7 protocols at Internet scale three times and found tens of thousands of devices being exposed to the public Internet. No matter what security protections these protocols have, these Internet-connected devices are at risk of security attacks. In fact, the Modbus protocol lacks of security protections. To approximate a "bird's eye view" of these devices at Internet scale, we can not estimate the total number and distribution of them. This is because that many devices are in the local network behind firewall or VPN. Passive listening [3] on the outgoing data traffic might be a good way to estimate the number of these devices, such as a listener at an Internet service provider.

Dynamic IP address. We use IP address to label an industrial control device on the Internet. IP address usually represents a device, but it is easily affected by DHCP with the dynamic joining and leaving of devices. In our experiments, we found that there are one third IP addresses are different. Hence, we need an approach to uniquely label an industrial control device at Internet scale.

General framework. Although only two device protocols are studied, our approach can be extended to other types of industrial devices with minor modifications. After analyzing the application-layer protocols, we identify a device by performing application-layer handshakes to get a response packet. With the increasing number of protocols in SCADA systems, an extensible automatic annotation for every type of devices will help researchers to identify a device. One promising way

is to use crowd-sourcing [29], i.e., to encourage people to contribute annotations for new types of devices.

VI. Conclusions

In this paper, we have proposed a two-stage mechanism to discover physical devices running one of the two popular industrial protocols: S7 and Modbus. We have implemented the mechanism in a simplified system and run it on Amazon EC2 to probe physical devices from the entire IPV4 space. We also have run the system three times to identify all visible physical devices running these two protocols and collected statistics. Results show that our approach can finish the probing with a high accuracy in a short time duration. In the future, we will propose a general framework to extend particular protocols to discover physical devices of other types. We will also further investigate the spatial and temporal relationship of these devices.

VII. Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (Grant No. U1536107), the Key Program of Institute of Information Engineering Chinese Academy of Sciences (No. Y5Z0151104) and "Strategic Priority Research Program" of the Chinese Academy of Sciences,Grant No. XDA06040101.

References

[1] S. A. Boyer, *SCADA: supervisory control and data acquisition*. International Society of Automation, 2009.

[2] W. Mühlbauer, A. Feldmann, O. Maennel, M. Roughan, and S. Uhlig, "Building an as-topology model that captures route diversity," in *ACM SIGCOMM Computer Communication Review*, vol. 36, no. 4. ACM, 2006, pp. 195–206.

[3] E. Bou-Harb, M. Debbabi, and C. Assi, "Cyber scanning: a comprehensive survey," *Communications Surveys & Tutorials, IEEE*, vol. 16, no. 3, pp. 1496–1519, 2013.

[4] Z. Durumeric, J. Kasten, D. Adrian, J. A. Halderman, M. Bailey, F. Li, N. Weaver, J. Amann, J. Beekman, M. Payer *et al.*, "The matter of heartbleed," in *Proceedings of the 2014 Conference on Internet Measurement Conference*. ACM, 2014, pp. 475–488.

[5] D. Yang, A. Usynin, and J. W. Hines, "Anomaly-based intrusion detection for scada systems," in *5th intl. topical meeting on nuclear plant instrumentation, control and human machine interface technologies (npic&hmit 05)*. Citeseer, 2006, pp. 12–16.

[6] K. Zetter, "A cyberattack has caused confirmed physical damage for the second time ever. wired, january 8, 2015," 2015.

[7] B. Drury, *Control techniques drives and controls handbook*. IET, 2001, no. 35.

[8] The go programming language. [Online]. Available: https://golang.org/

[9] Amazon elastic compute cloud (amazon ec2). [Online]. Available: https://aws.amazon.com/ec2/

[10] G. F. Lyon, *Nmap Network Scanning: The Official Nmap Project Guide to Network Discovery and Security Scanning*. Insecure, 2009.

[11] J. Heidemann, Y. Pradkin, R. Govindan, C. Papadopoulos, G. Bartlett, and J. Bannister, "Census and survey of the visible internet," in *Proceedings of the 8th ACM SIGCOMM conference on Internet measurement*. ACM, 2008, pp. 169–182.

[12] Y. Xie, F. Yu, K. Achan, E. Gillum, M. Goldszmidt, and T. Wobber, "How dynamic are ip addresses?" in *ACM SIGCOMM Computer Communication Review*, vol. 37, no. 4. ACM, 2007, pp. 301–312.

[13] D. Leonard and D. Loguinov, "Demystifying service discovery: implementing an internet-wide scanner," in *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement*. ACM, 2010, pp. 109–122.

[14] Q. Li, Q. Han, and L. Sun, "Collaborative recognition of queuing behavior on mobile phones," *Mobile Computing, IEEE Transactions on*, vol. 15, no. 1, pp. 60–73, 2016.

[15] C.-Y. Hong, F. Yu, and Y. Xie, "Populated ip addresses: classification and applications," in *Proceedings of the 2012 ACM conference on Computer and communications security*. ACM, 2012, pp. 329–340.

[16] Q. Li, Q. Han, and L. Sun, "Context-aware handoff on smartphones," in *Mobile Ad-Hoc and Sensor Systems (MASS), 2013 IEEE 10th International Conference on*. IEEE, 2013, pp. 470–478.

[17] Z. Durumeric, E. Wustrow, and J. A. Halderman, "Zmap: Fast internet-wide scanning and its security applications." in *Usenix Security*, 2013, pp. 605–620.

[18] E. Lee *et al.*, "Cyber physical systems: Design challenges," in *Object Oriented Real-Time Distributed Computing (ISORC), 2008 11th IEEE International Symposium on*. IEEE, 2008, pp. 363–369.

[19] E. K. Wang, Y. Ye, X. Xu, S. Yiu, L. Hui, and K. Chow, "Security issues and challenges for cyber physical system," in *Proceedings of the 2010 IEEE/ACM Int'l Conference on Green Computing and Communications & Int'l Conference on Cyber, Physical and Social Computing*. IEEE Computer Society, 2010, pp. 733–738.

[20] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," *Computer networks*, vol. 54, no. 15, pp. 2787–2805, 2010.

[21] Q. Li, Q. Han, X. Cheng, and L. Sun, "Queuesense: Collaborative recognition of queuing on mobile phones," in *Sensing, Communication, and Networking (SECON), 2014 Eleventh Annual IEEE International Conference on*. IEEE, 2014, pp. 230–238.

[22] V. M. Igure, S. A. Laughter, and R. D. Williams, "Security issues in scada networks," *Computers & Security*, vol. 25, no. 7, pp. 498–506, 2006.

[23] C.-W. Ten, C.-C. Liu, and G. Manimaran, "Vulnerability assessment of cybersecurity for scada systems," *Power Systems, IEEE Transactions on*, vol. 23, no. 4, pp. 1836–1846, 2008.

[24] S. Cheung, B. Dutertre, M. Fong, U. Lindqvist, K. Skinner, and A. Valdes, "Using model-based intrusion detection for scada networks," in *Proceedings of the SCADA security scientific symposium*, vol. 46, 2007, pp. 1–12.

[25] Connection-oriented transport protocol. [Online]. Available: http://www.ietf.org/rfc/rfc0905.txt

[26] Masscan, network scanner tool. [Online]. Available: https://github.com/robertdavidgraham/masscan

[27] Assigned numbers authority (iana). [Online]. Available: http://www.iana.org/

[28] Plcscan plc devices detection on the network. [Online]. Available: https://code.google.com/p/plcscan/

[29] J. Howe, "The rise of crowdsourcing," *Wired magazine*, vol. 14, no. 6, pp. 1–4, 2006.