2018 IEEE International Conference on Industrial Internet (ICII)

Towards Automation of Vulnerability and Exploitation Identification in IIoT Networks

Nour Moustafa
School of Engineering and Information Technology
University of New South Wales at ADFA
Northcott Dr, Campbell ACT 2612, Canberra, Australia
Email: nour.moustafa@unsw.edu.au

Benjamin Turnbull
School of Engineering and Information Technology
University of New South Wales at ADFA
Northcott Dr, Campbell ACT 2612, Canberra, Australia
Email: b.turnbull@adfa.edu.au

Kim-Kwang Raymond Choo
Department of Information Systems and Cyber Security and Department of Electrical and Computer Engineering
University of Texas at San Antonio, San Antonio, TX 78249-0631, USA
Email: Raymond.choo@fulbrightmail.org

***Abstract*—Since Industrial Internet of Things (IIoT) networks are comprised of heterogeneous manufacturing and technological devices and services, discovering previously unknown vulnerabilities and their exploitation vectors (also known as Penetration Testing - PT) is an arduous and risk-prone process. PT across IIoT networks requires system administrators to attempt multiple and often bespoke commercial tools for testing vulnerable network nodes, platforms, and software. In this paper, we propose a new testbed IIoT environment involving multiple vulnerable platforms connected to IIoT sensors and IoT gateways for designing automated vulnerability and exploitation identification techniques based on analyzing network flows. We utilize a particle filter technique for estimating the vulnerability and exploitation behaviors in a term of posterior probabilities. The proposed model is better than using traditional artificial planning algorithms that consume significant computational resources and demand termination criteria. The proposed testbed IIoT environment can be shared with other like-minded researchers to facilitate future evaluations**

***Index Terms*—IIoT networks, penetration testing, AI planning, particle filter**

I. Introduction

Internet of Things (IoT) integrates both physical and digital objects to improve our daily tasks, in applications such as Industrial IoT (IIoT), smart homes and cities, smart grids, cyber-physical systems (CPS) and Internet of Battlefield/Military Things [1], [2], [3], [4]. The term IIoT was proposed in 2012 [3] to differentiate between consumer- and industrial-based applications, with IIoT focusing on applications that combine IoT technologies and the manufacturing process. The advances in this domain are strongly based on the Industry 4.0 paradigm, integrating IoT, IIoT, CPS, and big data analytics to offer IIoT services that incorporate operational, manufacturing and technological systems [5].

In IIoT networks, physical objects are transformed into cloud-based services comprising platforms, infrastructure, and software, where the cost of consumers will generally be lower than buying the actual physical devices. Industrial IoT objects, such as smart grid and weather actuators and sensors, are deployed in conjunction with middleware tools such as Arduino IDE [6], and then the generated data are sent to IoT hubs such as HIVEMQ [7]. While the latter is sent via traditional network protocols of OSI and/or TCP/IP models, emerging IoT-specific protocols such as Message Queue Telemetry Transport (MQTT) and Advanced Message Queuing Transport (AMQT), provide publishing/subscribing capability for automatically performing Machine-to-Machine (M2M) and Machine-to-People (M2P) services [2].

The security and privacy of IIoT technologies are key challenges due to three main issues [8]. Firstly, the heterogeneity of IoT objects makes them vulnerable to different cyber-attacks while connecting with existing network systems. Secondly, existing security tools including encryption, firewalls, intrusion detection systems [9], [10], are not generally suitable for IoT environments. There are multiple underlying reasons for this; the resource-constrained nature of IoT appliances, as well as the incompatibility of IoT standards and traditional network protocols. Finally, inconsistent maintenance and updating of IoT systems lead to difficulties testing vulnerabilities in IIoT networks. This stems from the strong need for availability over confidentiality or integrity. This mentality often results in a reduced impetus for patching, secure configuration and updates. Inconsistent maintenance can also be attributed to a lack of updates from vendors, or even an inability to update.

Due to these challenges, Penetration Testing (PT) tools and systems are employed to identify loopholes in networks and prevent them from exploitation [8], [11]. PT in IIoT networks is still in its early stage of research, and is reliant on commercial and bespoke tools, for example, the Nessus tool [12] for vulnerability discovery and the Metasploit framework [13] for exploitation. Current PT tools, whether across corporate networks or IIoT installations, lack the automation process of discovering and exploiting vulnerabilities [8], [11], [14]. This process in large-scale IIoT networks cannot discover all vulnerabilities due to the complexity and large numbers of current IIoT network resources [8]. The correlations of vulnerable hosts, paths of exploiting vulnerabilities, and probabilities of exploiting vulnerabilities demand further research

978-1-5386-7771-1/18/$31.00 ©2018 IEEE
DOI 10.1109/ICII.2018.00023

IEEE computer society

Authorized licensed use limited to: SLIIT - Sri Lanka Institute of Information Technology. Downloaded on February 24,2021 at 04:32:21 UTC from IEEE Xplore. Restrictions apply.

for designing intelligent PT tools in IIoT networks.

Artificial Intelligence (AI) planning approaches have been proposed for automating the PT process to prevent known weaknesses in networks [11], [14], [15], [16]. Attack vectors are basically designed by developing a sequence of steps to exploit a target, which is similar to the potential process of designing AI planning approaches. In PT, an attacker's philosophy invariably includes two stages [14]. The first (scanning and exploitation) stage is the process of finding vulnerabilities in host networks. An example of this would be the discovery of available services to search for the existence of SMBv1, a known vulnerable service that has been previously exploited by the Wannacrypt malware [17]. Exploit(s) would then be achieved by launching multiple malicious scripts, such as the Metasploit Eternalblue exploit [18], to exploit these vulnerabilities. The second stage is called a payload stage, executed by the successful scripts of exploitation for performing the attacker motive, such as steal information and/or corrupt resources.

The two stages include sequences of activities which execute attacking techniques. This is closely similar to the potential procedures of AI planning, where a set of deterministic actions (e.g., exploiting SMBv1 vulnerability) are constrained using preconditions (e.g., knowing network structures and platform configurations) for achieving the effect (e.g., successfully hacking the target). Well-known planning approaches used for automating PT include Markov Decision Process (MDP) and Partially Observable Markov Decision Process (POMDP) that is a generalization of MDP [11], [15]. Nevertheless, MDPs cannot estimate the partial information that hackers could have actions that they need, whilst POMDP cannot be operated in large-scale networks with precisely estimated probability distributions to execute proper actions successfully [11].

In this paper, we propose an intelligent PT technique for IIoT networks. The technique is designed based on utilizing network flow features, and then the posterior probabilities are estimated using a particle filter model to track the events of vulnerabilities and exploitations. We also design a testbed IoT network configuration with different host vulnerabilities for estimating the performance of the proposed technique compared to MDP and POMDP approaches. The testbed will be made available upon request, so that other researchers can use this testbed (or its configurations) for their research.

The rest of the paper is structured as follows. Section II discusses the background and related studies. The IIoT testbed environment and proposed particle filter technique are described in Section III. Section IV explains the experiments and discussions. Finally, the paper is concluded in Section V.

II. Background and previous studies

IoT is defined as a network of intelligent and self-organizing objects that enable interactions of users and Internet services for sharing information and resources [1]. The provision of online services for factories and organizations by the integration of industrial sensors, including water and gas levels, temperature, computational resources, and wireless technologies. Combining industrial wireless sensor networks and manufacturing services enables IIoT networks to interact with each other.

IIoT installations are uniquely vulnerable to cyber-attacks for multiple reasons; the scale and heterogeneity challenges of IIoT devices [19], as well as the lack of software and firmware updates. IoT botnets are one of the sophisticated cyber threats that have been responsible for several distributed denial of service (DDoS) attacks. IoT botnets have exposed the capability of large-scale exploitation of poor security across Internet-connected systems including routers, digital video recorders and surveillance cameras, home automation systems and other IoT devices [5].

Existing security mechanisms are not suited to secure IIoT networks because there is no standard that unifies the distinctions between IoT appliances and typical internet devices [1], [8]. The large numbers of IoT devices embedded in networks demand new protocols that manage these devices [2]. The diversity of protocols and lack of standards complicate the efforts to introduce security across IIoT nodes. The low computational ability of IIoT nodes combined with their low-power requirement reduces their ability to be truly secure. In such cases, there is a reliance on protection at a network level [2], [19].

PT tools embody discovery and attack techniques to complement security systems, both before IoT entities are deployed and during current deployment [12], [13], [20]. There are three testing types in regard to PT tools; interface testing, transportation testing and system testing. These testing types are as recommended in the OWASP guidelines for IoT systems [8]. This enables the PT process to be partially automated for finding weaknesses of IoT networks and successfully exploiting them. Interface testing determines input validation vulnerabilities of applications. Transportation testing examines flaws of designing protocols and cryptographic techniques. System testing analyzes loopholes of firmware, platforms, and system services.

AI Planning approaches [11], [14], [15], [21], [3] have been applied to design semi-automated hacking techniques. Fikes and Nilsson [22] proposed a planning method that contains an initial state, a goal state, and planning method for creating a set of actions of a planning problem. Leitner and Roderick [16] suggested a methodology that combines a machine learning algorithm with planning to create plans at runtime. This methodology enhanced the planning executions, as the learning strategy can infer the dynamic changes of states to predict suitable actions. In 2010, AI planning approaches have become one of the main focus in the PT domain, where adaptive planning techniques were used for testing security protocols [21].

Beurdouche et al. [3] proposed a FlexTLS tool for testing the Transport Layer Security (TLS) protocol and defining its actual parameters and sequence of events. Recently, Shmaryahu et al. [11] studied using the MDP and POMDP approaches to develop automated attack testing scenarios. However, existing approaches still suffer from the issues of

Authorized licensed use limited to: SLIIT - Sri Lanka Institute of Information Technology. Downloaded on February 24,2021 at 04:32:21 UTC from IEEE Xplore. Restrictions apply.

collecting more information about host networks and running on large-scale networks, as well as no available testbed configurations that implement them properly. To provide an intelligent PT technique for IIoT networks, the three testing types described above should be incorporated for coping with the heterogeneity and large numbers of devices in IIoT networks.

III. Research Methodology

This section presents our proposed testbed environment for IIoT networks, and the particle filter model for identifying vulnerabilities and their exploitations.

A. Proposed Testbed Configuration for IIoT Network

A testbed IIoT network environment is designed to identify vulnerabilities and exploitations, as depicted in Figure 1. The environment comprises three main components: (A) Network platforms and applications, (B) IIoT devices and sensors, and (C) IIoT services, as explained below. The components clarify how IIoT physical and digital objects interact together for demonstrating the applicability of the vulnerability and exploitation identification in IIoT networks.

(A) Network platforms and applications: The component includes four client VMware Operating Systems (OS): Microsoft Windows XP SP3, Microsoft Windows 7 X64, Mobile-database Ubuntu 14.04 LTS and Metasploitable2 [23], linked with a VMware Ubuntu server 14.04 LTS. The clients were configured to connect with a VMware Kali Linux that contains vulnerability scanners and exploitation tools. On the Kali installation, we install the Nessus vulnerability scanner [12] and the Metasploit framework [13] and Scapy tool [20] for exploiting vulnerable systems throughout the subnet*192.168.159.0/24* and IoT hub networks. We selected the tools because of their capability of including multiple open-source vulnerability and exploitation techniques .

The client systems were designed with intentional vulnerabilities specifically to test designing intelligent PT techniques in IIoT networks. The Windows XP OS involves vulnerabilities in SNMP and SMB protocols, the Windows 7 OS includes a vulnerability of SMBv1, and the Metasploitable2 includes several vulnerabilities related to web services, backdoors, network protocols and SQL injections. On the Ubuntu server, we install some servers containing Zimbra email, SSH, DNS, FTP, and MYSQL to demonstrate that our environment can simulate realistic IIoT networks. Each of these servers is connected to the clients with their compatible client tools. We designed an API mobile system for simulating mobile commutations, and the Argus tool [24] for capturing network flows and storing their data in MYSQL databases on the Ubuntu server. The server also includes an Arduino IDE for executing the scripts of IIoT sensors and transmit them to IoT hubs using an IIoT gateway (i.e., MQTT services).

(B) IIoT devices and sensors: The devices and sensors were deployed using the Arduino IDE on the Ubuntu server. Any industrial systems, such as energy systems, health care systems, CPSs and weather measurements, demand specific sensors. Thus, we used four sensors for temperature, humidity, pressure and red laser light. The sensors are connected via WiFi using Espressif ESP8266 [25] devices.

(C) IIoT services: Once the codes of the sensors were implemented by the Arduino IDE, the data generated from the sensors were transmitted to the Internet using the Arduino ESP8266 library [25], which allows the connections of the IoT gateway via the MQTT protocol. The protocol permits subscribing and/or publishing different IIoT topics generated from the sensors. On the Ubuntu server, we installed an MQTT client subscribed to the topics published on a public IoT broker '*broker.hivemq.com*' [7].

The testbed was configured to simulate different normal and attack events, as outlined below. At the same time, the Argus tool runs on the Ubuntu server to capture network flows and stored them on a MySQL database to test the proposed model. The network flows of the IIoT sensors deployed to the internet are stored in the database while scanning and exploiting vulnerabilities, as well as capturing normal events.

B. Intelligent Vulnerability and Exploitation Identification Model

The potential process of vulnerability and exploitation identification consists of predefined information about a target and a sequence of states to execute specific hacking actions. Vulnerability scanning tools, such as Nessus, are used to find vulnerabilities in IIoT networks based on given information, in particular the IP addresses of the targets. If vulnerabilities are discovered, exploitation tools such as Metasploit and Scapy are used to launch malicious scripts designed to gain access on the targets. If the Ubuntu server is successfully exploited, then the attacker can control the servers and platforms in the IIoT network, as they are managed by a DNS Server that maps IP addresses to hostnames.

In AI planning, the steps of exploiting vulnerabilities are called states, and the events of successful hacking targets are named actions. To formulate testing security as a planning problem, the PT planning comprises an initial state (I), a goal state (G), hacking actions (A), and hacking predictions (P) for clarifying how a sequence of states and actions could be automatically implemented. The initial state refers to the first stage of scanning vulnerabilities while the goal state denotes the goal of hacking a target (e.g., SYN DoS for exploitation). The hacking actions ($a \in A$) denote the parameters, preconditions, and effects to successfully execute the exploitation.

The hacking predictions ($p \in P$) indicate a probabilistic function $f(p) = p(a|I, G)$, which is a stochastic Markovian model that can be estimated using POMDP approaches. POMDP approaches require reward functions and termination criteria that are complex to exactly infer in real-time and take high computational resources. Intuitively, the PT planning involves a sequence of actions $A = [a_1, a_2, \ldots, a_n]$ that corresponds to states $S = [s_1, s_2, \ldots, s_n]$, where n is the number of observations. The target of the PT planning is to find the best paths that identify vulnerabilities and their successful

Authorized licensed use limited to: SLIIT - Sri Lanka Institute of Information Technology. Downloaded on February 24,2021 at 04:32:21 UTC from IEEE Xplore. Restrictions apply.

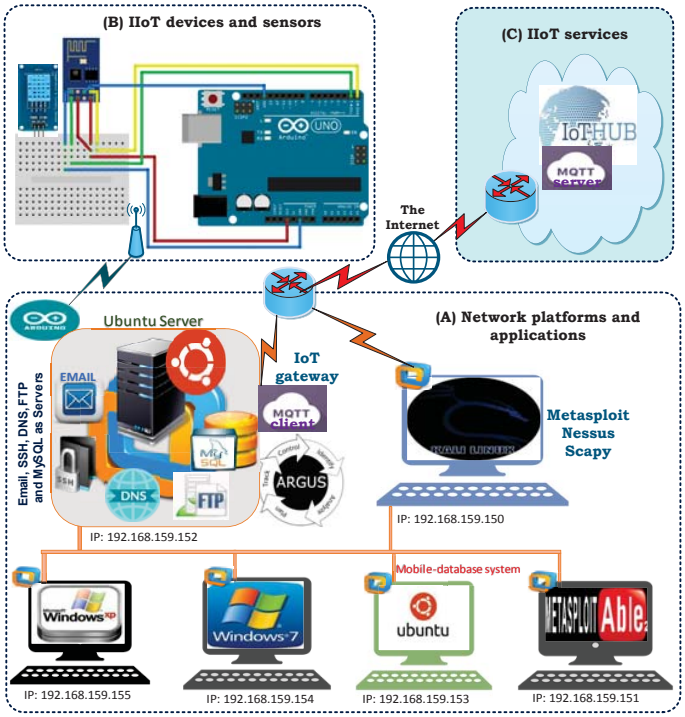

Fig. 1. Testbed configuration for IIoT networks

exploitation. This is formulated as $(I \rightarrow S_1^{a_1} \rightarrow ...S_n^{a_n} \rightarrow G)$, which can be modeled using a particle filter technique.

1) Particle Filter for automating security testing : We propose using the particle filter technique [26] to automatically find vulnerabilities and their exploitation in IIoT networks. We selected this technique because it is capable of modeling non-linear and non-Gaussian representations of network flows [10]. A particle filter model computes unknown states from the sampling particle observations $O_{1:T} = [o_1, o_2, \ldots, o_T]$ in a time series $T = [t_1, t_2, ..., t_n]$. It estimates posterior probabilities $p(A_{1:T}) = p(S_{1:T}|O_{1:T})$ using weighted particles $Z_t = \{o_t^i, w_t^i\}$, where $\sum_{i=1}^{N} w_t^i = 1$.

The dynamics of particle filter include a state transition model and an observation model, as formulated in equations (1) and (2), respectively [26], [27].

$$S_t = f_t(s_{t-1}, a_{t-1}, v_t) \tag{1}$$

$$O_t = h_t(s_t, a_t, w_t) \tag{2}$$

The state transition function (f_t) estimates the vulnerabilities and their exploitation dynamics being tracked by the previous state (s_{t-1}) with its corresponding action (a_{t-1}) and the system noise (v_t). The observation model (O_t) includes a measurement function (h_t) that estimates the relationship between the state (s_t) along with its action (a_t) and the estimated weight (w_t). The f_t is computed using a state transition probability $p(S_t|S_{t-1})$, whilst the h_t is calculated using an observation likelihood $p(O_t|S_t)$.

The posterior probabilities are computed using the Bayes' theorem to track the states and actions of vulnerabilities and exploitations by

$$p(S_t|Z_{1:t}) = \frac{p(Z_t|S_t).p(S_t|Z_{1:t-1})}{p(Z_t|Z_{1:t-1})} \tag{3}$$

From equation 3, the posterior probability $p(S_t|Z_{1:t})$ is measured by the observation probability $p(S_t|Z_{1:t-1})$ using equation (4).

$$p(S_t|Z_{1:t-1}) = \sum_{i=1}^{N} p(S_t|S_{t-1}^i).w_{t-1}^i \tag{4}$$

The following two scenarios were applied to explain how the particle filter is used for modeling vulnerabilities and their exploitation in IIoT networks.

Scenario 1- Scanning vulnerabilities: While scanning the IIoT network vulnerabilities using the Nessus tool installed on the Kali system, the Argus tool is running on the Ubuntu server to capture network flows from a destination node. It collects network traffic and stores them in a MySQL database on the server.

Scenario 2- Exploiting vulnerabilities: According to the vulnerabilities identified in Scenario 1, attacking techniques

Authorized licensed use limited to: SLIIT - Sri Lanka Institute of Information Technology. Downloaded on February 24,2021 at 04:32:21 UTC from IEEE Xplore. Restrictions apply.

are utilized to exploit the vulnerabilities. For example, Windows 7 has an SMBv1 vulnerability, the eternalblue exploit is used by the Metasploit to gain access to Windows 7. Similar to Scenario 1, the Argus tool was used to capture the network flows and store them in the MySQL database.

For the two scenarios, we added a new column called 'label' for tagging each observation in the database with the name of the vulnerability or exploitation occurred. The network flow features extracted using the Argus tool are described in Table I.

TABLE I
NETWORK FLOW FEATURES EXTRACTED FROM BOTH SCENARIOS

Feature	Description
dstip	Destination IP addresses of the targets
dport	Destination open port numbers of the targets
dtcpb	Destination TCP base sequence number
dbytes	Destination to source transaction bytes
dur	Record total duration
dpkts	Destination to source packet count
dttl	Destination to source time to live value
sload	Source bits per second
tcprtt	TCP connection setup round-trip time, the sum of 'synack' and 'ackdat'
synack	TCP connection setup time, the time between the SYN and the SYN_ACK packets
ackdat	TCP connection setup time, the time between the SYN_ACK and the ACK packets
label	A tag for each instance that represents names of vulnerabilities and exploits

The network features listed in Table 1 are used for applying the particle filter model to track the vulnerabilities and its exploitation. As shown in Figure 2, the process of the particle filter model consists of two phases: 1) capture and label IIoT network data, and 2) model vulnerabilities and their exploitation. Phase 1 describes the steps of extracting flows and label them for Scenarios 1 and 2. To build the particle filter, the probability $p(o)$ of each observation (o) in the flow features are estimated using the normal probability density function $p(o) \Rightarrow f(\mu, \delta)$. Phase 2 explains the steps of estimating the state posterior probabilities. If the probabilities of vulnerabilities and exploitations are different from normal ones, the two events could behave similarly but different from normal events. This enables in designing an automatic PT technique in IIoT networks rather than trying different exploitations by network administrators.

Since the PT automation process should couple each vulnerability with a specific exploitation, the particle filter model clarifies the track paths of posterior probabilities using the same the network flow features. Assume an IIoT network (II) has several hosts (H), where $II = [h_1, h_2, \ldots, h_H]$. Each host has some vulnerabilities $L = [l_1, l_2, \ldots, l_m]$, where m is the number of vulnerabilities identified by the scanning tools. Hacking the vulnerabilities depends on a successful exploit (e), where $e \in E = [e_1, e^2 \ldots [$ that is a set of exploits. The posterior probabilities $p(S|L)$ and $p(S|E)$ of network features ($F = [f1, f2, \ldots fd]$, where d is the number of selected features) are estimated for L and E, respectively. If

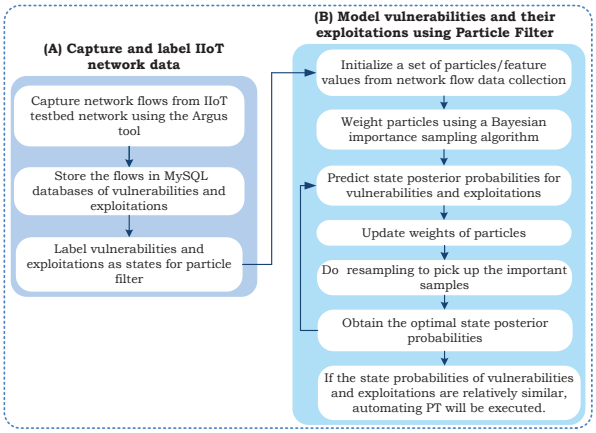

Fig. 2. Process of applying particle filter for testing security

$p(S|L) \approx p(S|E)$, then this enables automating the process of paring vulnerability and exploitation identification without human interference.

IV. EXPERIMENTS AND DISCUSSIONS

A. IIoT testbed data collection

We designed the testbed discussed above to develop an intelligent security testing technique based on network flows for IIoT networks. The particle filter technique was utilized to infer the state posterior probabilities of vulnerabilities and their exploitation, as described below.

Scanning vulnerabilities: While publishing/subscribing to the telemetry data of the IIoT sensors using the IoT hub 'broker.hivemq.com', the Nessus tool scanned the vulnerabilities of network platforms. In the same time of the scanning, the Argus tool is configured to capture the network flows listed in Table 1 and store them in a MySQL database.

Exploiting vulnerabilities: We used the Metasploit and Scapy tools to exploit the vulnerabilities discovered by the Nessus tool. Similar to the stage of discovering vulnerabilities, the Argus tool captures the same flows along with adding an exploit label. Some vulnerabilities and their exploitation are listed in Table II, and an example of demonstrating network flows with their flows are demonstrated in Table III.

TABLE II
EXAMPLES OF VULNERABILITIES AND THEIR EXPLOITATION USED IN THE TESTBED

Vulnerability	Exploitation	Host in IIoT network
MS17-010/SMBv1	Eternalblue using Metasploit	192.168.159.154
MS16-047/SAM badlock	SYN DoS using Scapy	192.168.159.154
Unix OS unsupported	SYN and ACK DoS using Scapy	192.168.159.152
Unencrypted MySQL data	Man-in-the-middle using Metasploit	192.168.159.153

Authorized licensed use limited to: SLIIT - Sri Lanka Institute of Information Technology. Downloaded on February 24,2021 at 04:32:21 UTC from IEEE Xplore. Restrictions apply.

TABLE III
SOME NETWORK FLOW SAMPLES WITH THEIR IDENTIFIED LABELS

daddr	dport	dtcpb	dbytes	dur	dpkts	dttl	sload	tcprtt	synack	ackdat	label
192.168.159.152	143	1.70E+08	597	0.0128	6	64	2682	0.001	0.004	0.016	Vulnerability
192.168.159.154	445	7.15E+08	1605	0.0281	9	128	4916	0.002	0.003	0.005	Vulnerability
192.168.159.154	445	6434789	344	0.139	5	128	277064	0.021	0.002	0.008	Exploit
192.168.159.152	80	3.58E+09	58	0.092	1	64	51835	0.003	0.001	0	Exploit
192.168.159.152	443	8485445	62	2.72	28	128	16.05	0.013	0.013	0.026	Normal
18.196.251.237	1883	2.01E+09	120	0.31	2	128	1392.15	0.312	0.312	0.004	Normal

We capture normal events that occurred while sending the IIoT data to the IoT hub, sending/replying emails using the Zimbra email server between the clients and server, and simulating the mobile communications on the mobile Ubuntu system along with storing them in a MySQL database on the Ubuntu server. The object of simulating normal events is to measure if there are statistical variations between vulnerability, exploit and normal network flows. This will help in the future to develop an integrated system that can define vulnerabilities and detect their attack vectors.

B. Vulnerability and exploitation evaluations

The vulnerability and exploitation flows found in the IIoT network are indexed using destination IP addresses (daddr) and destination ports (dports) for examining their behaviors. The particle filter is applied to the observation probabilities of the two states of vulnerabilities and exploitations for every 100-time series. The highest weighted observations are resampled with replacement to predict the important posterior probabilities of the two states.

The results reveal that the posterior probabilities of vulnerabilities vary in the same range as their counterpart of the exploits, but they are significantly different from normal probabilities. The posterior vulnerabilities and exploits of 100 observations are presented in Figure 3. On one hand, the vulnerability posteriors are greater than 0.3 until 50 observations, and then they frequently decrease and increase in a range of [0.01 – 0.3[. On the other hand, the exploitation posteriors considerably vary in the same range of the vulnerability posteriors until 38 observations, and then they are constant in the behavior.

Based on the posterior behaviors of both events, we observe that the vulnerabilities and exploitations are relatively similar with small differences in the values themselves. Consequently, building an automatic PT system is likely possible if implemented based on finding discriminative flow features. The particle filter model can filter the observations of the events, and also can allow for finding a pair of known vulnerabilities and their successful exploitation to automatically test IIoT networks.

The particle filter model is better than MDP and POMDP models, as it does not demand a reward function and termination action criteria. It also can be executed in real-time, as we did our experiments by taking sequential samples (e.g., 100 observations) as time series from the MySQL database and filter them using the model in the R programming language

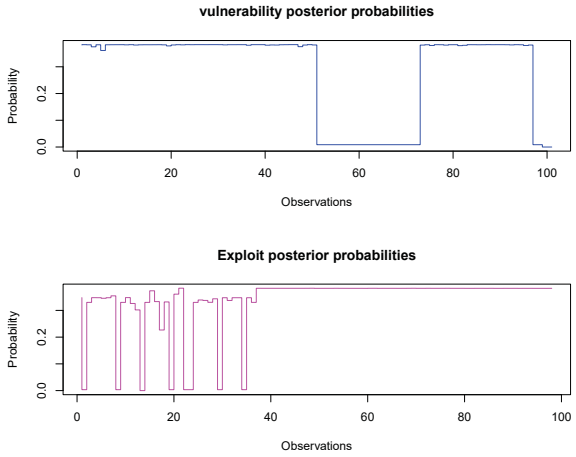

Fig. 3. Sample of venerability and exploit posterior probabilities

on the Ubuntu server. The results show that there are similar behaviors between these events that significantly different from benign events. The new testbed environment with the connections to many IIoT sensors and generate their IoT services permits to study the identification of these events in realistic scenarios.

V. CONCLUSION

This paper has proposed a new testbed environment for IIoT networks to test the applicability of establishing an intelligent penetration testing technique. The particle filter model is utilized using important network flows to discriminate between the posterior probabilities of vulnerabilities and their exploitation events. The results revealed that the behaviors of both events are relatively similar with low variations of values themselves. This outcome indicates that the particle filter technique can assist in estimating known vulnerability probabilities and pairing them with their successful exploitation to allow the automation of penetration testing.

In the future, we will extend the proposed methodology to test the pairing of vulnerabilities and their exploits with different scenarios in IIoT networks. We will expand our testbed to include a broader set of IIoT devices and systems, as well as vulnerabilities.

Authorized licensed use limited to: SLIIT - Sri Lanka Institute of Information Technology. Downloaded on February 24,2021 at 04:32:21 UTC from IEEE Xplore. Restrictions apply.

REFERENCES

[1] S. Madakam, R. Ramaswamy, and S. Tripathi, "Internet of things (iot): A literature review," *Journal of Computer and Communications*, vol. 3, no. 05, p. 164, 2015.

[2] S. Katsikeas, K. Fysarakis, A. Miaoudakis, A. Van Bemten, I. Askoxylakis, I. Papaefstathiou, and A. Plemenos, "Lightweight & secure industrial iot communications via the mq telemetry transport protocol," in *Computers and Communications (ISCC), 2017 IEEE Symposium on*. IEEE, 2017, pp. 1193–1200.

[3] B. Beurdouche, A. Delignat-Lavaud, N. Kobeissi, A. Pironti, and K. Bhargavan, "Flextls a tool for testing tls implementations," in *9th USENIX Workshop on Offensive Technologies, WOOT'15*, 2014.

[4] A. Azmoodeh, A. Dehghantanha, and K.-K. R. Choo, "Robust malware detection for internet of (battlefield) things devices using deep eigenspace learning," *IEEE Transactions on Sustainable Computing*, 2018.

[5] Q. Yan, W. Huang, X. Luo, Q. Gong, and F. R. Yu, "A multi-level ddos mitigation framework for the industrial internet of things," *IEEE Communications Magazine*, vol. 56, no. 2, pp. 30–36, 2018.

[6] "Arduino ide," May 2018. [Online]. Available: https://www.arduino.cc/en/Main/Software

[7] "Hivemq," May 2018. [Online]. Available: http://www.mqtt-dashboard.com/

[8] C.-K. Chen, Z.-K. Zhang, S.-H. Lee, and S. Shieh, "Penetration testing in the iot age," *Computer*, vol. 51, no. 4, pp. 82–85, 2018.

[9] N. Moustafa, G. Creech, and J. Slay, "Big data analytics for intrusion detection system: statistical decision-making using finite dirichlet mixture models," in *Data Analytics and Decision Support for Cybersecurity*. Springer, 2017, pp. 127–156.

[10] N. Moustafa, J. Slay, and G. Creech, "Novel geometric area analysis technique for anomaly detection using trapezoidal area estimation on large-scale networks," *IEEE Transactions on Big Data*, 2017.

[11] D. Shmaryahu, G. Shani, J. Hoffmann, and M. Steinmetz, "Partially observable contingent planning for penetration testing," in *IWAISe: First International Workshop on Artificial Intelligence in Security*, 2017, p. 33.

[12] "The nessus tool," May 2018. [Online]. Available: https://www.tenable.com/products/nessus/nessus-professional

[13] "The metasploit framework," May 2018. [Online]. Available: https://www.metasploit.com/

[14] K. Zaffarano, J. Taylor, and S. Hamilton, "Assessing effectiveness of cybersecurity technologies," Apr. 13 2017, uS Patent App. 15/286,990.

[15] J. Bozic and F. Wotawa, "Planning the attack! or how to use ai in security testing?" in *IWAISe: First International Workshop on Artificial Intelligence in Security*, 2017, p. 50.

[16] A. Leitner and R. Bloem, "Automatic testing through planning," *Technische Universität Graz, Institute for Software Technology, Tech. Rep.*, 2005.

[17] "Smbv1 of wannacrypt malware," May 2018. [Online]. Available: https://www.zdnet.com/article/how-wannacrypt-attacks/

[18] "Eternalblue metasploit codes," May 2018. [Online]. Available: https://www.rapid7.com/db/modules/exploit/windows/smb/ms17_010_eternalblue

[19] J. Fu, Y. Liu, H.-C. Chao, B. Bhargava, and Z. Zhang, "Secure data storage and searching for industrial iot by integrating fog computing and cloud computing," *IEEE Transactions on Industrial Informatics*, 2018.

[20] "The scapy tool," May 2018. [Online]. Available: https://scapy.net/

[21] A. Armando, R. Carbone, L. Compagna, K. Li, and G. Pellegrino, "Model-checking driven security testing of web-based applications," in *Software Testing, Verification, and Validation Workshops (ICSTW), 2010 Third International Conference on*. IEEE, 2010, pp. 361–370.

[22] R. E. Fikes and N. J. Nilsson, "Strips: A new approach to the application of theorem proving to problem solving," *Artificial intelligence*, vol. 2, no. 3-4, pp. 189–208, 1971.

[23] "The metasploitable2 os," May 2018. [Online]. Available: https://sourceforge.net/projects/metasploitable/files/Metasploitable2/

[24] "The argus tool," May 2018. [Online]. Available: https://qosient.com/argus/

[25] "Wifi esp8266 tools," May 2018. [Online]. Available: http://arduino-esp8266.readthedocs.io/en/latest/esp8266wifi/readme.html

[26] C.-C. Lin and M.-S. Wang, "Particle filter for depth evaluation of networking intrusion detection using coloured petri nets," in *Petri Nets Applications*. InTech, 2010.

[27] J. M. Pak, C. K. Ahn, Y. S. Shmaliy, and M. T. Lim, "Improving reliability of particle filter-based localization in wireless sensor networks via hybrid particle/fir filtering," *IEEE Transactions on Industrial Informatics*, vol. 11, no. 5, pp. 1089–1098, 2015.

Authorized licensed use limited to: SLIIT - Sri Lanka Institute of Information Technology. Downloaded on February 24,2021 at 04:32:21 UTC from IEEE Xplore. Restrictions apply.