IoTurva: Securing Device-to-Device (D2D) Communication in IoT Networks

Ibbad Hafeez
University of Helsinki
Helsinki, Finland
Email: ibbad.hafeez@helsinki.fi

Aaron Yi Ding
Technical University of Munich
Munich, Germany
Email: aaron.ding@tum.de

Sasu Tarkoma
University of Helsinki
Helsinki, Finland
Email: sasu.tarkoma@helsinki.fi

***Abstract*—We present IoTurva, a platform dedicated to securing Device-to-Device (D2D) communication in IoT networks. IoTurva targets an alarming and yet unexplored region in IoT security, where the interactions and dependencies across heterogeneous IoT devices are extremely hard to secure and regulate. Different from existing proposals that mostly tackle conventional device-to-infrastructure communication for IoT, our solution embeds the security functions within the connectivity for IoT devices and can provide selective network isolation at device-level granularity. To demonstrate IoTurva, we have implemented a prototype and evaluated it through several live scenarios in the testbed. Our results show that IoTurva is responsive and can effectively control cross-device dependencies and D2D interactions in IoT networks.**

I. Introduction

The proliferation of Internet-of-Things (IoT) is quickly turning the hype into an imminent reality. Recent forecasts expect more than 20 billion devices connected to the Internet by the end of this decade [1]. Residential and corporate networks provide connectivity to billions of IP-enabled devices. IoT brings automation to industrial and smarthome lighting, security etc. with a promise to improve life style of users.

This huge potential of IoT devices comes at a cost of user security and privacy. Most of IoT devices are developed by fast moving teams in large enterprises or independent startups. The manufacturers have constrained resources and tight deadlines for developing products and launching them in the market. Tight deadlines and limited resources force these manufacturers to do corner cutting such as using unverified code snippets and not following security by design principles [2]. Manufacturers tend to embrace insecure design practices e.g., allowing weak passwords for customer's ease of use.

Consumers and manufacturers do not pay attention to security and privacy attacks (using IoT devices) reported frequently, as we see similar kind of issues (re)appearing over time. Although IoT devices installed in edge networks are not easily accessible via Internet due to NAT-ing, it does not make these devices itself secure. Attackers have been able to hack IoT devices installed deep in edge networks [3], [4] for DDoS, spam, malware attacks, which made IoT networks a challenging environment in terms of security and privacy.

Traditionally, security is either provided by network intrusion detection systems (NIDS) e.g., firewalls, intrusion detection/prevention systems, installed on gateways or vantage points in the network or end host based security applications e.g., anti-virus or anti-malware. Additionally, software vendors provide updates and security patches to mitigate any security vulnerabilities discovered in their products. However, all these solutions fall short to address most of the security and privacy concerns [5], [6], related to IoT devices, for a number of reasons.

Complex Device-to-Device (D2D) interactions: Typical IoT installations consist of a number of devices specialized for monitoring environment or performing dedicated task (e.g., temperature sensors, smart bulb). IoT ecosystem allows these devices to intercommunicate and follow IF-This-Then-That (IFTTT) model [7] to perform various functions. For instance, if temperature is $\leq 90°C$, switch on heating. This interdependency between IoT device can be exploited by attackers to gain access over the whole ecosystem. For example, an attacker can generate a fake signal from CCTV camera to open the garage door giving him access to user home. Many IoT devices (e.g., smart locks, CCTV cameras, perimeter security sensors) are installed outside home, exposing them to illegal tampering. Therefore, it is very important to curtail these D2D interactions and dependencies in IoT ecosystem to reduce the possibilities for an attacker to trick the system into helping adversaries.

NIDS use a number of security policies or firewall rules to match state of any traffic flow in/out of the network [8]. Using $Match \rightarrow Action$ technique, they `allow/block` given traffic flow. However, direct D2D and out-of-band interactions (using state changes) are difficult to capture by traditional rule based security models. Hence, we require a solution which is able to take context and D2D dependencies into account for securing network communications.

In addition, NIDS follow *reactive* security model where a threat needs to be first detected and then blocked. Threat detection requires substantial manual efforts (analyzing logs, setting up alarms for attack detection) and also requires manual update of security configurations to block the threats. A number of smart home solutions use D2D dependencies to offer interactive features for users such that switching on lights when main door is unlocked. Such solutions make it complex to document and differentiate between normal D2D interactions and anomalous actions.

Heterogeneity of ecosystem: IoT ecosystems can consist of thousands of devices from hundreds of manufacturers. These devices run a very stripped down version of operating system (OS) or none at all. The diversity of firmware models, software application stack, lack of power and hardware resources, lack of software updates and security patches etc. make it nearly impossible to develop end-host based software security applications for IoT devices. Longevity of IoT installations also makes it impractical to leave unpatched, vulnerable IoT devices in the wild.

Lack of growth truth: NIDS and anti-viruses use huge databases of malware and attack signatures to identify anomalous behaviour from network traffic traces and device activities. The diversity of firmwares, application stacks and protocols used by IoT devices makes it challenging to develop such signature databases for IoT usecases. Traditional approaches for learning attack signatures e.g., honeypots also do not scale for IoT scenarios [5]. The information about D2D dependencies is vital for developing a practical, scalable security solution for IoT ecosystem. It is difficult to develop static configuration models (policies) for IoT networks given the complexity of D2D interactions and sheer number of devices. These configurations also need to be constantly and repeatedly updated as new devices join/leave the network.

Dynamic and automated enforcement mechanism: Typical residential and corporate networks provide connectivity to user's smart devices including smartphones, tablets, computers and IoT devices. Traditionally, network managers are responsible for setting up traffic filtering rules required to secure incoming/outgoing traffic. This approach does not scale with the huge number of networks as well as the heterogeneity of devices connected to these networks. In order to ensure consistent security for all connected devices all the time, networks should be able to identify connected devices. Using device-related information, gateways and access points (APs) should be able to automatically retrieve and enforce required set of traffic filtering rules to secure all connected devices e.g. if user connects smart TV to home network, gateway should be able to detect it and reconfigure network to ensure that smart TV is not able to send video and audio stream from microphone and webcam to an arbitrary server at any time.

Our work tackles these challenges and the contributions are:

- We present IoTurva, a security platform that combines contextual information with network data to enable automatic classification of network flows in network edge and provide security as integrated part of connectivity for all user devices.
- We implemented a prototype of IoTurva to demonstrate its efficacy, performance for resolving security challenges in edge networks.
- We analyze the challenges and identify the avenues of growth in developing lightweight context aware security enforcement mechanism for IoT ecosystem.

II. System Overview

Figure 1 shows the system design of IoTurva, primarily consisting of Turva Gateway and Turva Service. Turva Gateway is responsible for supervising all communications among IoT and other devices residing in edge network. Turva Gateway uses *Software-defined Networking* (SDN) to manage network communications and enforce traffic filtering and security policies in the network. The system supports plug and play architecture for security services, which are used to identify network attacks and anomalous device behavior by analyzing user traffic. Turva Service can either be centralized or it can be deployed as a set of disjoint interconnected installations, where each installation is responsible for one or more network functions such as traffic filtering, malware analaysis etc., provided by Turva Service.

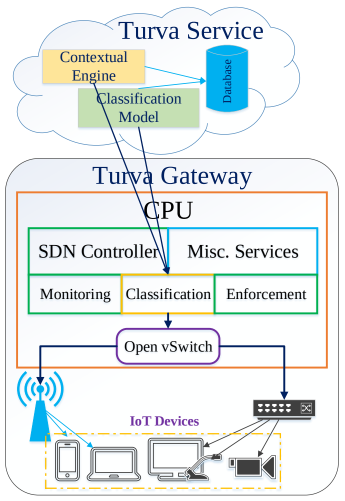


Fig. 1. System design for IoTurva.

A. Turva Gateway

We deploy Turva Gateway at the network edge for providing connectivity to all user devices including IoT devices. It is responsible for monitoring all traffic within and across the network to identify and block anomalous network interactions. It is a lightweight gateway which can be deployed using minimum hardware resources e.g., Raspberry PI or legacy APs, as it offloads resource intensive tasks e.g., classification model training, state management, context management etc. to the Turva Service.

Turva Gateway can oversee every network interaction, may it be network-local (both devices connected to same user network/AP) or remote (user device connecting to remote destination over the Internet or vice versa), and generate a signature from it. It combines context information (retrieved from user devices and Turva Service) with this signature. The final signature is then classified using a classification model as either *normal* or *abnormal* interaction. Based on the result, Turva Gateway generates a rule and installs it in the network to allow or prohibit similar network interactions. This practice allows us to provide *proactive* security for protecting user networks and device compared with *reactive* security provided by traditional network perimeter security systems.

Turva Gateway also acts as a sensor for aggregating traffic signatures (i.e. network data) from user networks and passing it to Turva Service where it is used to train and improve those classification models.

B. Turva Service

Turva Service is logically a centralized platform for supporting Turva Gateway in managing the device and network security. Turva Service platform allows to deploy *software-defined middleboxes* (SdMs) as well as *micro security services* µSS to perform analysis on user's network data [9]. It collects and maintains auxiliary information about the context of users and their IoT devices. This context information includes user location, device location, activity hours, normal usage patterns, device network activity etc.

Turva Service develops a global view across a number of networks using all collected information. This information combined with data from 3rd party sources results in generating improved traffic classification and network anomaly detection models with high generalization. These models are able to classify D2D interactions with higher accuracy.

IoTurva architecture is flexible to support different deployment models including centralized, distributed and hybrid model. We envision incremental deployment model to follow a *brownfield* approach where legacy setups can be used to setup our system. Therefore, we require minimal hardware resources for Turva Gateway, so that it can be deployed using typical APs available in traditional network or a small form factor PC e.g., Raspberry PI [1], Omega Onion [2]. Turva Service can also be deployed on premises by an enterprise otherwise it can be maintained by a service provider and subscribed by users, who configure their Turva Gateway to use the service for managing their networks.

C. Policy Engine

Traffic filtering rules in NIDS lack the expressiveness to address D2D interactions in IoT ecosystem. NIDS uses $State, Match \rightarrow Action$ model for traffic filtering where $State$ and $Match$ is retrieved from packet headers, connection state and $Action$ corresponds to whether given flow will be processed for delivery or dropped. It assumes that all devices are independent and each device's activity only changes its own state. However, the case is different in IoT scenario e.g. if CCTV detects a car pulling into the driveway, garage door will be opened. In order to use NIDS, it is impractical to specify all possible D2D dependencies to generate these rules. Due to a large number of IoT devices and diversity of their mutual interactions, any efforts of manually specifying D2D dependencies will result in inconsistent specifications that break down device functionalities and generating loopholes to be exploited by attackers. It will also expose users to far more attacks by giving them a false sense of security.

IoTurva uses *security profiles* (D_{sp}) for each user device. D_{sp} is estimated based on the set of known vulnerabilities associated to the device and device's (anomalous) network activity. Different ways to get this ground truth are mentioned in Section II-D. In essence, D_{sp} is a numerical value, referred to as *secure*, *suspicious* and *unsafe* for the sake of clarity.

When a device is first connected to the network, Turva Gateway identifies the device [10] and assigns D_{sp}. It is later updated based on device's previous and current network activity e.g., the profile of *safe* CCTV camera will be updated to *suspicious* if it tries to scan the network. IoTurva registers all users devices with Turva Service to support mobility, state and context management. It also allows users to specify preferences at device and context level granularity e.g., use parental control on tablet PC whenever it is at home.

The classification engine uses D_{sp}, network metadata and context information to assign a *suspicion index* (SI) for every connection request analyzed by Turva Gateway. Based on SI, Turva Service either directs Turva Gateway to immediately deny the traffic flow request, reroute traffic via µSS /SdMs or allow the traffic flow to be established e.g. if CCTV tries request to smartlock, requesting to open the garage door when $D_{sp}(CCTV) : suspiciouys, time : night, userlocation : home$ then $SI(request) = suspicious$ i.e., $SI(request) : 0.6$ (where $SI(i) \in [0-1]$) and request to open garage door will be *blocked* to minimize risks of security compromise for the premises. In case of a highly suspicious activity, IoTurva can notify the user via email, sms or phone call. It can generate monthly, weekly, daily reports depending on network activity, for the users

The efficiency of this scheme is dependent on the hardware resources available on Turva Gateway. The system also supports service mobility for the devices across all networks secured by IoTurva. In order to deal with *brute force scenario*, which results in poor performance as the number of devices, states and environment variable grow [5], we decouple training and classification parts. The training is done utilizing large resources available in Turva Service, to achieve high generalization. Classification is done in edge networks to support user privacy and prevent sharing of user's network data with external services.

D. Ground truth collection

Ground truth collection is of key importance for for training classification model and assigning D_{sp}. We can use malware signatures and honeypot-like mechanisms for learning attack signatures related to IoT devices. However, such technique does not scale to IoT scenarios where there is huge diversity in device interaction models, composition of ecosystem and services. Some potential sources for collecting up-to-date network attack and anomaly signatures for our system are outlined as follows.

Crowdsourcing: Crowdsourcing will allow users with IoT deployments to share their normal and anomalous device activity signatures. Turva Service allows subscribers as well as non-subscribers to publish their signatures with the system. The information submitted from subscribers can be used for providing subscriber-oriented security and mobility services.

[1]https://www.raspberrypi.org/products/raspberry-pi-2-model-b/
[2]https://onion.io/

Non-subscribers can get incentives for sharing this information with the system in form of monetary benefits or free access to signature database maintained by Turva Service.

However, crowdsourcing raises issues about privacy and quality of collected information. We can use privacy preserving techniques and anonymized submissions to ensure privacy of users who submitted the information [11], [6]. Data quality can be improved by cross examining the collected information by experts, using threshold-based acceptance of submitted signatures. Many other techniques including user reputation, voting etc. have been proposed in literature, to improved quality of crowdsourced data [12].

CVE, malware databases: Public CVE [13] and malware databases [14] publish information about vulnerabilities detected for IoT devices. We can also create attack signatures against common exploits, as reported in CWE [15], in software stack used by IoT devices. This information can be used by IoTurva to develop up-to-date ground truth for its classification models and configurations updates of µSS .

Device manuals and cloud services: Many IoT vendors provide information about device functionality and supported interactions with other IoT devices, in companion manuals. They also list of cloud services and devices from other vendors compatible with their devices along with how to enable cross device interactions in user setup. This information can be useful in covering cross device dependencies in IoTurva.

Testbeds: The testbeds developed by researchers working with IoT security and communications can provide in-depth data about device vulnerability and cross-device dependencies. These testbeds can also be used to individually monitor device behaviors and mode of interactions in several states. IoTurva uses such data to model device behaviors and identify (signatures) abnormal device interactions. A combination of these models can also be used to capture multi-level attacks using cross-device interactions.

Our proposed Turva Gateway can be regarded as a *sensor* in IoT networks. It can monitor network traffic and provide aggregated information to Turva Service. This information can be used to deduce device usage patterns, baseline device activity and device interaction profiles, which are then used to train classification models for detecting anomalous activities in the network.

III. Implementation and Evaluation

Our prototype implementation uses a Raspberry PI II model B [16] for deploying Turva Gateway. We have written an additional module in `Floodlight` controller v1.2 [17] to intercept every new connection attempt, classify it as a normal or abnormal, generate and deploy OF rules to enforce network policies to allow or block the connection request. User's location, network activity and other context information are obtained through a smartphone application. The interfaces offered by IoT devices can also be used to get context information e.g., CCTV camera interfaces can be accessed to verify whether there is a person/object in the field of view or not.

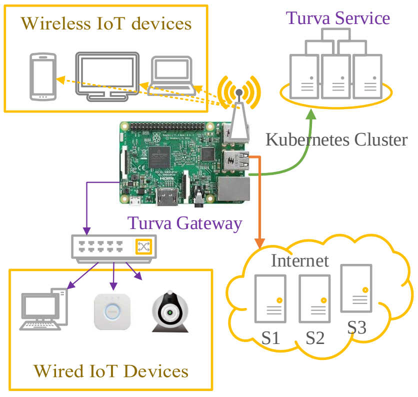


Fig. 2. Evaluation testbed layout.

We use `Kubernetes` [18] cluster setup in our lab to deploy Turva Service, as it provides builtin load balancing, service discovery etc. We deploy µSS using lightweight `Docker` containers [19]. Currently, we use Linux VMs for hosting µSS but lightweight `Click OS` [20] or other Unikernel OS [21] can be used to host these services.

We use D_{sp} to setup adhoc network overlays in edge networks, see Fig 3, to group together devices with *similar* D_{sp} i.e. for *safe* overlay contains $D_{sp}(device) = safe \ \forall \ devices$. These overlays allow us to specify different level of network access for individual devices by restricting the set of destinations any device can communicate with. IoTurva supports *selective network isolation* at device level granularity e.g. user's own smartphone can control smarthome lighting system, however, any guest's smartphone can not connect to any IoT devices in user smarthome. Fig 3 shows completely isolated (compromised) devices, who cannot talk to any other device in the network.

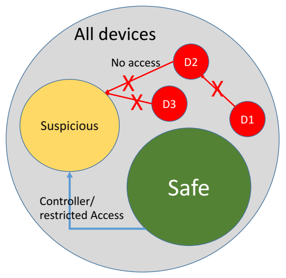
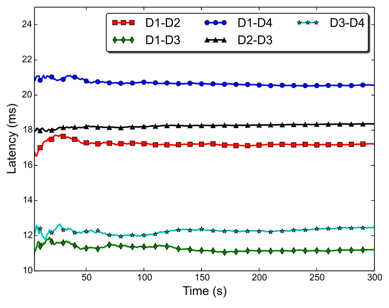


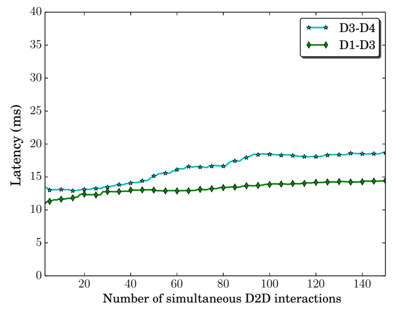
Fig. 3. Overlay networks developed for limiting cross-device interactions.

As shown in Fig. III, we used $\geq$ 10 devices in various testing scenarios to simulate a typical smart home IoT ecosystem where all devices are connected to same network and can communicate with each other openly. During evaluation, each of the simulations was performed for 300 seconds and results presented here are aggregated over 30 simulation runs.

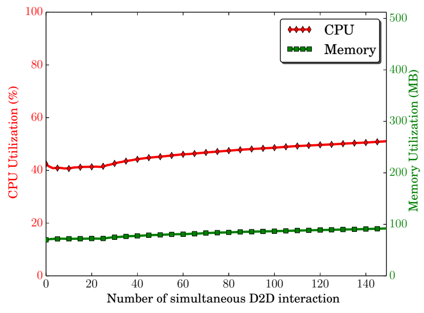
Table I shows that we can achieve similar latency with IoTurva as experienced in traditional network setups. Table II shows that throughput achieved by IoTurva is also comparable to that of traditional networks (where $S1 : localserver$, $S2 : iperf.funet.fi$, $S3 : iperf.scottlinux.com$, $S4 : bouygues.testdebit.info$). Figure 4 shows that IoTurva does not significantly affect user experience in terms of latency



(a)



(b)



(c)

Fig. 4. **IoTurva performance in a typical smart home IoT ecosystem.** (a) Latency experienced for D2D communication via Turva Gateway. (b) Increase in latency due to simultaneous D2D interactions in the network. (c) CPU and memory utilization for Turva Gateway functioning.

TABLE I
LATENCY (MS) EXPERIENCED FOR CROSS-DEVICE COMMUNICATIONS.

Latency	**D5**	**D6**	**D7**	**D8**
D1	25.3 ($\pm$2.1)	18.3 ($\pm$1.1)	28.2 ($\pm$3.3)	14.9 ($\pm$0.6)
D2	27.5 ($\pm$1.9)	17.2 ($\pm$1.3)	22.1 ($\pm$3.1)	13.7 ($\pm$0.8)
D3	27.3 ($\pm$2.5)	15.5 ($\pm$1.4)	25.9 ($\pm$3.4)	14.1 ($\pm$0.8)
D4	26.6 ($\pm$2.1)	16.8 ($\pm$1.6)	21.4 ($\pm$5.7)	13.6 ($\pm$0.9)

TABLE II
THROUGHPUT (MBPS) ACHIEVED BY USING IOTURVA.

Server	**D1**	**D2**	**D3**	**D4**
S1	32.1 ($\pm$0.2)	33.4 ($\pm$0.1)	36 ($\pm$0.1)	48.9 ($\pm$0.0)
S2	14.4 ($\pm$1.2)	14.4 ($\pm$0.9)	15.3 ($\pm$1.3)	32.2 ($\pm$0.1)
S3	7.6 ($\pm$9.8)	7.9 ($\pm$12.1)	7.5 ($\pm$11.1)	14 ($\pm$6.3)
S4	14.7 ($\pm$7.7)	15.8 ($\pm$4.5)	13.6 ($\pm$6.2)	30.9 ($\pm$2.1)

and overhead.

Figure 4a shows the latency experienced for D2D communications within IoTurva testbed network. Figure 4b shows that latency experienced in D2D communications is not significantly affected as the number of background traffic increases with simultaneous D2D interactions where each interaction corresponds to one traffic flow. Figure 4c show that memory and CPU utilization increases sub linearly with an increase in number of simultaneous flows in the network. Figure 4 shows that low memory and CPU footprint of Turva Gateway makes it easy to deploy using lightweight hardware to improve cost efficiency.

Figure 5a shows a direct D2D scenario where an attacker compromises an IP-connected CCTV camera and tricks smart lock to open garage door and get access to user's home. In normal setup, when CCTV sees a car pulling up in drive way, it will request smart lock system to open garage door. Attacker can therefore, use compromised CCTV to request smart lock to open garage door anytime.

In test setup, we assume CCTV is physically accessible to anyone, so $D_{sp}(CCTV) = suspicious$, whereas $D_{sp}(smartlock) = safe$ as it can only be configured using user's smartphone. In attack scenario, when adversary makes a request to smartlock setup (using CCTV) to open garage door at night, Turva Gateway classifies it, using contextual information about user location (obtained from Turva Service), local time and connection information, as suspicious $SI = 0.78$ and the connection is blocked.

Our current system uses a fuzzy inference logic[3] for classification purposes. The model is trained using Fuzzy C-Mean clustering technique to cluster 1000 training samples from each class into 20 clusters. The set of features (antecedent variables) include D_{sp}, connection parameters, local time, user location, device location and consequent variable gives SI.

We used 11 devices with 10 interactions between each pair of devices to get a total of $10 \times 10 \times 10$ test scenarios. We have been able to achieve $\geq 93\%$ success in accurately classifying direct D2D interactions as normal or abnormal. However, we argue that the number of parameters (considered in our test setup) can be further increased for accurately predicting SI and minimizing false positive predictions. Our high success rate is also owed to availability of complete ground truth for testbed setup, which is not the case for real world setups. Therefore, we suggest using machine learning based algorithms to train traffic classification models using a large number of connection and contextual parameters for covering a wide range of device interaction scenarios.

Figure 5 shows one of the test scenarios for multi-level D2D interaction. In normal setup, when car pulls into drive way, CCTV ($D_{sp}(CCTV) = suspicious$) requests lighting system ($D_{sp}(lightSrv) = safe$) to switch on lights and lighting system requests smart lock ($D_{sp}(smartlock) = safe$) service to open the main door. An adversary (as CCTV) can request lighting system to switch on lights and Turva Gateway with allow this connection. However, when lighting system requests smart lock to open main door, Turva Gateway will take context of previous (CCTV to lighting system) and current request combined with other information into account and block the request to open main entrance door.

We divided test devices into three classes for evaluating multi-level cross-device dependencies, where $D_{sp}(classA) = suspicious$, $D_{sp}(classB) = safe$ and $D_{sp}(classC) = safe$. Each $D_i \in ClassA$ (e.g. CCTV) requests $D_j \in ClassC$ (e.g. lighting system) to perform an action, which will lead to $D_k \in$

[3]Details omitted due to page limit

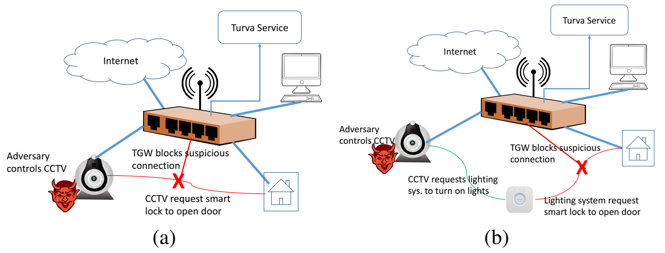

Turva Service
Internet
Adversary controls CCTV
TGW blocks suspicious connection
CCTV request smart lock to open door
(a)
Turva Service
Internet
Adversary controls CCTV
TGW blocks suspicious connection
CCTV requests lighting sys. to turn on lights
Lighting system request smart lock to open door
(b)

Fig. 5. **Test scenarios for securing cross-device dependencies.** (a) Direct D2D dependency scenario. (b) Multi-level D2D dependency scenario.

TABLE III
CONFIDENCE MATRIX FOR SUCCESSFULLY IDENTIFYING MULTI-LEVEL D2D INTERACTIONS.

		Class B						
		D6	**D7**	**D8**	**D9**	**D10**		
Class A	**D1**	1	1	0.9	1	1	**D11**	Class C
	D2	1	0.8	0.7	0.5	0.4	**D12**	
	D3	1	0.8	1	1	0.9	**D13**	
	D4	1	1	1	0.9	0.9	**D14**	
	D5	1	1	1	0.9	1	**D15**	

$ClassB$ (e.g. smart lock) to execute desired action (e.g. open main door). For each tuple (D_i, D_j, D_k) e.g., (D_1, D_{11}, D_6), 10 interactions were considered for classifications. Table III shows the number of interactions accurately classified by our system. Once again, availability of complete ground truth allowed us to achieve 90.8% accuracy in classifying normal/abnormal interactions in multistage D2D dependencies. However, this time more cases are misclassified(compared to direct D2D interactions) because of less feature available for classification.

IV. CONCLUSION

The emerging number of security and privacy incidents on IoT are becoming taxing to our everyday life. Our networks are littered with these vulnerable devices putting user privacy and data security at stake. Our analysis reveals that traditional security mechanisms fail to address the concerns related to IoT devices. There is a dire need to develop IoT focused network security solutions because IoT devices are the weak links, which allow adversaries to infiltrate the network and put the whole network security in jeopardy. Therefore, we proposed IoTURVA, to enable context aware security in IoT ecosystem to address D2D dependencies. Our system takes a incrementally deployable approach to solve major network challenges related to IoT. Although the system is not a silver bullet solution to address all security and privacy concerns related to IoT, we expect this work to serve as stepping stone towards building solid IoT specific security and privacy solutions. We have identified the limitations in our work and proposed possible avenues of improvement for future research.

REFERENCES

[1] "Gartner Says Hyperconverged Integrated Systems Will Be Mainstream in Five Years," http://www.gartner.com/newsroom/id/3308017, 2017, [Online; accessed 04-June-2017].
[2] Senrio, "400,000 Publicly Available IoT Devices Vulnerable to Single Flaw," http://blog.senr.io/blog/400000-publicly-available-iot-devices-vulnerable-to-single-flaw, 2017, [Online; accessed 04-June-2017].
[3] N. Woolf, "DDoS attack that disrupted internet was largest of its kind in history, experts say," https://www.theguardian.com/technology/2016/oct/26/ddos-attack-dyn-mirai-botnet, 2017, [Online; accessed 04-June-2017].
[4] P. Paganini, "OVH hosting hit by 1Tbps DDoS attack, the largest one ever seen," http://securityaffairs.co/wordpress/51640/cyber-crime/tbps-ddos-attack.html, 2017, [Online; accessed 04-June-2017].
[5] T. Yu, V. Sekar, S. Seshan, Y. Agarwal, and C. Xu, "Handling a trillion (unfixable) flaws on a billion devices: Rethinking network security for the internet-of-things," in *Proceedings of the 14th ACM Workshop on Hot Topics in Networks*, ser. HotNets-XIV. New York, NY, USA: ACM, 2015, pp. 5:1–5:7. [Online]. Available: http://doi.acm.org/10.1145/2834050.2834095
[6] H. Haddadi, H. Howard, A. Chaudhry, J. Crowcroft, A. Madhavapeddy, and R. Mortier, "Personal data: Thinking inside the box," *CoRR*, vol. abs/1501.04737, 2015. [Online]. Available: http://arxiv.org/abs/1501.04737
[7] "IFTTT Recipies," https://ifttt.com/recipes, 2016, [Online; accessed 29-August-2016].
[8] N. Feamster, "Outsourcing Home Network Security," in *Proceedings of the 2010 ACM SIGCOMM Workshop on Home Networks*, ser. HomeNets '10. New York, NY, USA: ACM, 2010, pp. 37–42. [Online]. Available: http://doi.acm.org/10.1145/1851307.1851317
[9] J. Sherry, S. Hasan, C. Scott, A. Krishnamurthy, S. Ratnasamy, and V. Sekar, "Making Middleboxes Someone else's Problem: Network Processing As a Cloud Service," in *Proceedings of the ACM SIGCOMM 2012 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, ser. SIGCOMM '12. New York, NY, USA: ACM, 2012, pp. 13–24. [Online]. Available: http://doi.acm.org/10.1145/2342356.2342359
[10] M. Miettinen, S. Marchal, I. Hafeez, N. Asokan, A. Sadeghi, and S. Tarkoma, "Iot sentinel: Automated device-type identification for security enforcement in iot," *CoRR*, vol. abs/1611.04880, 2016. [Online]. Available: http://arxiv.org/abs/1611.04880
[11] Y.-A. de Montjoye, E. Shmueli, S. S. Wang, and A. S. Pentland, "openpds: Protecting the privacy of metadata through safeanswers," *PLoS ONE*, vol. 9, no. 7, p. e98790, 07 2014. [Online]. Available: http://dx.doi.org/10.1371%2Fjournal.pone.0098790
[12] H. Kajino, H. Arai, and H. Kashima, "Preserving worker privacy in crowdsourcing," *Data Mining and Knowledge Discovery*, vol. 28, no. 5, pp. 1314–1335, 2014. [Online]. Available: http://dx.doi.org/10.1007/s10618-014-0352-3
[13] "Common Vulnerabilities and Exposures," https://cve.mitre.org/, 2017, [Online; accessed 04-June-2017].
[14] L. Zelster, "Malware Sample Sources for Researchers," https://zeltser.com/malware-sample-sources/, 2017, [Online; accessed 04-June-2017].
[15] "Common Weakness Enumeration," https://cwe.mitre.org/, 2017, [Online; accessed 04-June-2017].
[16] "Raspberry PI 2 Model B," https://www.raspberrypi.org/products/raspberry-pi-2-model-b/, 2017, [Online; accessed 04-June-2017].
[17] B. S. Networks, "Project Floodlight: Floodlight Open SDN Controller," http://www.projectfloodlight.org/floodlight/, 2016, [Online; accessed 1-August-2016].
[18] E. A. Brewer, "Kubernetes and the path to cloud native," in *Proceedings of the Sixth ACM Symposium on Cloud Computing*, ser. SoCC '15. New York, NY, USA: ACM, 2015, pp. 167–167. [Online]. Available: http://doi.acm.org/10.1145/2806777.2809955
[19] "Docker," https://www.docker.com/, 2016, [Online; accessed 29-August-2016].
[20] J. Martins, M. Ahmed, C. Raiciu, V. Olteanu, M. Honda, R. Bifulco, and F. Huici, "Clickos and the art of network function virtualization," in *Proceedings of the 11th USENIX Conference on Networked Systems Design and Implementation*, ser. NSDI'14. Berkeley, CA, USA: USENIX Association, 2014, pp. 459–473. [Online]. Available: http://dl.acm.org/citation.cfm?id=2616448.2616491
[21] S. Raza, L. Wallgren, and T. Voigt, "Svelte: Real-time intrusion detection in the internet of things," *Ad Hoc Netw.*, vol. 11, no. 8, pp. 2661–2674, Nov. 2013. [Online]. Available: http://dx.doi.org/10.1016/j.adhoc.2013.04.014