Secure IoT Data Analytics in Cloud via Intel SGX

Md Shihabul Islam
Department of Computer Science
The University of Texas at Dallas
Richardson, TX 75080, U.S.A.
md.shihabul.islam@utdallas.edu

Mustafa Safa Ozdayi
Department of Computer Science
The University of Texas at Dallas
Richardson, TX 75080, U.S.A.
mustafa.ozdayi@utdallas.edu

Latifur Khan
Department of Computer Science
The University of Texas at Dallas
Richardson, TX 75080, U.S.A.
lkhan@utdallas.edu

Murat Kantarcioglu
Department of Computer Science
The University of Texas at Dallas
Richardson, TX 75080, U.S.A.
muratk@utdallas.edu

***Abstract*—The growing adoption of IoT devices in our daily life is engendering a data deluge, mostly private information that needs careful maintenance and secure storage system to ensure data integrity and protection. Also, the prodigious IoT ecosystem has provided users with opportunities to automate systems by interconnecting their devices and other services with rule-based programs. The cloud services that are used to store and process sensitive IoT data turn out to be vulnerable to outside threats. Hence, sensitive IoT data and rule-based programs need to be protected against cyberattacks. To address this important challenge, in this paper, we propose a framework to maintain confidentiality and integrity of IoT data and rule-based program execution. We design the framework to preserve data privacy utilizing Trusted Execution Environment (TEE) such as Intel SGX, and end-to-end data encryption mechanism. We evaluate the framework by executing rule-based programs in the SGX securely with both simulated and real IoT device data.**

***Keywords*-IoT; Intel SGX; Data privacy; Rule-based IoT Platform;**

I. Introduction

The Internet of Things (IoT) has transformed the way we live and work with their ubiquitousness, inexpensiveness, and convenience of usage. Increasingly, IoT devices are found in our day-to-day life, such as in smart homes, industrial automation, agriculture, smart transportation, healthcare etc. They have become a fundamental part of the modern society and still offer plenty of opportunities to make our life more comfortable and constructive. The recent growth of IoT is astonishing and it is predicted that there will be 64 billion IoT devices by 2025 [1]. Although, IoT systems have many benefits, there are also plethora of security and privacy concerns related to IoT. IoT deals with vast amounts of highly vulnerable and sensitive data, which needs careful maintenance and secure storage and processing system to ensure user privacy and data protection. For instance, while listening for commands, Samsung's Smart TV captures every words of its users no matter how private the conversation and transmits to a third party for conversion of speech to text [2].

Recent development of cloud computing has provided the opportunity to use cloud-based services to collect, process, analyze, and mine large amounts of data [3]–[11], which is both cost effective and less time consuming [12]. A recent study found that, out of 81 common IoT consumer devices, 72 send data to third parties, and rest to the original device manufacturer [13], [14]. Although, the cloud service providers ensure that data is always protected at rest, they are vulnerable to many security threats during transmission, and computation [15]–[18]; e.g., data breaches, especially in the public cloud services [19]. For instance, CloudPets, which manufactures smart stuffed toys for children, stored all the data (i.e., email, password, photos, voice recordings) in the unsafe cloud, exposing over $820,000$ user accounts including 2.2 million voice recordings [20]. In addition, adversaries may physically access the machines or obtain root privileges of the machines deployed at the service providers' premises and thus steal sensitive information with ease [21].

Moreover, the availability of cheap yet powerful IoT devices has paved the way for platforms to enable information passing among IoT devices and online services to automate different processes. These platforms, such as Samsung's SmartThings [1] and IFTTT (If-This-Then-That) [2], offer users to automate their smart home or industrial system through customized policy-based rules that control the interactions between devices. For example, to conserve energy and reduce cost, a user may program a rule that automatically turns off the air conditioner and the light bulbs when the user is away from home. However, this enlarges potential attack surface and privacy risks, since

[1]https://www.smartthings.com
[2]https://ifttt.com

these automation policies and sensitive device information are shared with untrusted parties over the internet. For instance, suppose we have a temperature sensor which can open windows in a room. A temperature-related application can periodically check the room temperature and if the temperature is above a predefined threshold, then the sensor will open the window. Now, if an attacker can get access to the logic code in the cloud, he/she can change the value of the threshold, which could trigger the window opening action and cause a potential problem of break-in. Therefore, conventional security mechanisms of the cloud services need to be enhanced to thwart adversaries from stealing sensitive data and information.

In this paper, we present a system that is established based on our previously proposed framework [22]. More specifically, in our previous work, we envisioned a system to securely store and process IoT information in a privacy-preserving manner by utilizing proper cryptography techniques and the Trusted Execution Environments (TEEs). In this work, we develop and empirically evaluate the envisioned framework to ensure the integrity and confidentiality of sensitive IoT data, private user information, and vulnerable automation policies in the untrusted cloud by performing rule-based analytics on a popular TEE called Intel Software Guard Extensions (SGX) [23]. Intel SGX creates an isolated secure memory container, where the code and data can be safely stored and executed. No adversaries, not even higher privileged software such as operating system (OS) or virtual machine manager (VMM) can access the contents of SGX. Therefore, our framework stores delicate IoT data, and user information in encrypted format, and securely executes rule-based interactions of IoT devices in the enclave, so that adversaries cannot manipulate or steal information. Moreover, we ensure data security in transit from IoT devices to cloud service provider with SGX by following strong end-to-end encryption mechanism. That means, in transit data is always kept in encrypted form, except when it is in the SGX. We evaluate our framework for the IoT rule-based home automation setting with both simulated and real device data and study its efficacy in terms of both performance and security.

To summarize, in this paper, we propose the following contributions.

- We propose and develop an end-to-end encrypted system for securely analyzing IoT data using TEEs, particularly Intel SGX.
- We perform thorough evaluations to assess the framework with both simulated and real IoT device data.
- We conduct security evaluations for potential vulnerabilities of the system.

The rest of the paper is organized as follows. Section II presents some background on Intel SGX and IoT system. Section III explains the problem statement and threat model. Section IV introduces our framework architecture and its components. Section V describes the experiments and evaluation of the framework. Section VI and Section VII describes future work and related work, respectively. Finally, Section VIII concludes our work.

II. Background

A. Intel SGX

Intel's Software Guard Extensions (SGX) [23] is one of the state-of-the-art Trusted Execution Environments (TEE), that provides hardware-assisted secure area of memory where trusted part of an application can be executed. This ensures the integrity and confidentiality of an application's security-sensitive computation and data on a computer where all the privileged software such as operating system is potentially malicious. With the help of SGX, application developers can protect their code and data from modification or disclosure by an adversary by creating a private memory region called Enclave and deploying those sensitive code and information within the Enclave. The contents of enclaves are stored in the Enclave Page Cache (EPC), which is a piece of cryptographically protected memory with a page size of 4KB. Enclave is isolated from other processes or applications running at the same or higher privilege levels. No code, not even the higher privileged code such as Operating System (OS) or Virtual Machine Manager (VMM), can alter the contents of the Enclave, which makes it pretty robust from outside attacks and makes the attack surface of the SGX as minuscule as possible [24].

In SGX, a remote entity can cryptographically verify the integrity of an enclave and create a secure channel for sharing secrets with it. In Intel SGX architecture, this process is called *Attestation*. Intel SGX guarantees protection of data when it is maintained within the boundary of the enclave. When the data needs to be stored outside the enclave, SGX encrypts the contents before writing to untrusted memory, so that integrity and confidentiality of data remains intact. The process of encrypting the data is called *Sealing*. The data can be read back in by the enclave at a later date and then decrypted or unsealed. The encryption keys are derived internally on demand and are not exposed to the enclave.

B. IoT System and Security

In an IoT system, a collection of smart devices and users communicate with each other to achieve a common goal in the industrial and commercial environments as well as in our personal life [25]. IoT security refers to securing those connected devices and networks in the internet of things ecosystem. With cosmic IoT ecosystem, security threats are getting amplified and the IoT security must be designed to protect systems, networks, and data from a broad spectrum of attacks. Specially, cloud-based services provide solutions to connect the IoT devices and collect data from the most sensitive and personal domains of our life to process, manage, and analyze the data utilizing different data mining and machine learning techniques [26]–[33]. These solutions must ensure data anonymity, confidentiality, and integrity as well as prevent unauthorized access to the system.

There already exist some solutions for the IoT, such as Amazon AWS IoT [3], IBM Watson IoT Platform [4], Microsoft Azure IoT [5], Mozilla WebThings [6] and so forth. Even though, these solutions offer some level of security related to data [34]–[38], they are highly dependent on users' trust towards their platform. The users trust these services with their private data and an unfortunate event of compromised cloud could endanger the privacy and confidentiality of user data [39]. Therefore, we need a more robust strategy and technique to protect the data in both trusted and untrusted cloud environments.

C. Automation using IoT

One of the most powerful features of the IoT system is the ability to automate processes with the help of devices without any human intervention. The most obvious conveniences of the IoT automation are more operations, more accuracy, and low cost. Usually, IoT devices consist of embedded sensors and actuators, which help the devices to interact with the physical environment. Sensors can collect physical states, which are known as *Events*. These events, such as temperature reading, dust level, or door lock state, are sent to the cloud or hub for further processing. Afterwards, based on user-defined protocols and event data, appropriate action commands are sent to the device actuators. Generally, to transfer data between devices and cloud/hub, suitable protocol is used that supports limitations of the environment such as low powered devices. There are some IoT programming platforms such as Samsung's SmartThings, IFTTT, Apple's HomeKit [7], Zapier [8], openHAB [9] etc. that provide app-specific services of controlling and managing devices, data collection, and device interactions. They also provide tools that allow developers to write applications and automations through various APIs [40].

One of the most widely approved IoT programming platforms, especially for the home automation, is the rule-based *Trigger-Action* platform. This platform allows users to create custom simple and complex automations on services through rules that operate on the cloud. More specifically, the trigger-action rule platform performs some actions when a certain trigger event takes place. Typically, users define the rule by connecting a trigger-event in a service and an action-command in a separate service. When a device event matches the trigger-event, the appropriate action-command will be fired on the relevant service. For example, a user may define a rule: *Turn on the hall lights if motion is detected on the lawn*. Here, the trigger event is the detection of motion by the motion sensor and the action command is to turn on the lights using a smart switch. Triggers may contain trigger properties that determine under what circumstances the trigger event should occur. Similarly, action commands have action properties which are the parameters of the action [41]. These rule-based platforms are substantially benefiting smart home and industry automation systems. For instance, IFTTT has a community of 11 million users running over 1 billion rules each month with over 600 partner services [42].

[3]https://aws.amazon.com/iot/
[4]https://www.ibm.com/us-en/marketplace/internet-of-things-cloud
[5]https://azure.microsoft.com/en-us/overview/iot/
[6]https://iot.mozilla.org/
[7]https://www.apple.com/ios/home/
[8]https://zapier.com
[9]https://www.openhab.org

III. Problem Statement & Threat Model

A. Problem Statement

The use of rule-based platforms to control and interact with IoT devices is a powerful tool, but without proper and thorough security measures, it could lead to various unsafe conditions and unrecoverable loses. Generally, IoT devices expose three categories of information: *Stored Data* (i.e., device identifiers, user identifiers, activity logs), *Sensor Data* (i.e., information or physical states obtained from the environment by the sensors of devices), and *Activity Data* (i.e., information about how the devices are used via automation rules or user interaction) [14]. These information may be shared with two kinds of party: *First party* and *Third party*. First party includes the manufacturer of the IoT devices that are responsible for the device functionalities. On the other hand, Third parties are the organizations providing computing resources such as cloud providers or analytics companies. IoT devices expose those three types of data explicitly with these parties, which could pose potential data privacy issues.

In the IoT ecosystem, as the service providers are trusted with abundant user information, a major challenge arises in the form of balancing trust in these service providers and need for privacy. Although, the cloud service providers ensure that data is always protected at rest, during transmission, and computation; in reality they are vulnerable to many security threats, e.g., data breaches, especially the public cloud services [19]. In addition, severe lack of proper encryption techniques could expose sensitive information about the users. As a consequence, significant privacy risks could emerge as malicious third party services can track information about users for monetary purposes as well as learning user activities within homes. For instance, smart speakers in home can covertly record user conversations without permission and stream it to other users or parties [43]. Moreover, adversaries may physically access the machines deployed at the service providers premises or obtain root privileges of the machines by taking advantage of weak access control mechanism and thus steal sensitive information with ease.

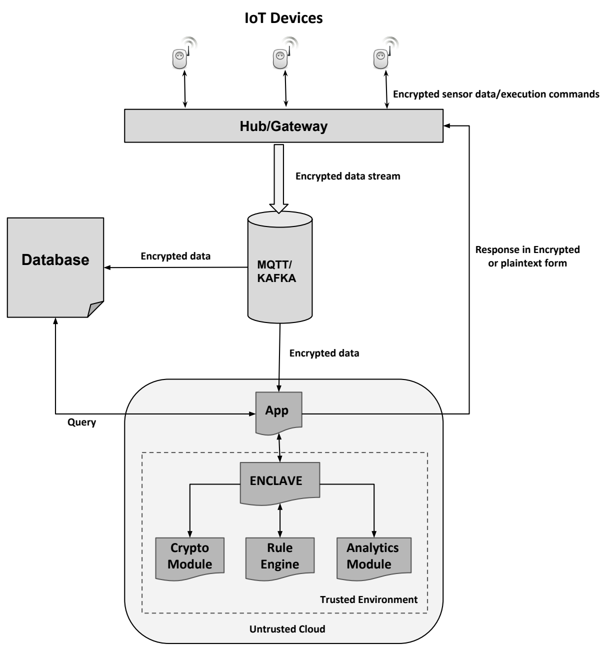

Figure 1: Framework architecture.

B. Threat Model

In this paper, we consider an adversary that seeks to surreptitiously gain insight into sensitive user information in the IoT system. More specifically, the adversary tries to access IoT device information and data stored in the cloud. The adversary tries to deploy a rule-level attack either by compromising the existing stored rule, by injecting malicious rule into the system, or by simply observing the rules to gain insight. Moreover, adversaries may eavesdrop the network traffic to retrieve information. The principal objective of the adversary would be to obtain private information of the user, specially his/her surrounding environment such as in a smart home system. We assume that adversaries cannot get root access to the devices or compromise communication protocols. Denial-of-Service (DoS) attacks [44] and protocol flaw attacks [45] are out of our scope.

IV. Proposed System Architecture

Our aim is to develop a secure cloud-based end-to-end encrypted data analytics platform, especially designed for IoT setting. Our goal is to alleviate data security and privacy issues by utilizing proper cryptographic techniques and trusted execution environments such as Intel SGX. In this paper, we particularly focus on developing a rule-based secure IoT platform for smart home automation in the untrusted cloud.

As discussed in section II, rule-based trigger-action platform is one of the most widely used IoT programming platforms in the world of IoT automation. Users provide trigger-action rules to automate their smart homes or smart industries leveraging the connectivity and ubiquitousness of IoT devices. These rules are stored and processed in the untrusted cloud platform or company-owned data silos, which poses a threat to the security and the privacy of the users. Moreover, lack of proper encryption techniques when communicating with untrusted cloud could expose sensitive information such as the identity of a device, user interactions with the device or private user information to eavesdroppers. Therefore, in our framework, we aim to use Intel SGX to guarantee confidentiality and integrity of sensitive data coming from IoT devices to untrusted remote platforms. By utilizing SGX's enclave features, we securely perform rule-based programming on delicate IoT data, so that no unauthorized personnel can unlawfully access data, user provided rules or any analytical results.

Usually, the required SGX enclave instances will be initialized by the cloud provider in the untrusted cloud platform. Once the enclave is initialized, it is expected to participate in a software attestation process, where it authenticates itself to a remote application server. Upon successful authentication, the application server is expected to disclose some secrets, in this case encryption/decryption keys, to the enclave on the untrusted platform over a secure communication channel.

The enclave in the cloud will communicate with the IoT devices in user homes via IoT gateways or hubs over HTTPS connection [46]. The communication protocol of HTTPS is encrypted with Transport Layer Security (TLS) [47], or formerly known as Secure Sockets Layer (SSL). In addition, to ensure end-to-end secure system, we use symmetric key encryption to communicate between the enclave in the cloud and the IoT hub. We use one of the most popular and widely adopted symmetric key encryption algorithms Advanced Encryption Standard (AES) [48] in our framework for this purpose. Hence, data in transit is always secure and eavesdropping on it is almost hopeless.

To create an automation, a user first needs to register a trigger-action rule in the cloud via any web or app interface. For instance, in Samsung SmartThings, automation is created via SmartApps, which is essentially an AWS Lambda function or a WebHook endpoint [49]. SmartThings follows REST API architecture to control and communicate with SmartThings devices from the cloud [50]. We follow a similar architecture in our platform so that our framework is aligned with the well-established SmartThings system. We also adopt SmartThings JSON rule structure [51].

After registering the smart devices, users can define their rules for the automation of their devices. The rule contains a list of conditions for trigger and a list of actions for the desired operation. On one hand, the conditions specify the device events received from smart devices that triggers the rule. The device event could be a state of the device (i.e., switch on/off, door open/closed etc.) or a sensor reading of the device (i.e., temperature 90F, dust level 20 PM10, energy 130 kwh etc.). On the other hand, the actions specify what rules actually do. They are the commands sent to specific devices to control or actuate them in response of the defined trigger condition. Listing 1 presents a sample rule in JSON format. These rules are then sent to the untrusted cloud enclave after encrypting it. In our framework, rules will be safely stored in the database in encrypted form at all times and are only decrypted inside the SGX enclave, thus preventing the attacker from accessing or manipulating the rules.

```
{
 "name": "If user is home, set the thermostate mode
 ↪  to cool and turn on the lights",
 "ruleID": "hpGHOiCPPeE9-Nz8CYTO1Cmj5",
 "userID": "AI4gwcJ6I6DE-s5QwIgXXtm3q-oUbYddxRmCqe",
 "actions": [
  {
   "if":{
    "equals": {
     "left": {
      "device": {
       "devices": ["420CC6DD-5932-9DF4-945D4539"],
       "component": "main",
       "capability": "PresenceSensor",
       "attribute": "presence"
      }
     },
     "right": {
      "string": "present"
     }
    },
    "then": [
     {
      "command": {
       "devices": ["61902075-855B-4EF6-FE7AD97B"],
       "commands": [
        {
         "component": "main",
         "capability": "ThermostatMode",
         "command": "cool",
         "arguments": []
        }
       ]
      }
     },
     {
      "command": {
       "devices": ["39A56C99-9A3A-45D7-D6537244"],
       "commands": [
        {
         "component": "main",
         "capability": "Switch",
         "command": "on",
         "arguments": []
        }
       ]
      }
     }
    ],
    "else": []
   }
  }
 ]
}
```

Listing 1: Sample Rule in JSON.

The framework architecture is illustrated in Figure 1. The IoT devices send device states or sensor values to the cloud via the hubs or gateways. The data is encrypted in the hub/gateway before sending to cloud and upon receiving a stream of such data from devices, SGX loads and decrypts the associated rules with the device in the enclave. As the system needs to deal with multiple data streams from various devices [52], we use MQTT (Message Queuing Telemetry Transport) [53], which is designed as a lightweight publish/subscribe messaging transport, as our connectivity protocol. Additionally, note that, data is decrypted only inside the enclave using the secret key, which ensures data protection in transmission. Now, device event is compared with the condition of the rule (i.e., trigger) and generate corresponding response using action-command in the rule. This action-command is then encrypted and sent to the appropriate hub/gateway to control or actuate for the automation. The hub/gateway eventually takes care of transmitting the decision to the particular IoT device after decryption. Furthermore, users can define rules such that when the rule is triggered, user receives a notification instead of device actuation.

```
{
    "deviceID": "420CC6DD-5932-9DF4-945D4539",
    "deviceEvents": [
        {
            "component": "main",
            "capability": "PresenceSensor",
            "attribute": "presence",
            "value": {
                "string": "present"
            },
            "unit": "",
            "data": []
        }
    ]
}
```

Listing 2: Sample Device Event in JSON.

For instance, Listing 2 represents a sample device event received in the enclave. The event is generated from a *Presence Sensor*. It contains the sensor attribute *Presence* and current reading value, which is *present*. After receiving the device event, the rule-engine in the enclave fetches from the cache corresponding rules for that device, in this case, the rule in Listing 1. The rule-engine then proceeds to inspect the *equals* condition of the rule. Here, the device attribute value and the rule condition value are the same, that is *present*. Therefore, the rule is satisfied and will trigger the action commands, which are in the *then* clause of the rule. These action commands will be sent to respective devices and executed there. In this example, a command will be sent to the *thermostat* to set its state to *cool* and another command to a smart *switch* to set its state to *on*.

To summarize, our framework ensures the integrity and confidentiality of IoT data and user rules and perform secure analytics by leveraging isolated memory containers such as SGX enclave. Moreover, we ensure data security in transit

Case	Type	Ruleset size	Devices count	Total Device Events	Cache size
No SGX SGX (w/o encryption) SGX	Simulation	100	32	10000	100
		400			
		1000			
		5000			
		10000			
	Real	10		1000	

Table I: Experimental setting

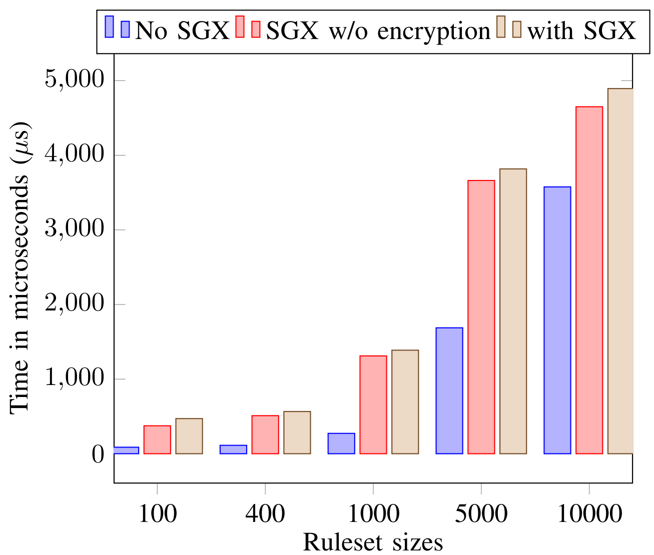

Figure 2: Average execution time of the experiment based on 10000 device events and ruleset sizes of 100, 400, 1000, 5000, and 10000; performed for three cases: no SGX, SGX without encryption, and with SGX.

from IoT devices to cloud service provider with SGX by following robust end-to-end encryption mechanism of the data. That means, in transit data is always kept in encrypted form, except when it is in the SGX. Even if adversaries manage to steal the data in transit, they cannot reveal any information from it as it will be always encrypted.

V. Implementation & Evaluation

We evaluate the proposed framework by measuring computational time overhead of the whole process with simulated IoT data as well as data from real devices. In addition, we analyze memory access traces of the program to empirically evaluate the possibility of security threats due to an adversary that may analyze access patterns to encrypted data [54].

Computational Evaluation. Our goal of this evaluation is to measure the computational time overhead of the framework. More specifically, we want to discover how the integration of Intel SGX and the cryptographic techniques alter the time overhead of the process. For this experiment, we use simulated IoT data and rules in accordance with the Samsung's SmartThings format (discussed in section IV). The experimental setting is represented in Table I. We consider three cases for the experiment: *No SGX*, that provides no security guarantee of data; *with SGX but without encryption mechanism*, which may provide integrity of data but lacks confidentiality; and *with SGX*, that provides total security guarantee. For SGX cloud, we use a system containing 8-core i7-6700 (Skylake) processor operating at 3.4GHz, running Ubuntu 18.04 with 64GB RAM, and client programs written in python (3.6) for simulating IoT devices.

At first, the client generates a set of rules $R = \{r_1, r_2, \ldots, r_n\}$ for a set of his devices $D = \{d_1, d_2, \ldots, d_m\}$. A device may have multiple rules associated with itself, e.g., r_1 and r_2 might both belong to d_1. Then, client encrypts the rules using AES encryption scheme, and sends to the SGX cloud. The SGX cloud loads the encrypted rules into the enclave and decrypts them. After parsing the rules, SGX enclave re-encrypts each rule separately with enclave's own secure key k_{SGX} and stores it in a database as a $\langle$key, value$\rangle$ pair, where *key* is the device ID and *value* is the encrypted rules associated with that device. There's also a caching system (i.e., LRU, LFU) in the enclave to cache most frequently/recently used rules. Now, simulated IoT devices periodically send device states or sensor values to the cloud in encrypted form. Upon receiving this stream of encrypted data, the SGX loads the device data into the enclave, decrypts it, fetches associated rules from the database into the enclave, and decrypts the rules using k_{SGX}. It then generates the corresponding response using the triggers and actions specified in rules and sends the response back to the device after encrypting it. Upon receiving the response from SGX, device first verifies the integrity and the authenticity of the message. If both checks pass, device executes the message.

Figure 2 represents the average execution time comparison for the three cases mentioned in Table I for varying number of rules and 10000 simulated device events. Needless to say, execution time of the experiment with SGX takes longer than the operation when we do not include SGX. Fortunately, the time execution overhead is not that significant.

Moreover, we perform a basic experiment with real IoT devices to evaluate the soundness of the system. We use sensor data from *Foobot* [55] to control *Philip Hue Bulb* [56] with some predefined rules in SGX. *Foobot* is an indoor air quality monitor sensor, which can measure temperature, humidity, carbon dioxide level, volatile compounds in the air, and so on. Philip Hue Bulb is a smart bulb, which can be controlled with apps to turn on or off. At first, we store some predefined rules in the SGX enclave, where the trigger component of the rules involve *Foobot* sensor values (i.e, temperature, humidity, and carbon dioxide level) and action component involve changing the status of the *Philip*

Set Comparison	KL divergence score
S_1 vs S_2	0.199
S_2 vs S_3	0.46
S_1 vs S_3	0.39

Table II: KL divergence score of memory trace distributions.

Hue light bulb. Then, we periodically gather temperature, humidity and carbon dioxide level values from *Foobot* and send to SGX cloud after encryption. Just like the above experiment, SGX enclave generates a response command according to the rule, which is then encrypted and sent to a python written program simulating the behavior of a hub. The response command is then decrypted and sent via https connection to the smart bulb to change the state.

We observe the overall average execution and network delay time for the previously mentioned three cases with 10 predefined rules and 1000 device events (sensor values). We notice a similar result as before; *SGX* incurring a slight time overhead with average overall time of 1.1×10^5 μs, where *no SGX* and *SGX without encryption* achieved 9.3×10^4 μs and 1.0×10^5 μs, respectively.

Security Evaluation. As an adversary may obtain memory access traces of the program execution, s/he can infer sensitive information by analyzing access patterns from these traces, if the program displays distinguishing characteristics [54]. Therefore, our goal of this evaluation is to discover if the memory access traces of the program are indistinguishable or not. For this purpose, we use a randomly selected ruleset of size 10 and 3 set of device events (i.e., S_1, S_2, S_3) with each set containing 10 instances. Among these 3 sets, S_1 and S_2 are almost identical. We use Intel Pin tool [57] to capture memory access traces (i.e., sequence of read and write operations) of the program executing in SGX simulation mode. We create probability distributions from these traces and use *Kullback–Leibler divergence* (KL divergence) to differentiate between each traces. Table II represents the comparison among the traces in terms of KL divergence score. From the table, we can observe that KL divergence score of near identical set S_1 and S_2 are lower than other two non-identical set comparison. As low KL divergence score means two distributions are more similar, we can deduce that almost indistinguishable data events create same memory access patterns in the experiment. As a result, this could be vulnerable to side channel attacks as adversaries can resend similar data continuously to the SGX cloud and observe memory access sequences to infer secrets from the enclave.

VI. Limitations & Future Work

Although Intel SGX is secure in design, it still suffers from pattern leakage attacks such as side channel attack as in [23]. These attacks, both memory level (shown in Section V) and network level, leak information and endanger data security. We plan to thwart such attacks by hiding the memory access patterns by introducing inconsistency in the side channel information [58] with injection of dummy data or by incorporating oblivious random access memory technique. Also, to make the security more robust, we plan to incorporate efficient access control mechanism that features decentralized authorization, protected permissions, and transitive permission delegation. Furthermore, Intel SGX only supports a limited memory space (up to 128MB EPC) for data and code inside the enclave. This memory limitation calls for a distributed SGX system that will handle streaming data from IoT devices without any memory issues or sluggishness of the system. In the future, we aim to make our SGX system distributed, so that the framework do not face any unwanted memory issue or system slowdown.

VII. Related Work

There has been some significant research on secure IoT data management over the past couple of years. *Talos* stores IoT data securely in the cloud using cryptographic techniques and allows query processing over encrypted data [59]. Even though the system is proved to be secure, the proof mainly depends on the robustness of the encryption algorithm as well as the application logic. In [60], authors present a secure IoT data management system that uses a blockchain [61]. They develop a decentralised framework that uses Ethereum smart contracts [62] to control access permission of data, store audit trail of data access in the blockchain, and store raw data in encrypted form using Intel SGX. Although the framework is integrated with Intel Sgx and blockchain to ensure the security of the data, it does not handle any processing of the data securely. In addition, [63] utilizes Intel SGX to create enclaves that run virtual clones of physical IoT devices in the cloud to store, process, and share device generated data.

VIII. Conclusion

As the usage of IoT devices increase, it is imperative that we protect sensitive user information and automation policy rules from malicious attacks. This paper proposes a framework that provides secure data analytics system by leveraging Intel SGX and strong cryptographic techniques. We execute basic trigger-action rule-based program for automation in the SGX enclave to ensure user privacy, data integrity and confidentiality. Moreover, strong encryption mechanism guarantees data privacy in transit and storage, making the system end-to-end encrypted. We evaluate the proposed framework by using data from simulated and real IoT devices, by performing rule-based decision making inside SGX enclave securely, and show that the overhead due to encryption and SGX based processing is not significant.

Acknowledgment

The research reported herein was supported in part by NIH award 1R01HG006844, NSF awards CICI-1547324, IIS-1633331, CNS-1837627, OAC-1828467, DMS-1737978, DGE-2039542, MRI-1828467, ARO award W911NF-17-1-0356, FAIN award number 1906630, and IBM faculty award (Research).

References

[1] B. Insider, "The security and privacy issues that come with the internet of things)," https://www.businessinsider.com/iot-security-privacy, retrieved: 2020-06-27.

[2] D. Goldman, "Your samsung tv is eavesdropping on your private conversations)," https://money.cnn.com/2015/02/09/technology/security/samsung-smart-tv-privacy/index.html, retrieved: 2020-06-27.

[3] L. Khan and D. McLeod, "Audio structuring and personalized retrieval using ontologies," in *Proceedings IEEE Advances in Digital Libraries 2000*. IEEE, 2000, pp. 116–126.

[4] L. Wang and L. Khan, "Automatic image annotation and retrieval using weighted feature selection," *Multimedia Tools and Applications*, vol. 29, no. 1, pp. 55–71, 2006.

[5] V. A. Petrushin and L. Khan, *Multimedia data mining and knowledge discovery*. Springer, 2007, vol. 521.

[6] S. Abrol and L. Khan, "Twinner: understanding news queries with geo-content using twitter," in *Proceedings of the 6th Workshop on Geographic information Retrieval*, 2010, pp. 1–8.

[7] L. Wang, L. Liu, and L. Khan, "Automatic image annotation and retrieval using subspace clustering algorithm," in *Proceedings of the 2nd ACM international workshop on Multimedia databases*, 2004, pp. 100–108.

[8] M. A. Awad and L. R. Khan, "Web navigation prediction using multiple evidence combination and domain knowledge," *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, vol. 37, no. 6, pp. 1054–1062, 2007.

[9] M. Awad, L. Khan, and B. Thuraisingham, "Predicting www surfing using multiple evidence combination," *The VLDB Journal*, vol. 17, no. 3, pp. 401–417, 2008.

[10] S. Nessa, M. Abedin, W. E. Wong, L. Khan, and Y. Qi, "Software fault localization using n-gram analysis," in *International Conference on Wireless Algorithms, Systems, and Applications*. Springer, 2008, pp. 548–559.

[11] I.-L. Yen, J. Goluguri, F. Bastani, L. Khan, and J. Linn, "A component-based approach for embedded software development," in *Proceedings Fifth IEEE International Symposium on Object-Oriented Real-Time Distributed Computing. ISIRC 2002*. IEEE, 2002, pp. 402–410.

[12] F. Shaon, M. Kantarcioglu, Z. Lin, and L. Khan, "Sgx-bigmatrix: A practical encrypted data analytic framework with trusted processors," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 2017, pp. 1211–1228.

[13] S. Fisher, "Why does iot security present such a risk?)," https://www.avast.com/c-iot-security-risks, retrieved: 2020-06-27.

[14] J. Ren, D. J. Dubois, D. Choffnes, A. M. Mandalari, R. Kolcun, and H. Haddadi, "Information exposure from consumer iot devices: A multidimensional, network-informed measurement approach," in *Proceedings of the Internet Measurement Conference*, 2019, pp. 267–279.

[15] M. Masud, L. Khan, and B. Thuraisingham, *Data mining tools for malware detection*. CRC Press, 2011.

[16] M. M. Masud, L. Khan, and B. Thuraisingham, "A hybrid model to detect malicious executables," in *2007 IEEE International Conference on Communications*. IEEE, 2007, pp. 1443–1448.

[17] B. Thuraisingham, L. Khan, M. M. Masud, and K. W. Hamlen, "Data mining for security applications," in *2008 IEEE/IFIP International Conference on Embedded and Ubiquitous Computing*, vol. 2. IEEE, 2008, pp. 585–589.

[18] K. Al-Naami, A. El Ghamry, M. S. Islam, L. Khan, B. M. Thuraisingham, K. W. Hamlen, M. Alrahmawy, and M. Rashad, "Bimorphing: A bi-directional bursting defense against website fingerprinting attacks," *IEEE Transactions on Dependable and Secure Computing*, 2019.

[19] F. Paul, "Top 10 iot vulnerabilities," https://www.networkworld.com/article/3332032/top-10-iot-vulnerabilities.html, retrieved: 2020-06-19.

[20] S. Larson, "Stuffed toys leak millions of voice recordings from kids and parents)," https://money.cnn.com/2017/02/27/technology/cloudpets-data-leak-voices-photos/index.html, retrieved: 2020-06-27.

[21] G. Ayoade, K. A. Akbar, P. Sahoo, Y. Gao, A. Agarwal, K. Jee, L. Khan, and A. Singhal, "Evolving advanced persistent threat detection using provenance graph and metric learning."

[22] M. S. Islam, H. Verma, L. Khan, and M. Kantarcioglu, "Secure real-time heterogeneous iot data management system," in *2019 First IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA)*. IEEE, 2019, pp. 228–235.

[23] V. Costan and S. Devadas, "Intel sgx explained." *IACR Cryptology ePrint Archive*, vol. 2016, no. 086, pp. 1–118, 2016.

[24] V. Karande, E. Bauman, Z. Lin, and L. Khan, "Sgx-log: Securing system logs with sgx," in *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, 2017, pp. 19–30.

[25] S. Sicari, A. Rizzardi, L. A. Grieco, and A. Coen-Porisini, "Security, privacy and trust in internet of things: The road ahead," *Computer networks*, vol. 76, pp. 146–164, 2015.

[26] M. M. Masud, J. Gao, L. Khan, J. Han, and B. Thuraisingham, "A multi-partition multi-chunk ensemble technique to classify concept-drifting data streams," in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 2009, pp. 363–375.

[27] A. Haque, L. Khan, M. Baron, B. Thuraisingham, and C. Aggarwal, “Efficient handling of concept drift and concept evolution over stream data,” in *2016 IEEE 32nd International Conference on Data Engineering (ICDE)*. IEEE, 2016, pp. 481–492.

[28] M. M. Masud, T. M. Al-Khateeb, L. Khan, C. Aggarwal, J. Gao, J. Han, and B. Thuraisingham, “Detecting recurring and novel classes in concept-drifting data streams,” in *2011 IEEE 11th International Conference on Data Mining*. IEEE, 2011, pp. 1176–1181.

[29] M. M. Masud, J. Gao, L. Khan, J. Han, and B. Thuraisingham, “Classification and novel class detection in data streams with active mining,” in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 2010, pp. 311–324.

[30] M. Awad, L. Khan, F. Bastani, and I.-L. Yen, “An effective support vector machines (svms) performance using hierarchical clustering,” in *16th IEEE international conference on tools with artificial intelligence*. IEEE, 2004, pp. 663–667.

[31] T. Al-Khateeb, M. M. Masud, L. Khan, C. Aggarwal, J. Han, and B. Thuraisingham, “Stream classification with recurring and novel class detection using class-based ensemble,” in *2012 IEEE 12th International Conference on Data Mining*. IEEE, 2012, pp. 31–40.

[32] C. Breen, L. Khan, and A. Ponnusamy, “Image classification using neural networks and ontologies,” in *Proceedings. 13th International Workshop on Database and Expert Systems Applications*. IEEE, 2002, pp. 98–102.

[33] M. M. Masud, L. R. Khan, B. M. Thuraisingham, Q. Chen, J. Gao, and J. Han, “Systems and methods for detecting a novel data class,” Oct. 20 2015, uS Patent 9,165,051.

[34] M. M. Masud, T. M. Al-Khateeb, K. W. Hamlen, J. Gao, L. Khan, J. Han, and B. Thuraisingham, “Cloud-based malware detection for evolving data streams,” *ACM transactions on management information systems (TMIS)*, vol. 2, no. 3, pp. 1–27, 2008.

[35] P. Parveen, J. Evans, B. Thuraisingham, K. W. Hamlen, and L. Khan, “Insider threat detection using stream mining and graph mining,” in *2011 IEEE Third International Conference on Privacy, Security, Risk and Trust and 2011 IEEE Third International Conference on Social Computing*. IEEE, 2011, pp. 1102–1110.

[36] G. Lavee, L. Khan, and B. Thuraisingham, “A framework for a video analysis tool for suspicious event detection,” *Multimedia Tools and Applications*, vol. 35, no. 1, pp. 109–123, 2007.

[37] M. Tu, P. Li, I.-L. Yen, B. M. Thuraisingham, and L. Khan, “Secure data objects replication in data grid,” *IEEE Transactions on dependable and secure computing*, vol. 7, no. 1, pp. 50–64, 2008.

[38] M. Abedin, S. Nessa, L. Khan, and B. Thuraisingham, “Detection and resolution of anomalies in firewall policy rules,” in *IFIP Annual Conference on Data and Applications Security and Privacy*. Springer, 2006, pp. 15–29.

[39] J. Daubert, A. Wiesmaier, and P. Kikiras, “A view on privacy & trust in iot,” in *2015 IEEE International Conference on Communication Workshop (ICCW)*. IEEE, 2015, pp. 2665–2670.

[40] Z. B. Celik, G. Tan, and P. D. McDaniel, “Iotguard: Dynamic enforcement of security and safety policy in commodity iot.” in *NDSS*, 2019.

[41] I. Bastys, M. Balliu, and A. Sabelfeld, “If this then what? controlling flows in iot apps,” in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, 2018, pp. 1102–1119.

[42] J. A. Martin and M. Finnegan, “What is ifttt? how to use if this, then that services,” https://www.computerworld.com/article/3239304/what-is-ifttt-how-to-use-if-this-then-that-services.html, retrieved: 2020-06-19.

[43] N. Chokshi, “Is alexa listening? amazon echo sent out recording of couple’s conversation,” https://www.nytimes.com/2018/05/25/business/amazon-alexa-conversation-shared-echo.html, retrieved: 2020-06-19.

[44] I. X.-F. Research, “The weaponization of iot devices,” https://www.ibm.com/downloads/cas/6MLEALKV, retrieved: 2020-06-19.

[45] E. Ronen, A. Shamir, A.-O. Weingarten, and C. O’Flynn, “Iot goes nuclear: Creating a zigbee chain reaction,” in *2017 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2017, pp. 195–212.

[46] E. Fernandes, J. Jung, and A. Prakash, “Security analysis of emerging smart home applications,” in *2016 IEEE symposium on security and privacy (SP)*. IEEE, 2016, pp. 636–654.

[47] Z. Kerravala, “What is transport layer security (tls)?” https://www.networkworld.com/article/2303073/lan-wan-what-is-transport-layer-security-protocol.html, retrieved: 2020-06-22.

[48] J. Daemen and V. Rijmen, “Aes proposal: Rijndael,” 1999.

[49] S. S. Developers, “Smartapps,” https://smartthings.developer.samsung.com/docs/smartapps/smartapp-basics.html, retrieved: 2020-06-22.

[50] ——, “Create an automation,” https://smartthings.developer.samsung.com/docs/smartapps/automation.html, retrieved: 2020-06-22.

[51] ——, “Working with rules,” https://smartthings.developer.samsung.com/docs/rules/overview.html, retrieved: 2020-06-22.

[52] G. De Francisci Morales, A. Bifet, L. Khan, J. Gama, and W. Fan, “Iot big data stream mining,” in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 2016, pp. 2119–2120.

[53] MQTT, “Message queuing telemetry transport (mqtt),” http://mqtt.org/, retrieved: 2020-06-22.

[54] M. S. Islam, M. Kuzu, and M. Kantarcioglu, "Access pattern disclosure on searchable encryption: ramification, attack and mitigation." in *Ndss*, vol. 20. Citeseer, 2012, p. 12.

[55] Foobot®, "Foobot," https://foobot.io/features/, retrieved: 2019-08-22.

[56] Philips, "Philips hue bulbs," https://www2.meethue.com/en-us/bulbs, retrieved: 2019-08-24.

[57] C.-K. Luk, R. Cohn, R. Muth, H. Patil, A. Klauser, G. Lowney, S. Wallace, V. J. Reddi, and K. Hazelwood, "Pin: building customized program analysis tools with dynamic instrumentation," *Acm sigplan notices*, vol. 40, no. 6, pp. 190–200, 2005.

[58] S. Chandra, V. Karande, Z. Lin, L. Khan, M. Kantarcioglu, and B. Thuraisingham, "Securing data analytics on sgx with randomization," in *European Symposium on Research in Computer Security*. Springer, 2017, pp. 352–369.

[59] H. Shafagh, A. Hithnawi, A. Droescher, S. Duquennoy, and W. Hu, "Talos: Encrypted query processing for the internet of things," in *Proceedings of the 13th ACM conference on embedded networked sensor systems*. ACM, 2015, pp. 197–210.

[60] G. Ayoade, V. Karande, L. Khan, and K. Hamlen, "Decentralized iot data management using blockchain and trusted execution environment," in *2018 IEEE International Conference on Information Reuse and Integration (IRI)*. IEEE, 2018, pp. 15–22.

[61] S. Nakamoto *et al.*, "Bitcoin: A peer-to-peer electronic cash system," 2008.

[62] V. Buterin *et al.*, "Ethereum white paper.(2013)," *URL https://github. com/ethereum/wiki/wiki/White-Paper*, 2013.

[63] R. Tahir, A. Raza, F. Zaffar, F. U. Ghani, and M. Zulfiqar, "Using sgx-based virtual clones for iot security," in *2018 IEEE 17th International Symposium on Network Computing and Applications (NCA)*. IEEE, 2018, pp. 1–4.