Secure Real-Time Heterogeneous IoT Data Management System

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

Harsh Verma
Department of Computer Science
The University of Texas at Dallas
Richardson, Texas 75080
harsh.verma@utdallas.edu

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

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

***Abstract*—The growing adoption of IoT devices in our daily life engendered a need for secure systems to safely store and analyze sensitive data as well as the real-time data processing system to be as fast as possible. The cloud services used to store and process sensitive data are often come out to be vulnerable to outside threats. Furthermore, to analyze streaming IoT data swiftly, they are in need of a fast and efficient system. The Paper will envision the aspects of complexity dealing with real time data from various devices in parallel, building solution to ingest data from different IOT devices, forming a secure platform to process data in a short time, and using various techniques of IOT edge computing to provide meaningful intuitive results to users. The paper envisions two modules of building a real time data analytics system.**

In the first module, we propose to maintain confidentiality and integrity of IoT data, which is of paramount importance, and manage large-scale data analytics with real-time data collection from various IoT devices in parallel. We envision a framework to preserve data privacy utilizing Trusted Execution Environment (TEE) such as Intel SGX, end-to-end data encryption mechanism, and strong access control policies. Moreover, we design a generic framework to simplify the process of collecting and storing heterogeneous data coming from diverse IoT devices.

In the second module, we envision a drone-based data processing system in real-time using edge computing and on-device computing. As, we know the use of drones is growing rapidly across many application domains including real-time monitoring, remote sensing, search and rescue, delivery of goods, security and surveillance, civil infrastructure inspection etc. This paper demonstrates the potential drone applications and their challenges discussing current research trends and provide future insights for potential use cases using edge and on-device computing.

***Index Terms*—IoT, data security, trusted execution environment, machine learning, edge device, edge computing, cloud, drones, real-time systems, embedded devices.**

I. Introduction

Recent emergence of the Internet of Things (IoT) has caused data deluge, which needs careful maintenance and secure storage system to ensure data integrity and protection, along with fast processing technologies to cope with the myriad amount of data that reflect the era of big data. IoT devices are found in our day-to-day life, such as smart homes, industrial automation, agriculture, smart transportation, healthcare etc. They have become a fundamental part of modern society and as a result, contains highly vulnerable and sensitive data that must be carefully dealt with. Moreover, the abundant use of IoT devices generates continuous flow of information that must be processed swiftly.

Recent development of cloud computing has provided the opportunity to use cloud-based services to collect, process, analyze, and mine large amounts of IoT data, which is both cost effective and less time consuming. 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 [1]. Moreover, 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.

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, only security policy-based rules or operating system access control mechanisms of the cloud services are not sufficient to thwart adversaries from stealing sensitive data and information.

The availability of cheap IoT devices has paved the way for its incredible proliferation in our surroundings. These devices are slowly becoming an indispensable component of our habitual lives. As a result of the regular and frequent usage, these devices are generating enormous amounts of data that is stored and analyzed in cloud services. As many IoT applications demands real-time processing of its data, sometimes these cloud services fail to perform efficiently due to lack of proper distributed architecture to manage real-time data.

Moreover, one of the major obstacles in real-time analytics with cloud-based servers is the latency caused over the network transmission. Some applications need real-time results with strict delay and cloud-based services may not provide fast responses. This is where edge computing comes into play, to process computation tasks as close as possible to the data sources. Recent advancement of edge computing with deep learning frameworks to provide intelligent services on the edge, has received great attention among IoT applications. Even though edge computing seems to mitigate the drawback, its still may not be fast enough to provide output by performing analytics using machine learning frameworks. As a consequence, researchers are trying to run neural network models in low powered embedded devices, which would deliver the opportunity to perform analytics on IoT devices itself, opening up a new era of IoT applications.

In this paper, we envision two modules of building a real-time data analytics system by challenging the above mentioned issues. In module 1, we present a secure data analytics system with the help of Trusted Execution Environments (TEEs). With the help of secure data processing module, we ensure the integrity and confidentiality of IoT data by performing simple analytics on a popular TEE called Intel Software Guard Extensions (SGX) [2]. 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. Moreover, we ensure data security in transit from IoT devices to cloud service provider with SGX by following 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.

Additionally, as we have to deal with diverse amount of data from different IoT devices, we propose a generic framework to simplify the heterogeneity of the process. With the advent of enormous real-time data from IoT devices, its highly important to build a dynamic system that could ease the way to store dynamic diverse data and process it securely. As a result, we propose a solution to handle the data in a standard format before storing them and using for further processing.

In module 2, we envision a drone-based data processing system on the edge and device itself with help of TinyML [3]. As drones are easier to deploy anywhere and not that expensive, these can be used in many applications. Recently, data processing with the help of drones has become a popular topic, where data are captured from drones and sent to the cloud server to perform analytics. However, as discussed above, this could introduce severe latency as the continuous data transfer between the drones and the cloud servers heavily burdens the networks. Here, we propose a drone-based data processing system on the edge or even on the device itself. Introduction of edge computing that can run neural network models has made it possible to run data analytics close to the device and TinyML has given us hope of processing data in the device itself. We propose a use case of detecting abnormal or suspicious behavior of ground vehicles by capturing images from the drone and processing it with machine learning model in the embedded device attached to the drone or on the edge.

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

- We propose an end-to-end encrypted system for securely analyzing IoT data using TEEs, particularly Intel SGX.
- We propose to simplify the heterogeneity of data collection and storage by a generic framework.
- We envision a drone-based data processing system on the edge and device itself.

The rest of the paper is organized as follows. Section II presents some background on related topics. Section III introduces some related work. Section IV describes the proposed two modules in detail. Finally, Section V concludes our work.

II. Background

A. Trusted Execution Environment

Trusted computing has been an active field of research over the last decade, and several architectures and methodologies have been proposed for many devices. The goal of trusted computing is to give users guarantees about the behaviour of the software running on their devices by developing secure technologies. More specifically, a device can be trusted if it always behaves in the expected manner for the intended purpose. This means that even when an attacker gains control of the system, he cannot make it misbehave. This is a complex goal, covering many aspects, resulting in a wide range of solutions based on software, hardware, or co-design of both. As a result, Trusted Execution Environments (TEEs) are designed, which is a secure area inside the main processor. It runs in a parallel way with the operating system in an isolated environment, which ensures the software components loaded in the TEE are protected with respect to confidentiality and integrity. Everything outside TEE is untrusted. Moreover, unpredictable software running in the main system software cannot affect software running in the TEE.

One of the state-of-the-art TEE is *Intel SGX*. It was introduced in 2015 with the sixth generation Intel Core processors, which are based on the Skylake micro-architecture. Intel's Software Guard Extensions (SGX) [2] is an extension to the Intel x86 architecture, that provides hardware-assisted trusted execution environment that allows trusted part of an application to be executed in a secure area of memory. 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.

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 little as possible.

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. This 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 Analytics Platforms

IoT analytics is basically an application, used to understand the large volumes of data generated from the connected IoT devices and data analysis tools around the world. With the help of widely increased usage of IoT sensing devices, some sensory data are being collected over time, specialized for different fields, applications and infrastructures. The aim of these sensing devices is to generate fast/real-time big data streams, depending on the requirement of the applications. Yet the main challenge is to implement analytics over these data streams, to gain meaningful insights, more accurate predictions, sensitive information and make control decisions. Hence, this is how IoT contributes in improving the world of business, technology and quality-of-life.

C. Edge Computing

The idea of edge computing is to perform the computation of data as close as possible to the data sources, rather than performing that on remote or distant sources. This can be achieved by simply adding an edge device nearby the data resource or data generating devices. Generally, edge devices are competent in both computation and communication tasks. However, the computation part which is too complicated, are sent to be performed at remote and more powerful servers. For performing some basic machine learning tasks (i.e: language processing, object detection, face recognition etc.), handling massive streams of IoT data, also for reducing data latency and dependence on the cloud, edge computing can be a potential and slick solution.

D. TinyML

Tiny machine learning (TinyML) is broadly defined as a fast growing field of machine learning technologies and applications. It refers to performing data analytics at extremely low power embedded devices, typically in the mW range and below. This gives us the opportunity to run machine learning algorithms in IoT devices itself such as any sensors, instead of sending data back and forth from a server, helping to improve latency, privacy, connectivity, and power consumption. Typically, TinyML is designed to run on microcontroller chips that are small, low-powered computing devices (a few milliwatts of power) that are often embedded within hardware that requires basic computation, including household appliances and IoT devices. They are often optimized for low energy consumption and small size, at the cost of reduced processing power, memory, and storage. By running machine learning inference on microcontrollers, developers can add AI to a vast range of hardware devices without relying on network connectivity, which is often subject to constraints such as bandwidth and power, and results in high latency. As no data has to leave the device, it also helps to preserve privacy.

E. Unmanned Ariel Vehicles

The use of Unmanned Aerial Vehicles (UAVs), also known as Drones, is growing swiftly across many domains, specially civil application domain. Emerging technologies such as 4G/5G networks have made it possible for the UAVs equipped with cameras, sensors, and GPS receivers in delivering Internet of Things services from great heights, creating an airborne domain of the IoT. Some of the UAV applications include remote sensing, search and rescue, real-time traffic monitoring, providing wireless coverage, construction & infrastructure inspection, delivery of goods, security and surveillance, and precision agriculture. As UAVs provide new opportunities in different application domains, it is emerging as a new revolution due to their ease of deployment, low maintenance cost, and high-mobility.

III. Related Works

A. Drones: Use Cases

1) UAVs can be used to transport food, packages and other goods. In health-care field, ambulance drones can deliver medicines, immunizations, and blood samples, into and out of unreachable places. They can rapidly transport medical instruments in the crucial few minutes after cardiac arrests. They can also include live video streaming services allowing paramedics to remotely observe and instruct on-scene individuals on how to use the medical instruments.
2) Cloud and Big Data: A cloud for UAVs contains data storage and high-performance computing, combined with big data analysis tools. It can provide an economic and efficient use of centralized resources for decision making and network-wide monitoring. If UAVs are utilized by a traditional cellular network operator (CNO), the cloud is just the data center of the CNO (similar to a private cloud), where the CNO can choose to share its knowledge with some other CNOs or utilize it for its own business uses. On the other hand, if the UAVs are utilized by an infrastructure provider, the infrastructure provider can utilize the cloud to gather information from mobile virtual network operators and service providers. Under such scenario, it is important to

guarantee security, privacy, and latency. To better exploit the benefit of the cloud, we can use a programmable network allowing dynamic updates based on big data processing, for which network functions virtualization and software defined networking can be research trends.

B. Edge Computing with IoT

Edge computing has become an important solution to break the bottleneck of emerging technologies by virtue of its advantages of reducing data transmission, improving service latency and easing cloud computing pressure. The edge computing architecture will become an important complement to the cloud, even replacing the role of the cloud in some scenarios.

In the development of edge computing, there have been various new technologies aimed at working at the edge of the network, with the same principles but different focuses, such as Cloudlet [4], Fog Computing [5] and Mobile Edge Computing [6]. We use a common term "edge computing" for this set of emerging technologies.

Edge Computing Systems: Solutions for edge computing systems are blooming. For DL services with complex configuration and intensive resource requirements, edge computing systems with advanced and excellent micro service architecture are the future development direction. Currently, Kubernetes [7] is a mainstream container-centric system for the deployment, maintenance, and scaling of applications in cloud computing [8]. Based on Kubernetes, Huawei develops its edge computing solution "KubeEdge" [9] for networking, application deployment, and metadata synchronization between the cloud and the edge (also supported in Akraino Edge Stack). "OpenEdge" [10] focus on shielding computing framework and simplifying application production. For IoT, Azure IoT Edge and EdgeX are devised for delivering cloud intelligence to the edge by deploying and running AI on cross-platform IoT devices.

C. TinyML

Over the past decade, there has been some significant work [11]–[18] conducted to handle myriad of data in efficient way. Now-a-days, more and more researchers are focusing on TinyML. Some of the related work on Tiny ML is described below:

1) TinyML audio algorithms using asynchronous audio events and Nyquist sampled audio: development of audio TinyML algorithms and their implementation onto embedded hardware have enabled pre-ASIC study of the power-latency tradeoff of deep network architectures and feature representations. TinyML deep networks is used on audio tasks including keyword spotting and speaker verification.
2) Voice Separation with tiny ML on the edge: Dr. Niels H. Pontoppidan, Research Area Manager, Augmented Hearing Science, Eriksholm Research Centre, Oticon: With the recent advances in many areas of tiny ML several use cases where tiny ML is an absolute requirement has emerged. Hearing devices is an area where tiny ML holds the potential to radically transform the functionality of a 1 mW always on device. People with hearing problems can benefit from hearing device processing that separates competing voices into individual channels followed by re-synthesizing of the auditory scene with spatial augmentation. The first successful segregation enhancement of competing voices required deep neural networks to achieve enough separation for the spatial augmentation to enhance segregation. It is furthermore a requirement that the latency of processing is below 20 ms – preferably less – and thus the processing must take place at the ear level without uplink and downlink latencies. Thus, for voice separation to work on the ears of people with hearing problems tiny ML is a necessity.
3) Thinking Big with Tiny ML: Low Power High Performance DNN Accelerators for Mobile and IoT Applications KAIST ICT Endowed Chair Professor, School of Electrical Engineering, KAIST: The artificial intelligence (AI) revolution is being widely spread even to the IoT with the help of 5G wireless communication. Compared to the Cloud-based or Edge-based AI applications, Internet-of-Things (IoT) applications require more autonomous, adaptive, and cooperative operations with extremely limited power, computing and memory resources without stable communication channels. AI, especially deep neural network (DNN), is the key technology to support such autonomy and adaptivity of the IoT machines in an unpredictable environment with limited available information. The IoT machines should contain not only inference but also training capabilities to adapt to environmental changes based on their experiences. Therefore, software and hardware co-optimization for DNN training is necessary for low-power and high-speed accelerators, in the same way it brought a dramatic increase in the performance of DNN inference accelerators. In addition, deep reinforcement learning (DRL) accelerators will be an essential part of the tide, showing a lot of benefits at making continuous decisions in an unknown environment, where labeled data is difficult to acquire.

D. Trusted Execution Environment : Data Security

Protecting applications and their data from unauthorized access by privileged system software is a long-standing research objective. Initial work such as NGSCB [19] and Proxos [20] executes untrusted and trusted OSs side by side using virtualization, with security-sensitive applications hosted by the trusted OS. Subsequent work, including Overshadow [21], InkTag [22] and Virtual Ghost [23], has focused on reducing the size of the TCB by directly protecting application memory from unauthorized OS accesses. SEGO [24] extends these approaches by securing data handling inside and across devices using trusted metadata. Minibox [25] is a hypervisor-based sandbox that provides two-way protection between native applications and the guest OS. Unlike SCONE [26], all of these systems assume a trusted virtualization layer and struggle to protect applications from an attacker with physical access to the machine or who controls the virtualization layer. Trusted hardware can protect security-sensitive applications, and implementations differ in their performance, commodification, and security functionality.

Secure co-processors offer tamper-proof physical isolation and can host arbitrary functionality. However, they are usually costly and limited in processing power. While in practice used to protect high-value secrets such as cryptographic keys demonstrate that secure co-processors can be used to split a database engine into trusted and untrusted parts. SCONE instead focuses on securing entire commodity container workloads and uses SGX to achieve better performance.

ARM TrustZone [27] has two system personalities, secure and normal world. This split meets the needs of mobile devices in which a rich OS must be separated from the system software controlling basic operations. Santos [28] use TrustZone to establish trusted components for securing mobile applications. However, isolation of mutually distrustful components requires a trusted language runtime in the TCB because there is only a single secure world. TrustZone also does not protect against attackers with physical DRAM access.

IV. Proposed Work

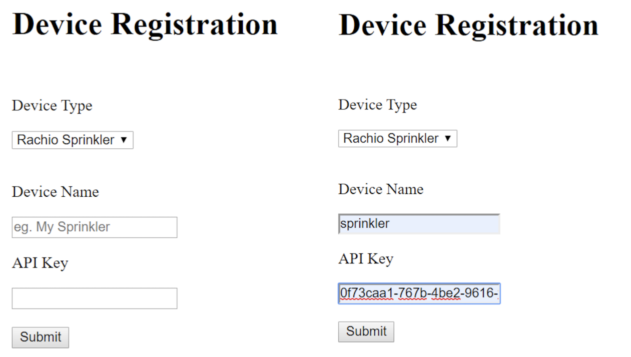
A. *Module 1: Secure Streaming IoT Data Management and Processing System with Multivariate IoT devices*

With the advent of enormous real time data, its highly important to build a dynamic system that could ease the way to store dynamic diverse data and process it securely. The proposed solution can be further categorized in two parts. The first part of the proposed solution provides a generic framework which is used to handle and store diverse content traversing from various IoT Devices and store them intuitively, so that it could be used for processing in further steps providing important analytic. The second part of the proposed solution presents a secure distributed IoT data analytics system with the help of TEEs to ensure the integrity and confidentiality of IoT data.

1) Proposed Solution, Part-1 : Multi-level Web-framework to Simplify Heterogeneous IoT Data Processing: The first proposed solution comprise of a multi-level web-framework which helps to register new IOT devices, collect raw data from it, allow users to dynamically select filtered metrics, which would be later considered for analysis. The system uses nosqlDB (MongoDB [29]) to store all users details, various IOT devices information, and the filtered metrics in encrypted form to be considered for secure analysis.

Below, we explain the web application architecture to filter metrics from IOT Devices:

- In first step the admin user registers a new device to the system using web interface. The related device details are stored in MongoDB. The snapshots of the solution with sample data is illustrated in Figure 1 and the stored data in MongoDB can be viewed in Figure 2. In this sample, we used Rachio Smart Sprinkler Controller [30] for an IoT device.



Device Registration

Device Type: Rachio Sprinkler ▾

Device Name: eg. My Sprinkler

API Key:

Submit

Device Registration

Device Type: Rachio Sprinkler ▾

Device Name: sprinkler

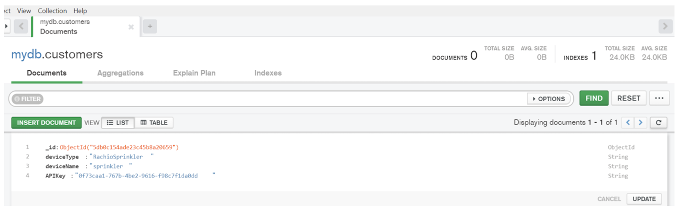
API Key: 0f73caa1-767b-4be2-9616-

Submit

(a) IoT device registration page sample

(b) IoT device registration page sample with provided data

Fig. 1: Snapshots of IoT Device Registration Page



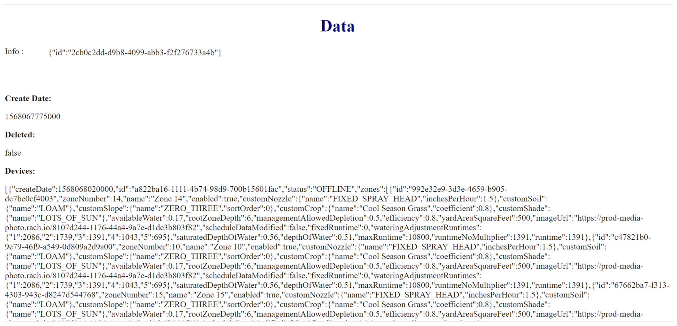
mydb.customers — DOCUMENTS 0, TOTAL SIZE 0B, AVG. SIZE 0B; INDEXES 1, TOTAL SIZE 24.0KB, AVG. SIZE 24.0KB

Displaying documents 1 - 1 of 1

```
_id: ObjectId("5db0c154ade23c45b8a20659")      ObjectId
deviceType  : "RachioSprinkler   "             String
deviceName  : "sprinkler  "                    String
APIKey  : "0f73caa1-767b-4be2-9616-f98c7f1da0dd     "   String
```

Fig. 2: IOT Device details in mongoDB.

- After registering the device, the users can view the registered new device detail in “Device Details” webpage. The user can request raw data from registered device to view at runtime. A sample snapshot of the raw data from Rachio Smart Sprinkler is shown in Figure 3.
- This allows user to visualize the data coming from various IOT devices at run-time in web application. Now, the web-app provides the functionality to users to select the required fields from web interface for important metrics to be considered from the raw data which will be later used for data analysis. The checkbox option mentioning all related fields of IOT data helps to ease the data filtration part and provides easy solution to get important fields of data from IOT devices. This is illustrated in Figure 4.



Data

Info : {"id":"2cb0c2dd-d9b8-4099-abb3-f2f276733a4b"}

Create Date:

1568067775000

Deleted:

false

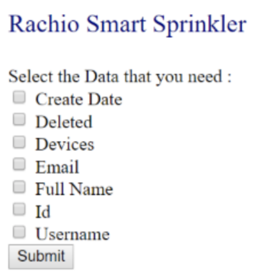
Devices:

[{"createDate":1568068020000,"id":"a822ba16-1111-4b74-98d9-700b15601fac","status":"OFFLINE","zones":[{"id":"992e32e9-3d3e-4659-b905-de7be0cf4003","zoneNumber":14,"name":"Zone 14","enabled":true,"customNozzle":{"name":"FIXED_SPRAY_HEAD","inchesPerHour":1.5},"customSoil":{"name":"LOAM"},"customSlope":{"name":"ZERO_THREE","sortOrder":0},"customCrop":{"name":"Cool Season Grass","coefficient":0.8},"customShade":{"name":"LOTS_OF_SUN"},"availableWater":0.17,"rootZoneDepth":6,"managementAllowedDepletion":0.5,"efficiency":0.8,"yardAreaSquareFeet":500,"imageUrl":"https://prod-media-photo.rach.io/8107d244-1176-44a4-9a7e-d1de3b803f82","scheduleDataModified":false,"fixedRuntime":0,"wateringAdjustmentRuntimes":{"1":2086,"2":1739,"3":1391,"4":1043,"5":695},"saturatedDepthOfWater":0.56,"depthOfWater":0.51,"maxRuntime":10800,"runtimeNoMultiplier":1391,"runtime":1391},{"id":"c47821b0-9e79-46f9-a549-0d809a2d9a00","zoneNumber":10,"name":"Zone 10","enabled":true,"customNozzle":{"name":"FIXED_SPRAY_HEAD","inchesPerHour":1.5},"customSoil":{"name":"LOAM"},"customSlope":{"name":"ZERO_THREE","sortOrder":0},"customCrop":{"name":"Cool Season Grass","coefficient":0.8},"customShade":{"name":"LOTS_OF_SUN"},"availableWater":0.17,"rootZoneDepth":6,"managementAllowedDepletion":0.5,"efficiency":0.8,"yardAreaSquareFeet":500,"imageUrl":"https://prod-media-photo.rach.io/8107d244-1176-44a4-9a7e-d1de3b803f82","scheduleDataModified":false,"fixedRuntime":0,"wateringAdjustmentRuntimes":{"1":2086,"2":1739,"3":1391,"4":1043,"5":695},"saturatedDepthOfWater":0.56,"depthOfWater":0.51,"maxRuntime":10800,"runtimeNoMultiplier":1391,"runtime":1391},{"id":"67662ba7-f313-4303-943c-d8247d544768","zoneNumber":15,"name":"Zone 15","enabled":true,"customNozzle":{"name":"FIXED_SPRAY_HEAD","inchesPerHour":1.5},"customSoil":{"name":"LOAM"},"customSlope":{"name":"ZERO_THREE","sortOrder":0},"customCrop":{"name":"Cool Season Grass","coefficient":0.8},"customShade":{"name":"LOTS_OF_SUN"},"availableWater":0.17,"rootZoneDepth":6,"managementAllowedDepletion":0.5,"efficiency":0.8,"yardAreaSquareFeet":500,"imageUrl":"https://prod-media-

Fig. 3: Raw data collected from the IoT device.

- After the filtered data is generated from user, it is stored in MongoDB in encrypted form, so that the system uses the filtered data to make any real time processing.
- Now any system can read the filtered data from MongoDB and pick the important fields for analysis related to IOT device.

The overall architecture of the system can be visualized as illustrated in Figure 5.

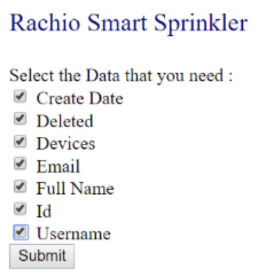


Rachio Smart Sprinkler

Select the Data that you need :

☐ Create Date
☐ Deleted
☐ Devices
☐ Email
☐ Full Name
☐ Id
☐ Username

(a) Fields from raw data to make intuitive selection



Rachio Smart Sprinkler

Select the Data that you need :

✓ Create Date
✓ Deleted
✓ Devices
✓ Email
✓ Full Name
✓ Id
✓ Username

(b) Filtered data after selection of important fields

Fig. 4: Snapshots of IoT data selection Page.

2) Proposed Solution, Part-2: Secure Stream Data Processing using Intel SGX: Intel's Software Guard Extensions (SGX) is Intel's latest iteration of a trusted hardware solution to the secure remote computation problem. The SGX design is centered around the ability to create an isolated container (Enclave), whose content receives special hardware protections that are intended to translate into confidentiality, integrity, and freshness guarantees. 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. Utilizing SGX's enclave features, we securely perform simple analytics on delicate IoT data, so that no unauthorized personnel can unlawfully access data or any analytical results.

To ensure end-to-end secure system, IoT devices and SGX use symmetric key encryption to communicate. We use one of the most popular and widely adopted symmetric key encryption algorithms Advanced Encryption Standard (AES) [31] in our framework. Once the enclave is initialized in the untrusted platform, 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 the symmetric key for encryption/decryption, to the enclave on the untrusted platform over a secure communication channel.

We collect data from IoT devices over HTTPS connection. The communication protocol of HTTPS is encrypted with Transport Layer Security (TLS) [32], or formerly known as Secure Sockets Layer (SSL). Hence, data in transit over HTTPS connection is always secure and eavesdropping on it is almost hopeless. After data is received through secure channel, it gets filtered with required fields only (discussed in previous part), converted to a unique HSML [33] format, which is very

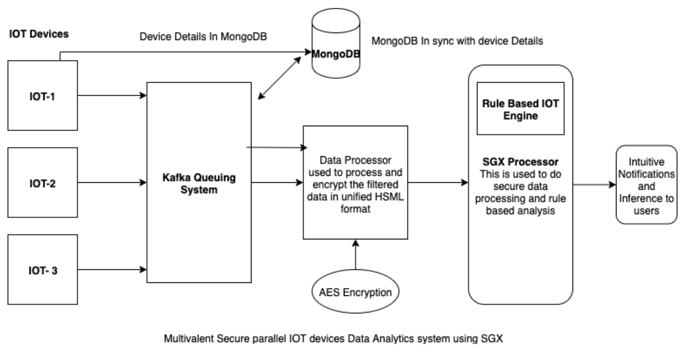

IOT Devices
Device Details In MongoDB
MongoDB
MongoDB In sync with device Details
IOT-1
IOT-2
IOT- 3
Kafka Queuing System
Data Processor used to process and encrypt the filtered data in unified HSML format
AES Encryption
Rule Based IOT Engine
SGX Processor This is used to do secure data processing and rule based analysis
Intuitive Notifications and Inference to users
Multivalent Secure parallel IOT devices Data Analytics system using SGX

Fig. 5: Secure IoT data analytic system using SGX

light weight. Then, it is immediately encrypted using AES algorithm with the symmetric key, which was already agreed upon with SGX enclave after the attestation process.

Next, this encrypted data is sent to Kafka system [34], which is a distributed stream processing platform that follows publish and subscribe methodology to streams of records. Kafka is fault-tolerant, exceptionally fast, and used for building streaming applications that handle streams of data, and real-time data pipelines. Next, from Kafka data reaches SGX, which is on an untrusted cloud platform. The data is decrypted only inside SGX enclave, which is the trusted environment of the SGX, where the symmetric key is safely stored. Once the data is decrypted, we perform some decision making on the data based on the application. In our framework, the logic code will be safely stored inside the SGX enclave, thus preventing the attacker from manipulating the code or the value. Furthermore, as the values are always encrypted outside of enclave, the attacker cannot access those values as well, making the mechanism pretty secure and robust.

Our framework also handles sending the decision back to the IoT devices from the SGX depending on the user requirement. For this, decision data is encrypted with the same encryption key inside the enclave and then passed to the untrusted part of the machine. The cloud server machine sends this data back to the controller, which is essentially a gateway that contains information of the IoT devices. The controller eventually takes care of transmitting the decision to the particular IoT device after decryption. Hence, in our system data stays encrypted always in transmission for both ways and also in rest with the help of SGX.

B. *Module 2: UAV Data Processing System on edge and device itself*

As discussed in Section I, with the advent of more IoT applications, it also demand fast real-time data processing with minimal latency. As a result, researches have been conducted to process data in the IoT devices itself. TinyML [3] proposes a solution of running neural network models in microcontrollers, which consumes only a few milliwatts of power and only kilobytes of memory. This gives us the opportunity to use the deep learning models in the IoT devices itself and perform real-time anaytics with low latency. Here, we envision a UAV-

Model Name	Model size	mAP (VOC 2007)	computational cost (ops)
Tiny YOLOv2 [13]	60.5MB	57.1%	6.97B
Tiny YOLOv3 [14]	33.4MB	58.4%	5.52B
YOLO Nano	**4.0MB**	**69.1%**	**4.57B**

TABLE I: Yolo Nano comparison results: Object detection accuracy results of tested compact networks on VOC 2007 test set. [35]

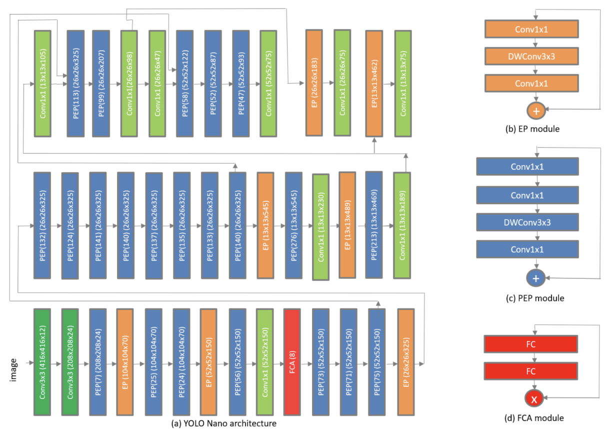


Fig. 6: Yolo Nano Architecture [35]

based real-time data processing system with TinyML. More specifically, we propose a framework that will use TinyML to perform real-time image processing captured by a drone with the help of an embedded device mounted on the drone. As an example, our framework will try to detect suspicious or abnormal behaviors of ground vehicles by capturing images from the drone and processing it with machine learning model in the embedded device attached to the drone or on the edge.

Traffic monitoring is typically very dynamic and requires continuous and accurate monitoring systems. Conventional traffic surveillance relies on a set of fixed cameras or other detectors, requiring a high density of the said devices in order

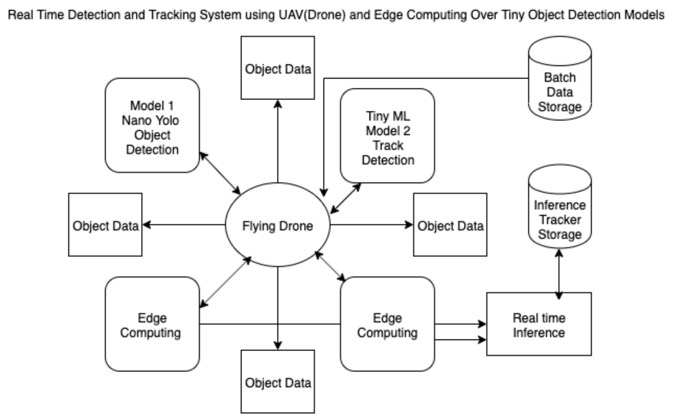

Real Time Detection and Tracking System using UAV(Drone) and Edge Computing Over Tiny Object Detection Models
Object Data
Batch Data Storage
Model 1 Nano Yolo Object Detection
Tiny ML Model 2 Track Detection
Object Data
Flying Drone
Object Data
Inference Tracker Storage
Edge Computing
Edge Computing
Real time Inference
Object Data

Fig. 7: Real-time detection and tracking system using UAVs (Drone) and edge computing over tiny object detection models

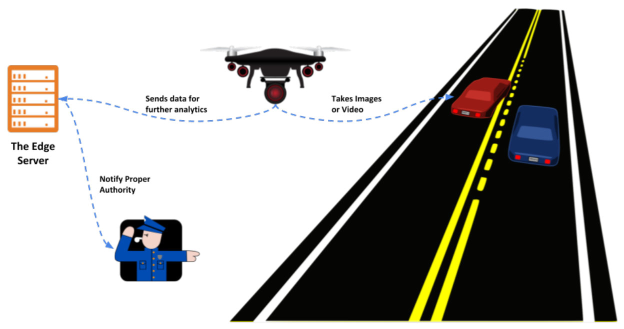


Fig. 8: Abnormal Vehicle Trajectory Detection using UAVs.

to monitor the intersection in its entirety and to provide data insufficient quality. Alternatively, a UAV can be converted to a very agile and responsive mobile sensing platform for data collection from such large scenes. UAVs can be easily deployed anywhere, especially high traffic zone areas, and images captured from these drones are analyzed for abnormal vehicle trajectory. Recent study shows that, alcohol-impaired drivers are behind the wheel more than 300,000 times every day, however, only around 2,800 are arrested [36]. This low number of arrests could be because of lack of proper traffic monitoring system, which fails to notify local authorities in real-time.

As a consequence, we envision to propose a framework to detect unconventional vehicle trajectory with UAVs. The UAVs will take images of the vehicles moving in the street and these images will be used to carry out real time object tracking. The tracking will help to detect the abnormal driving behavior on the edge using object detection using nano yolo [35]. The nano yolo is miniature deep learning model for object detection and can be used to track vehicles on the edge for low resolution image system. The architectural view of yolo nano can be visualized in Figure 6 and its comparison in terms of model size, MAP, and computational cost with other tiny models is shown in Table I. This can help to perform vehicle trajectory detection which can be used to foresee abnormal driving behaviour and provide related notification to nearby authorities. The overall architecture and process is illustrated in Figures 7 and 8. As the object tracking is computationally expensive procedure, it can also be processed on the edge with today's edge computing power to minimize the latency. However, running the procedure in the device itself is a challenging task, which will be an interesting problem to solve in the near future, as it will give the best real-time performance to this time sensitive problem. In addition, to process the detection procedure securely, we could utilize our envisioned module IV-A for any sensitive computation.

V. Conclusion

As the usage of IoT devices increase, preserving data privacy and processing substantial amount of data in real-time becomes more crucial. This paper envisions two solutions that provides secure data analytics system as well as a real-time data analytics system. Our secure data processing system utilizes Intel SGX to perform simple analytics that ensures data integrity and confidentiality. Moreover, strong encryption mechanism guarantees data privacy in transit, making the system end-to-end encrypted. Also, it handles the heterogeneity of data from various IoT devices. On the other hand, our drone-based real-time data analytics system proposes to perform data analytics on the edge or on the device itself to significantly reduce latency. Although, depending on the application, this could be computationally expensive for such devices, the recent development of running machine learning models on tiny embedded devices has opened up doors of many new opportunities.

ACKNOWLEDGMENT

This material is based upon work supported by the National Science Foundation under Award No. DMS-1737978, DGE-1723602, DGE-1931800, SMA-1539302 and OAC-1828467. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation."

REFERENCES

[1] F. Paul, "Top 10 iot vulnerabilities," https://www.networkworld.com/article/3332032/top-10-iot-vulnerabilities.html, retrieved: 2019-08-24.

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

[3] D. Situnayake and P. Warden, *TinyML*, 1st ed. O'Reilly Media, Inc., 2019.

[4] S. Jeong, O. Simeone, and J. Kang, "Mobile edge computing via a uav-mounted cloudlet: Optimization of bit allocation and path planning," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 3, pp. 2049–2063, 2017.

[5] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the internet of things," in *Proceedings of the first edition of the MCC workshop on Mobile cloud computing*. ACM, 2012, pp. 13–16.

[6] Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, and V. Young, "Mobile edge computing—a key technology towards 5g," *ETSI white paper*, vol. 11, no. 11, pp. 1–16, 2015.

[7] Kubernetes, "Kubernetes: Production-grade container orchestration," https://kubernetes.io, retrieved: 2019-11-22.

[8] P. Mell, T. Grance *et al.*, "The nist definition of cloud computing," 2011.

[9] KubeEdge, "Kubeedge: An open platform to enable edge computing," https://kubeedge.io/, retrieved: 2019-11-22.

[10] W. Mazzarella, *Censorium: Cinema and the open edge of mass publicity*. Duke University Press, 2013.

[11] A. Haque, L. Khan, and M. Baron, "Sand: Semi-supervised adaptive novel class detection and classification over data stream," in *THIRTIETH AAAI Conference on Artificial Intelligence*, 2016.

[12] 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.

[13] 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.

[14] 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, p. 16, 2011.

[15] 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.

[16] 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.

[17] M. Masud, L. Khan, and B. Thuraisingham, *Data mining tools for malware detection*. Auerbach Publications, 2016.

[18] 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.

[19] M. Peinado, Y. Chen, P. England, and J. Manferdelli, "Ngscb: A trusted open system," in *Australasian Conference on Information Security and Privacy*. Springer, 2004, pp. 86–97.

[20] Y. Reich, "Exemplar-based knowledge acquisition," 1991.

[21] X. Chen, T. Garfinkel, E. C. Lewis, P. Subrahmanyam, C. A. Waldspurger, D. Boneh, J. Dwoskin, and D. R. Ports, "Overshadow: a virtualization-based approach to retrofitting protection in commodity operating systems," *ACM SIGOPS Operating Systems Review*, vol. 42, no. 2, pp. 2–13, 2008.

[22] O. S. Hofmann, S. Kim, A. M. Dunn, M. Z. Lee, and E. Witchel, "Inktag: Secure applications on an untrusted operating system," in *ACM SIGPLAN Notices*, vol. 48, no. 4. ACM, 2013, pp. 265–278.

[23] J. Criswell, N. Dautenhahn, and V. Adve, "Virtual ghost: Protecting applications from hostile operating systems," in *ACM SIGPLAN Notices*, vol. 49, no. 4. ACM, 2014, pp. 81–96.

[24] Y. Kwon, A. M. Dunn, M. Z. Lee, O. S. Hofmann, Y. Xu, and E. Witchel, "Sego: Pervasive trusted metadata for efficiently verified untrusted system services," *ACM SIGPLAN Notices*, vol. 51, no. 4, pp. 277–290, 2016.

[25] Y. Li, J. McCune, J. Newsome, A. Perrig, B. Baker, and W. Drewry, "Minibox: A two-way sandbox for x86 native code," in *2014 {USENIX} Annual Technical Conference ({USENIX}{ATC} 14)*, 2014, pp. 409–420.

[26] S. Arnautov, B. Trach, F. Gregor, T. Knauth, A. Martin, C. Priebe, J. Lind, D. Muthukumaran, D. O'Keeffe, M. L. Stillwell *et al.*, "{SCONE}: Secure linux containers with intel {SGX}," in *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, 2016, pp. 689–703.

[27] J. Winter, "Trusted computing building blocks for embedded linux-based arm trustzone platforms," in *Proceedings of the 3rd ACM workshop on Scalable trusted computing*. ACM, 2008, pp. 21–30.

[28] N. Santos, H. Raj, S. Saroiu, and A. Wolman, "Using arm trustzone to build a trusted language runtime for mobile applications," in *ACM SIGARCH Computer Architecture News*, vol. 42, no. 1. ACM, 2014, pp. 67–80.

[29] K. Chodorow and M. Dirolf, *MongoDB: The Definitive Guide*, 1st ed. O'Reilly Media, Inc., 2010.

[30] R. inc., "Rachio 3," https://www.rachio.com/rachio-3/, retrieved: 2019-11-22.

[31] J. Daemen and V. Rijmen, "Aes proposal: Rijndael," 1999.

[32] Z. Kerravala, "What is transport layer security (tls)?" https://www.networkworld.com/article/2303073/lan-wan-what-is-transport-layer-security-protocol.html, retrieved: 2019-08-24.

[33] A. Mazayev, J. A. Martins, and N. Correia, "Semantically enriched hypermedia apis for next generation iot," in *Interoperability, Safety and Security in IoT*. Springer, 2017, pp. 19–26.

[34] T. A. S. Foundation, "Apache kafka," https://kafka.apache.org, retrieved: 2019-08-14.

[35] A. Wong, M. Famuori, M. J. Shafiee, F. Li, B. Chwyl, and J. Chung, "Yolo nano: a highly compact you only look once convolutional neural network for object detection," *arXiv preprint arXiv:1910.01271*, 2019.

[36] T. Arevalo, "30 harrowing drunk driving statistics – 2019," https://carsurance.net/blog/drunk-driving-statistics, retrieved: 2019-10-24.