2019 IEEE International Congress on Internet of Things (ICIOT)

# Profiling-based classification algorithms for security applications in Internet of Things

*Eunil Seo, †Hyoungshick Kim, and ‡Tai-Myoung Chung
*Department of electrical and computer Eng., Sungkyunkwan University, Suwon, Korea
†‡College of software, Sungkyunkwan University, Suwon, Korea
{*seoei2, †hyoung, ‡tmchung}@skku.edu

***Abstract*—Due to the various types of network resources involved in the Internet of Things (IoT), it becomes challenging to detect security incidents and unexpected faults in IoT environments. The nature of network objects (e.g., system, user, service, and devices) is too various and changeable to predict objects' behaviors and to identify the best parameters for the machine learning model in order to detect anomalies against IoT protection. We propose a new profiling method called "Management Information Base for IoT (MIB-IoT)" by extending conventional MIB to a more generalized structure in order to represent not only the structured properties of network objects but also the best machine learning model for each network object in a systematic fashion. MIB-IoT profiles can be defined for various applications such as abnormal behavior detection, malicious behavior detection, and even data source identification. To demonstrate the feasibility of the proposed MIB-IoT, we apply various classification algorithms on datasets consisting of normal operation data, hardware fault data, and malicious data. The experiment results show that the classification algorithm using MIB-IoT is capable of achieving an accuracy of $99.81\%$ for malicious behavior detection and an accuracy of $78.51\%$ for data source identification respectively.**

***Keywords*-Machine learning, classification, abnormal behavior detection, Internet of Things (IoT), Management Information Base (MIB).**

## I. Introduction

The systems involved in the Internet of things (IoT) network typically consist of numerous devices equipped with a variety of computing, power, sensor, and communication capabilities. Therefore, it is very difficult to conceptualize a universally working machine learning (ML) model across various IoT devices [1]. The features used in models and their behaviors can dynamically be changed with devices and applications.

To address this issue, we must consider the use of profiles to represent various IoT devices and applications as opposed to building a single universal ML model [2]. Specifically, we propose a new profiling method called "Management Information Base for IoT (MIB-IoT)" by extending conventional MIB [3] to a more generalized structure in order to systematically represent not only the structured properties of network objects but also the best ML model for each network object. Profiling is a useful technique for analyzing just a few samples when the samples can be grouped based on their different characteristics and behaviors [4].

As security attacks have evolved in complexity over time, naive rule-based detection techniques are becoming increasing limited in detecting new types of attacks. Our proposed profiling method is designed to incorporate various ML algorithms to better detect such new security attacks as well as data faults along with data sources in the IoT environment. Our contributions are summarized as follows:

1) To the best of our knowledge, this study is the first to consider the flexible structure of MIB-IoT involving ML algorithms in order to detect malicious behavior and identify data source in the IoT environment.
2) We presented how to build an MIB-IoT from selected ML algorithms against a service compromising regarding to HVAC (Heating,Ventilation, and Air Conditioning).
3) To show the feasibility of the proposed idea, we evaluated the performances of profiling-based classification algorithms with datasets consisting of a normal operation dataset, a hardware fault dataset, and a malicious dataset collected from 54 real-world sensors [5]; we also examined research on multiple ML algorithms for malicious behavior detection [6] and data source identification [7]. The experimental results indicate that our classification algorithm using MIB-IoT is capable of achieving an accuracy of $99.81\%$ for malicious behavior detection (three classes) and an accuracy of $78.51\%$ for data source identification (54 classes), respectively.

The remainder of this paper is organized as follows. In Section II, we discuss background information and related work. In Section III, we present the structure of MIB-IoT for profiling IoT devices and applications. In Section IV, we show the effectiveness of MIB-IoT based classification algorithms for malicious behavior detection and data source identification. In Section V, we introduce the MIB-IoT profiles to represent ML models for malicious behavior detection and data source identification. Finally, conclusions and further work are provided in Section VI.

## II. Background

In this section, we explain the basic knowledge and terms that are required to understand the proposed system.

### A. Profiling

Automated profiling involves different technologies (i.e., hardware), such as RFID-tags [8], biometrics, sensors, and computers, along with different techniques (i.e., software), such as data cleansing, data aggregation, and data mining. These technologies and techniques are integrated into profiling practices that allow for both the construction and application of the profiles; these profiles are then used to make

978-1-7281-2714-9/19/$31.00 ©2019 IEEE
DOI 10.1109/ICIOT.2019.00033

decisions. Achieving the purpose of the ambient intelligence or ubiquitous networked environments depends entirely on autonomic profiling, the type of profiling that allows machines to communicate with other machines and make decisions without human intervention [9]. MIB-IoT of IoT devices is the integrated classifier that applies multiple ML algorithms to detect anomalous behavior and predict threats for IoT device protection [10].

### B. Threats and attack vectors in IoT

In IoT environments, various attack vectors exist such as device tampering, privacy breach, denial of service, spoofing, elevation of privilege, signal injection, and side-channel [11]. In this work, we aim to detect malicious data, which can compromise an IoT service (e.g., air conditioning) while the service functionality is still working.

### C. Anomaly detection

According to the attack vectors in IoT, two types of attacks should be considered: (1) service attack and (2) device attack.

*1) Service attack detection:* One of the approaches that can be used to compromise a service involves manipulating the data for a service that does not work properly [12]. For instance, if the temperature that is transmitted to the air-conditioning controller is lower than the real value, then the HVAC system will not cool down an office, even in hot weather. In this work, we tested the proposed system using MIB-IoT based on a sensor dataset collected from Intel Berkeley Research Lab [13]. We assume that a service may behave abnormally when the sensing values transmitted by the sensor are intentionally modified by a malicious attacker and the sensing values significantly differ from the normal sensing values.

Lee et al. [14] proposed the concept of IoT profiles to detect abnormal behavior. To develop IoT profiles in an automated manner, a clustering algorithm (e.g., $K$-means and support vector machine (SVM)) was used. They tested a proof-of-concept implementation using IoT profiles with two datasets related to faults: one-type fault of the four data types (e.g., temperature) and four-types faults (temperature, humidity, light, and voltage) (see study [15] for more detailed descriptions of the datasets). The IoT profile is built into a two-dimensional space by using a clustering algorithm (e.g., $K$-means and SVM) after transforming four attributes to two attributes through Principal Component Analysis (PCA). However, this method of forming an IoT profile is limited to applications to certain types of data, because the IoT profile is a group of normal operation data in two-dimensional space. To address this issue, we create IoT profiles to represent multiple ML algorithms without the limitation of dimension space. In addition, we extend the existing MIB structure for IoT environments, as MIB is already popularly used in the network industry.

*2) Device attack detection:* Ahmed et al. [16] proposed a method to use applications' network packets at the transport layer and flow-level features as their fingerprints. The normal profile generation was carried out by well-known features gathered by the transport layer packet-level, and the normal profile updating was conducted by other features extracted by the flow-level. The profile of the normal application is based on a multi-modal probability distribution. In order to detect distributed denial of service attacks, the application classification framework can be extended from the flow-level statistical information. However, it is not easy to export or provide the trained model or information to the requiring user even though its approach and results are significant. To overcome this issue, we can use the proposed MIB-IoT profiles to represent the trained ML models and transfer them to other users.

## III. Structure and components of MIB-IoT

In IoT environments, network devices can be exposed to various types of threats, which lead to significant damages (e.g., devices' abnormal behavior, hardware fault, data fault, etc.). Therefore, security solutions must be developed to mitigate such threats. Recently, a promising technique is to involve ML models against a large amount of data, which include suspicious ones. In this paper, we suggest a profiling method called MIB-IoT to represent ML models and share them with other users in a flexible and efficient manner. The proposed method can be specifically useful for representing multiple ML algorithms to improve the performance of threat detection systems.

### A. Structure of MIB-IoT

In this work, MIB-IoT is based on the conventional MIB, as shown in Fig. 1. MIB, which is a tree-structured hierarchical database, is used to manage and control the network entities for the communication network. Each entity has an address denoted by an object identifier (OID). RFC 1155 [3] and RFC 1213 [17] describe the structure and identification of an MIB entity, respectively. As shown in Fig. 1, until enterprise entity, its OID is a pre-defined one such as 1.3.6.1.4.1, and the other private companies can obtain their own unique OIDs by request. Consequently, private companies can specify their own private MIBs to their own products and systems by themselves, like the MIB-IoT proposed in this work, shown in Fig. 2.

We extend the existing MIB into MIB-IoT to represent additional properties for IoT environments. An example of MIB tree extended to MIB-IoT is shown in Fig 1. HVAC(1).MIB-IoT(1) is dedicated to the HVAC system of a company with the ID of 1. The HVAC(1).MIB-IoT(1) contains dataset information and ML training information provided by the ML algorithms for particular purposes, such as malicious behavior detection, data fault detection, and data source identification, as shown in Fig. 2.

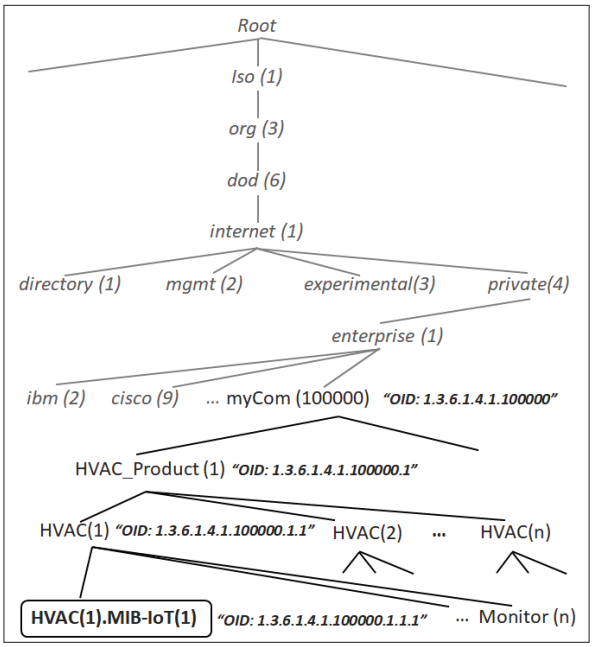



Fig. 1. MIB-IoT (e.g., HVAC(ID).MIB-IoT(1)) along with the MIB tree.

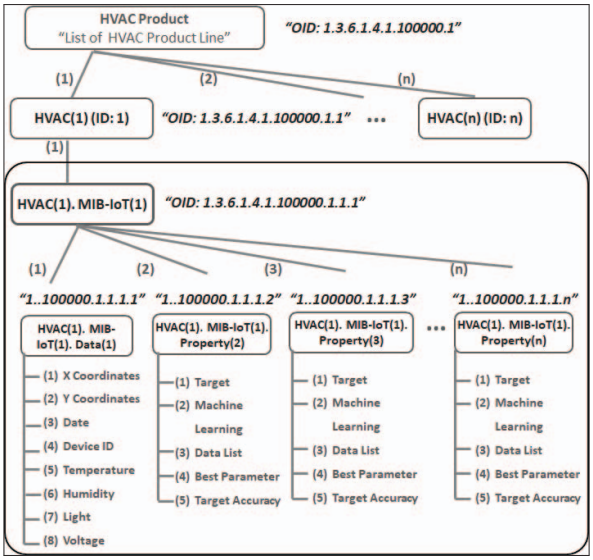



Fig. 2. Components of MIB-IoT: MIB-IoT(1).Data(1), MIB-IoT(1).Property(2), MIB-IoT(1).Property(3), etc.

### B. Components of MIB-IoT

In IoT network environment, many objects can be profiled with a dataset, which can be used to detect abnormal behavior as well as identify the data source. Such an object can be a service, a system, a user, a behavior, etc. In this work, we build an MIB-IoT of a HVAC system to detect abnormal behavior from data transmitted by 54 sensors in an office of the Intel Berkeley Research Lab [13]. These sensors provide the following data: temperature, humidity, light, and voltage. The HVAC system operates according to the data obtained by these sensors. However, the transmitted data can be unreliable, due to the easily fragile nature of the sensors as well as the potential actions of malicious attackers.

To justify HVAC(1), we create HVAC(1).MIB-IoT(1) based on the transmitted data as shown in Fig 2. As a result, we can classify data as either normal operation data, hardware faulty data, or malicious data, and we can also identify the data source through the sensor ID. Consequently, only normal operation data can be set to be used to control and manage the HVAC system in an office. In this work, each dedicated MIB-IoT corresponds to a system or a service: HVAC(1), HVAC(2), etc. In other words, each system has its own multiple profiles: HVAC(1).MIB-IoT(1).Property(2), HVAC(1).MIB-IoT(1).Property(3), etc. with a common HVAC(1).MIB-IoT(1).Data(1) as shown in Fig. 2.

**HVAC(ID).MIB-IoT(1):** The HVAC profile is built with each ID. Each system, HVAC, has its own profile consisting of an dataset and multiple properties based on MIB as follows:

**HVAC(ID).MIB-IoT(1)** OBJECT-TYPE
SYNTAX MIB-IoT(1).Data(1) and
sequence of MIB-IoT(1).Property(n)
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION "MIB-IoT of system, service, etc."
INDEX { MIB-IoT(1)ofHVAC(1) }
::= { HVAC(ID) 1}

where the profile Object Identification (OID) is "1.3.6.1.4.1. 100000.1.1.1", as shown in Fig 2. It can be imported to each system under private enterprise along the MIB tree.

**HVAC(ID).MIB-IoT(1).Data(1):** It has information on the dataset for the ML training as follows:

**MIB-IoT(1).Data(1)** OBJECT-TYPE
SYNTAX SEQUENCE of DataEntry
MAX-ACCESS read-only
STATUS current
DESCRIPTION "Data Table of dataset for ML training"
::= { HVAC(ID).MIB-IoT(1) 1}

MIB-IoT(1).Data(1) means the dataset for ML training and it is a subset of HVAC(ID).MIB-IoT(1). Along the MIB tree, the Object Identification (OID) is 1.3.6.1.4.1.100000.1.1.1.1. MIB-IoT(1).Data(1) consists of entries of the data table. The data table entries are in the following form:

**MIB-IoT(1).DataEntry** OBJECT-TYPE
SYNTAX DataEntry
MAX-ACCESS read-only
STATUS current
DESCRIPTION "Data Entry of table for the ML training"
INDEX {ID, Class} ::= { MIB-IoT(1).Data 1}

where DataEntry consists of the data used for ML training purposes, such as sensor ID, temperature, light, voltage of sensors, etc., as follows:

**DataEntry** ::=
SEQUENCE {

X Coordinates Integer32,
Y Coordinates Integer32,
Date DateAndTime,
ID Integer32,
Temperature float32,
Voltage, float32,
Light Integer32,
Humidity Integer32
}

**HVAC(ID).MIB-IoT(1).Property(2):** This has information on the MIB-IoT(1) property related to the training ML algorithm as follows:

**MIB-IoT(1).Property(n)** OBJECT-TYPE
SYNTAX PProperty
MAX-ACCESS read-only
STATUS current
DESCRIPTION "MIB-IoT Property"
::= { HVAC(ID).MIB-IoT(1) 2}

where PProperty consists of the properties of ML training information, such as the name of the ML algorithm with the best training parameters, as follows:

**PProperty** ::=
SEQUENCE {
TargetName (e.g., malicious behavior detection, etc.),
MachineLearning (e.g., Random Forest, etc.),
DataList (e.g.,temperature, ID, etc.),
MLparameter (e.g., estimater, etc.),
TargetAccuracy (e.g., $98.51\%$, etc.)
}

where TargetName indicates the purpose of the ML model, such as malicious behavior detection, data source identification, etc. MachineLearning represents the ML algorithms used, such as Random Forest, MLP, SVM, Gradient Boosting, Ridge Regression and KNN [18]. DataList represents the list of data attributes such as temperature, humidity, sensor ID, etc. MLparameter represents the best parameters of ML algorithms such as estimators: 30, learning rate: 0.1, etc. Finally, TargetAccuracy indicates the accuracy rate.

## IV. Case studies

Six models for malicious behavior detection and data source identification are built using one of six ML algorithms (e.g., Random Forest, MLP, SVM, KNN, Gradient Boosting, and Ridge Regression) for each model, and these are evaluated in terms of detection rate and identification rate; furthermore, those models are trained with a dataset consisting of normal operation data, hardware fault data, and malicious data, as provided by 54 sensors.

### A. Data acquisition

To serve people in the IoT environment, a larget number of smart sensors, actuators, meters, and cloud servers should be connected and operating [19]. From the system's point of view, two transactions, input and output, are fundamental for completing a service. In this section, we demonstrate an MIB-IoT of the HVAC system in order to show a practical method for detecting malicious behavior and hardware fault data, which lead to a compromising service. For an MIB-IoT of the HVAC system, we utilize the dataset of 54 sensors provided the Intel Berkeley Research Lab [5]. Each data entry has X coordinates, Y coordinates, device ID (1 to 54), date, temperature value, humidity value, light illumination value, and voltage value, as shown in Table I.

TABLE I
Eight attributes of the dataset provided by Intel Berkeley Research Lab [5]

| X | Y | ID | Date | Temp | Humidity | Light | Volt |
|---|---|---|---|---|---|---|---|

Based on the 54 deployed sensors, the dataset contains a total of over 2 million data entries, and after eliminating all of the null values, 2,210,084 data entries are utimately used for ML training in this work.

### B. Three feature datasets: $D^{Normal}$, $D^{Hardware}$, and $D^{Malicious}$

We briefly describe the terminology and notation relevant to the dataset in Table II.

TABLE II
Notation for dataset.

| Notations | Definition |
|---|---|
| $D^{Intel\ Lab}$ | Dataset provided by Intel Lab |
| $D^{Normal}$ | Dataset during normal operation |
| $D^{Hardware}$ | Dataset under hardware fault circumstance |
| $D^{Malicious}$ | Dataset involving modified temperature value |
| $D_{ID}$ | Dataset involving Sensor ID |
| $D_i$ | $i^{th}$ Data entry from Dataset |
| $\mathbb{X}_i$ | Input data entry for ML training |
| $y_i$ | Output data vector for ML Training |

The dataset of $D^{Intel\ Lab}$ as measured from 54 sensors consists of 2,210,084 data entries in total. Each entry has eight attributes: X coordinates, Y coordinates, ID (from 1 to 54), date, temperature, humidity, light, and voltage, as follows:

- $D^{Intel\ Lab} = \{\ D_1^I,\ D_2^I,\ D_3^I,\ \ldots,\ D_n^I\ \}$, where $n$ is 2,210,084.
- $D_i^I = (\mathbb{X}_i, y_i)$
- $\mathbb{X}_i = (x_{i1}, x_{i2}, \ldots, x_{id})$, where d is eight, representing the eight attributes: X coordinates, Y coordinates, ID, date, temperature, humidity, light, and voltage, as shown in Table I.
- $y_i = (y_i)$, where $y_i$ can represent the $Data\ Feature$ for malicious behavior detection as well as the sensor ID for data source identification.

To build MIB-IoT for malicious behavior detection and data source identification, we need to label the provided dataset as one of three potential classifications: $D^{Normal}$, $D^{Hardware}$, and $D^{Malicious}$. $D^{Normal}$ is classified by the conditions described in Algorithm 1, and $D^{Normal}$ is specified as follows:

- $D^{Normal} = \{D_1^N, D_2^N, D_3^N, \ldots, D_n^N\}$ where $n$ presents 1,603,477 dataset entries out of the original 2,210,084 after removing hardware fault data.
- $D_i^N = (\mathbb{X}_i, y_i)$ where $D_i^N$ is the $i^{th}$ data entry of the dataset, $\mathbb{X}_i$ is the vector of the input data, and $y_i$ is the output data: normal operation data as a data feature.
- $\mathbb{X}_i = (x_{i1}, x_{i2}, \ldots, x_{id})$ where d is four representing the four attributes: temperature, humidity, light, and voltage as shown in Table III.
- $y_i = (y_i)$ where $y_i$ has one output and $y_i$ represents the feature of data entry as a normal operation data as shown in Table III.

TABLE III
SIX ATTRIBUTES OF NORMAL OPERATION DATA ENTRY.

| Temp | Humidity | Light | Volt | ID | Data Feature: normal operation data |
|---|---|---|---|---|---|

**Algorithm 1:** Normal operation data extraction.

**Data:** normal operation dataset
**Result:** $D^{Normal}$
/* After importing the provided dataset, normal operation data are extracted with following conditions */
**begin**
  $Data^{Intel\ Lab} = pd.read\_csv(Dataset.csv)$
  $Data^{Normal} = Data^{Intel\ Lab}.query(temperature > 0\ and\ temperature < 40)$
  $Data^{Normal} = Data^{Normal}.query(humidity > 4\ and\ humidity < 80)$
  $Data^{Normal} = Data^{Normal}.query(light < 900)$
  $Data^{Normal} = Data^{Normal}.query(voltage > 1.8)$
**end**

The provided dataset has abnormal values, such as a high temperature of 122 Celsius, and this hardware fault data is caused by the sensor battery. $D^{Hardware}$ is specified as follows:

- $D^{Hardware} = \{D_1^H, D_2^H, D_3^H, \ldots, D_n^H\}$ where $n$ represents 606,607 dataset entries out of 2,210,084.
- $D_i^H = (\mathbb{X}_i, y_i)$ where $D_i^H$ is the $i^{th}$ data entry of the dataset, $\mathbb{X}_i$ is the vector of the input data, and $y_i$ is the output data: hardware fault data.
- $\mathbb{X}_i = (x_{i1}, x_{i2}, \ldots, x_{id})$ where $d$ is four representing the four attributes: temperature, humidity, light, and voltage as shown in Table IV.
- $y_i = (y_i)$ where $y_i$ has one output and $y_i$ represents the feature of data entry as a hardware fault data, as shown in Table IV.

TABLE IV
SIX ATTRIBUTES OF HARDWARE FAULT DATA ENTRY.

| Temp | Humidity | Light | Volt | ID | Data Feature: hardware fault data |
|---|---|---|---|---|---|

To generate a malicious dataset, we assume a service compromising scenario in which a malicious attacker purposefully manipulates the temperature value on purpose. In order to emulate such a scenario, one of the data inputs, temperature, is changed from that in the $D^{Normal}$. According to the measured temperature scope (0 - 40 Celsius) during normal operation, only temperatures between 30 and 40 are randomly decreased by a range of 0 to 5 for ten days out of a total of thirty eight days between February 28th and April 5th, 2004. In this sense, the HVAC system does not operate properly when the office temperature is between 30 to 40 for the 10 days chosen by a malicious attacker. The malicious dataset, $D^{Malicious}$, is specified as follows:

- $D^{Malicious} = \{D_1^M, D_2^M, D_3^M, \ldots, D_n^M\}$, where $n$ represents 523,591 dataset entries, which are modified data out of 1,603,477 normal operation data for the purpose of compromising a service.
- $D_i^M = (\mathbb{X}_i, y_i)$ where $D_i^M$ is the $i^{th}$ data entry of the dataset, $\mathbb{X}_i$ is the vector of the input data, and $y_i$ is the output data: malicious data.
- $\mathbb{X}_i = (x_{i1}, x_{i2}, \ldots, x_{id})$, where $d$ is four representing the four attributes: temperature, humidity, light, and voltage as shown in Table V.
- $y_i = (y_i)$, where $y_i$ represents the feature of data entry as malicious data as shown in Table V.

TABLE V
SIX ATTRIBUTES OF MALICIOUS DATA ENTRY.

| Temp | Humidity | Light | Volt | ID | Data Feature: malicious data |
|---|---|---|---|---|---|

### C. MIB-IoT for malicious behavior detection

To determine which ML algorithm is the best for detecting malicious behavior from the sensor dataset, we employ six supervised ML algorithms: $K$-Nearest Neighbors (KNN), Ridge Regression, Random Forest classifier (Random Forest), Gradient Boosting Regression Tree (Gradient Boosting), Support Vector Machine (SVM), and Multi-layer Perceptron classifier (MLP). In this work, six models are built by corresponding ML algorithms based on three datasets: $D^{Normal}$, $D^{Hardware}$, and $D^{Malicious}$.

To model MIB-IoT for malicious behavior detection, five attributes are used: temperature value (Temp), humidity value (Humidity), light value (Light), voltage Value (Volt), and data features (normal operation data, hardware fault data, and malicious data), as shown in Table VI.

TABLE VI
FIVE ATTRIBUTES FOR MALICIOUS BEHAVIOR DETECTION.

| Temp | Humidity | Light | Volt | Data Feature |
|---|---|---|---|---|

To evaluate the six models, we use a provided function, ShuffleSplit (test_size=.4, train_size=.6, $n$_splits = 10) with the three datasets. This means that each dataset is split into 10 groups with five rounds: $60\%$ of dataset is used for the training set while $40\%$ of the dataset is used for the testing

set, as shown in Fig 3. The dataset is shuffled randomly prior to the data split.

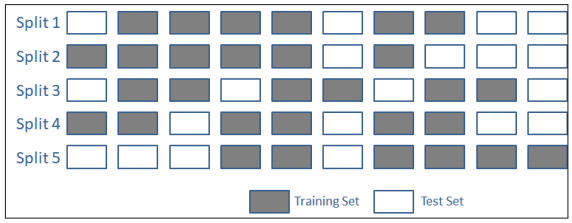



Fig. 3. Shuffle split dataset with the 60% training set and 40% test set.

In algorithm 2, for malicious behavior detection, we import six ML algorithms with $ML\_object = ML\_Algorithms\ (random\_state = 0)$ after the data split.

**Algorithm 2:** ML modeling for malicious behavior detection with six ML algorithms.

**Data:** Three datasets: $D^{Normal}$, $D^{Hardware}$, and $D^{Malicious}$

**Result:** $Detection\_Accuracy$

```
/* Data are separated into train & test sets */
sklearn.model_selection import train_test_split
/* Cross Validation Splitter is used for model
   evaluation */
sklearn.model_selection import ShuffleSplit
shuffle_split = ShuffleSplit(test_size=.4, train_size=.6,
 n_splits=10)
/* cross_val_score has parameters: target
   model, train data, and target output: y_i */
sklearn.model_selection import cross_val_score
/* According to each ML algorithms, the
   accuracy of malicious behavior detection is
   resulted */
begin
    sklearn.ensemble import ML_Algorithms
    ML_object = ML_Algorithms(random_state=0)
    Detection_Accuracy = cross_val_score(ML_object,
     X_test, y_test, cv=shuffle_split)
end
```

In Table VII, all of the ML algorithms aside from for Ridge Regression exhibit the high detection accuracy. Almost 100% detection accuracy is obtained from each of KNN, Random Forest, Gradient Boosting, and SVM. However, the powerful MLP (deep learning algorithm) does not provide accuracy as good as those of the other four ML algorithms. Consequently, the test results suggest that detecting malicious behavior can be achieved with a high rate in IoT environment.

TABLE VII
DETECTION ACCURACY FOR MALICIOUS BEHAVIOR ACROSS SIX MODELS.

| Random | MLP | SVM | KNN | Gradient | Ridge |
|---|---|---|---|---|---|
| 99.81% | 94.37% | 99.07% | 98.58% | 96.82% | 26.82% |

### D. MIB-IoT for data source identification

*1) Identification accuracy for the data source:* We also employ six ML algorithms to identify the data source. The dataset consists of data transmitted by 54 sensors, and the dataset structure is shown in Table VIII.

TABLE VIII
FIVE ATTRIBUTES FOR DATA SOURCE IDENTIFICATION.

| Temp | Humidity | Light | Volt | Device ID |
|---|---|---|---|---|

We also specify the dataset for data source identification as follows:

- $D^{ID} = \{D_1^I, D_2^I, D_3^I, \ldots, D_n^I\}$, where $n$ represents 2,210,084.
- $D_i^I = (\mathbb{X}_i, y_i)$ where $D_i^I$ is the $i^{th}$ entry of the dataset, $\mathbb{X}_i$ is the vector of the input data, and $y_i$ is the output data: the sensor ID.
- $\mathbb{X}_i = (x_{i1}, x_{i2}, \ldots, x_{id})$, where $d$ is four representing the four attributes: temperature, humidity, light, and voltage.
- $y_i = (y_i)$, where $y_i$ represents the device ID: 1 to 54.

The test result of data source identification is described in Table IX. It should be noted that all of the models, except for Random Forest, show poor modeling accuracy. Even MLP, despite its long training time, shows the poorest identification accuracy compared to other ML algorithms.

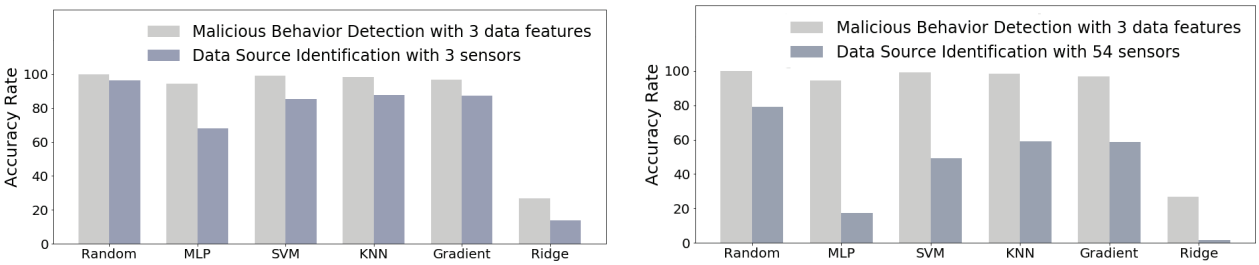
TABLE IX
DATA SOURCE IDENTIFICATION ACCURACY ACROSS SIX MODELS.

| Random | MLP | SVM | KNN | Gradient | Ridge |
|---|---|---|---|---|---|
| 78.51% | 17.51% | 49.27% | 58.98% | 58.72% | 1.75% |

Basically, the ML modeling for malicious behavior detection is similar to the ML modeling for data source identification; furthermore, the accuracy depends on the number of $y_i$, which is the output vector. The model of malicious behavior detection has three different kinds of output vectors: normal operation data, hardware fault data, and malicious data. However, data source identification has 54 kinds of output vectors consisting of 54 sensors.

In Fig. 4(a), comparison between two target ML models for malicious behavior detection and for data source identification is provided along with the accuracy rate. The accuracy rates (99.81% and 96.53%) are similar between the two Random Forest models for malicious behavior detection (e.g., three different data features: normal operation data, hardware fault data, and malicious data) and data source identification (e.g., three different sensors) due to the number of output vectors, as three kinds. The difference in accuracy between each of the two target models (malicious behavior detection and data source identification) of SVM, KNN, and Gradient Boosting, are 13.60%, 11.14%, and 9.29%, respectively.

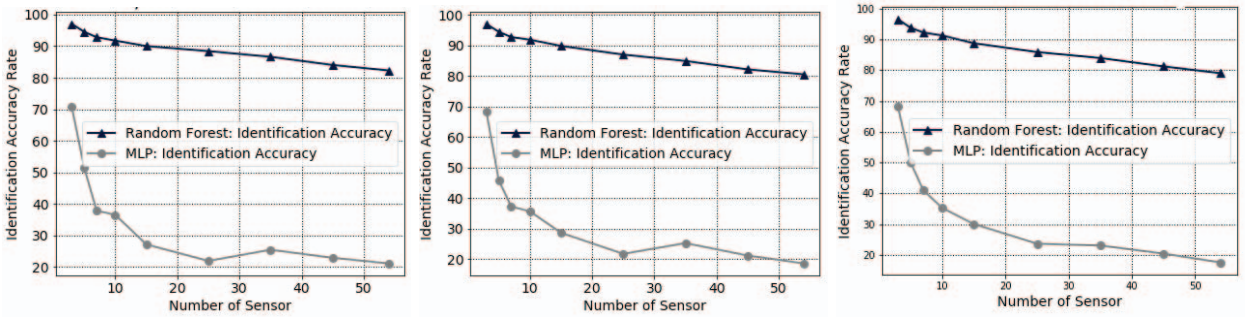
In Fig. 4(b), with more data (e.g., over 2M data entries) and sensors (e.g., 54), two ML models for malicious behavior detection and data source identification are compared and analyzed. For malicious behavior detection the models of Random




(a) Malicious detection & data source identification with three sensors.

(b) Malicious detection & data source identification with 54 sensors.

Fig. 4. Comparison between malicious behavior detection and data source identification with different numbers of sensor: three and 54.




(a) Scenario 1: Normal operation data.

(b) Scenario 2: Normal and hardware data.

(c) Scenario 3: Normal, hardware, and malicious data.

Fig. 5. Random Forest and MLP with three dataset scenarios: scenario 1, scenario 2, and scenario 3.

Forest, MLP, SVM, KNN, and Gradient Boosting still have the high accuracy rates of 99.81%, 94.37%, 99.07%, 98.58%, and 96.82% respectively; however, for data source identification, the models of those ML algorithms have poorer accuracy rates of 78.51%, 17.51%, 49, 27%, 58.98%, and 58.72%. As the number of classification increases, the accuracy of the ML models dynamically drops except for the Random Forest model, which shows 99.81% for malicious behavior detection and 78.51% for data source identification.

*2) Relationship between sensor number and data features:* For the modeling and simulation, we use 54 sensors and three data features: normal operation data, hardware fault data, and malicious data. The relationship between the number of sensors and the data features is analyzed and illustrated in Fig. 5. We test the accuracy of data source identification by Random Forest classifier and MLP with three dataset scenarios:

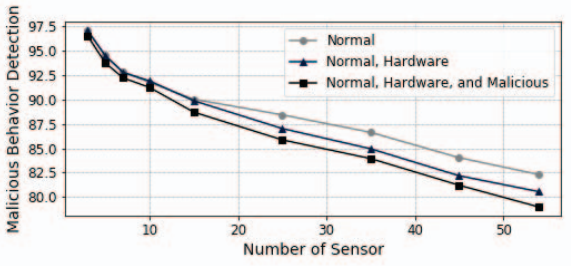
- Scenario 1: Dataset consisting of normal operation data only.
- Scenario 2: Dataset consisting of normal operation data and hardware fault data.
- Scenario 3: Dataset consisting of normal operation data, hardware fault data, and malicious data.

Fig. 5(a) shows the data source identification accuracies of Random Forest and MLP models with the dataset consisting of normal operation data only, that is scenario 1. As the number of sensor increases from three to 54, the identification accuracy of the Random Forest model decreases from 97.04% to 82.31%. Regarding to the Random Forest model, the coincident result has been observed that the accuracy of data source identification with only three sensors is similar to the accuracy of malicious behavior detection. It is worth noting that the Random Forest classification is consistent with the sensor data for not only detecting malicious behavior, but also for identifying the data source. The identification accuracy of MLP model drops dynamically from 68.29% to 17.51%.

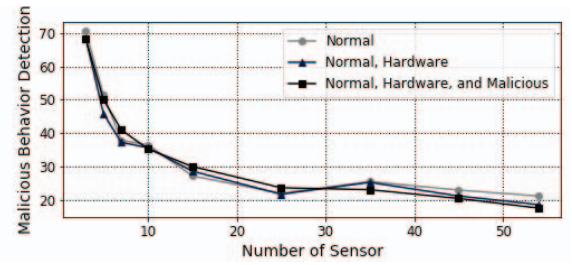
As shown in Fig. 5(b), the identification accuracy of the Random Forest model decreases from 97.09% to 80.55% with the dataset consisting of normal operation data and hardware fault data, as is the case in scenario 2. The identification accuracy is slightly lower than that in scenario 1, due to the different composition of the dataset, which indicates that hardware fault data distracts the accuracy of data source identification. The identification accuracy of the MLP model drops dynamically from 68.45% to 18.57%.

As shown in Fig. 5(c), the identification accuracy of the Random Forest model decreases from 96.53% to 78.51% with the dataset consisting of normal operation data, hardware fault data, and malicious data, as is the case in scenario 3. The identification accuracy of MLP model drops dynamically from 68.29% to 17.51%.

In Fig. 6, the identification accuracies of the Random Forest and MLP models are compared across three different datasets. Regarding to the Random Forest model, the identification accuracy is worse with the dataset consisting of normal operation data, hardware fault data, and malicious data as




(a) Random Forest model with three dataset.




(b) MLP model with three datasets.

Fig. 6. Comparison between Random Forest and MLP models with three datasets.

shown in Fig. 6(a). it can be seen that the sensor data has a high localization feature, and that malicious data is the key factor for decreasing the identification accuracy. However, the MLP model has no affection of malicious data, as shown in Fig. 6(b).

Malicious data mimics normal operation data, which makes it hard for the Random Forest model to distinguish sensor identification, even though the Random Forest classifier has high detection accuracy for malicious behavior.

## V. MIB-IoT for Malicious Behavior Detection and Data Source Identification

One of the benefits of using MIB-IoT, the improved accuracy can be secured because MIB-IoT represents the properties of the ML model and features of dataset, because the ML models has high dependence on the parameters for ML training as well as the number of data feature in terms of the accuracy rate.

### A. MIB-IoT of malicious behavior detection

For malicious behavior detection, two types of MIB-IoT have been built by Random Forest classifier and SVM with their highest accuracies: 99.81% and 99.07%, respectively.

*1) MIB-IoT(1).Property(2):* The Random Forest classifier is used to build MIB-IoT for malicious behavior detection as specified in Table X. MIB-IoT.Property(2) is modeled using five data attributes: temperature, humidity, light, voltage, and data feature, and its target accuracy is 99.81%.

*2) MIB-IoT(1).Property(3):* SVM is used to build an MIB-IoT for malicious behavior detection as specified in Table XI. MIB-IoT.Property(3) is modeled using five data attributes: temperature, humidity, light, voltage, and data feature, and its target accuracy is 99.07%.

TABLE X
Random Forest-based MIB-IoT for malicious behavior detection.

| HVAC(1).MIB-IoT(1): 1.3.6.1.4.1.100000.1.1.1 | | |
|---|---|---|
| HVAC(1).MIB-IoT(1).Property(2): 1.3.6.1.4.1.100000.1.1.1.**2** | | |
| OID | Name | Value |
| 1 | Target | Malicious behavior detection |
| 2 | Machine Learning | Random Forest |
| 3 | Data List | Temp, Humidity, Light, Voltage, Data Feature |
| 4 | Best Parameter | Estimators: 30 |
| 5 | Target Accuracy | 99.81% |

TABLE XI
SVM-based MIB-IoT for malicious behavior detection.

| HVAC(1).MIB-IoT(1): 1.3.6.1.4.1.100000.1.1.1 | | |
|---|---|---|
| HVAC(1).MIB-IoT(1).Property(3): 1.3.6.1.4.1.100000.1.1.1.**3** | | |
| OID | Name | Value |
| 1 | Target | Malicious behavior detection |
| 2 | Machine Learning | SVM |
| 3 | Data List | Temp, Humidity, Light, Voltage, Data Feature |
| 4 | Best Parameter | Gamma: 0.1 |
| 5 | Target Accuracy | 99.07% |

### B. MIB-IoT for data source identification

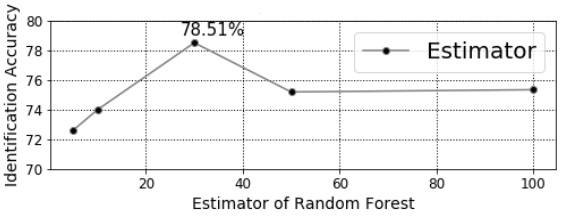
*1) MIB-IoT(1).Property(4):* For data source identification, only the Random Forest classifier is used, due to the fact that the other ML algorithms have poor identification accuracies as shown in Fig. 4(b).

With the estimators of the auto option, the Random Forest classifier shows an accuracy of 72.63% as shown in 4(b). Further, in order to improve the accuracy, we adjust the estimators parameter. As shown in Fig. 7(a), the results show that the accuracy is improved when the estimator is 30. With 30 estimators, the Random Forest model has the highest accuracy of 78.51%; consequently, 5.88% is improved with MIB-IoT.

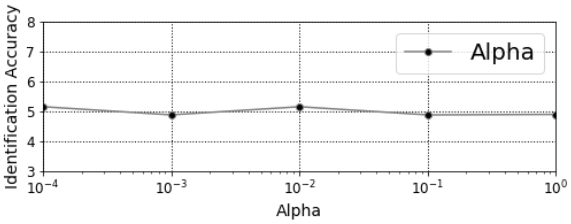
The Random Forest classifier is used to build an MIB-IoT of data source identification as specified in Table XII. In Fig. 2, the Profile.Property(4) is modeled using five data attributes: temperature, humidity, light, voltage, and sensor ID, and its target accuracy is 78.51%.

## VI. Conclusion and Future Plan

To mitigate suspicious attack attempts, we can typically construct an ML model. Unsurprisingly, information on the constructed ML model (e.g., algorithm parameters, hyper parameters, input data list, and target accuracy) would be valuable and can sometimes be shared with other organizations and/or entities to reuse and extend exiting one. To meet this requirement, we propose a new profiling method called "Management Information Base for IoT (MIB-IoT)" by extending




(a) Random Forest model with different estimator.




(b) MLP model with different Alphas.

Fig. 7. Comparison between Random Forest model with different Estimators and MLP model with different alphas trained by a dataset consisting of normal operation data, hardware fault data, and malicious data

TABLE XII
RANDOM FOREST-BASED MIB-IOT FOR DATA SOURCE IDENTIFICATION.

| HVAC(1).MIB-IoT(1): 1.3.6.1.4.1.100000.1.1.1 | | |
|---|---|---|
| HVAC(1).MIB-IoT(1).Property(4): 1.3.6.1.4.1.100000.1.1.1.4 | | |
| OID | Name | Value |
| 1 | Target | Data source identification |
| 2 | Machine Learning | Random Forest |
| 3 | Data List | Temp, Humidity, Light, Voltage, and Sensor ID |
| 4 | Best Parameter | Estimators: 30 |
| 5 | Target Accuracy | 78.51% |

conventional MIB to a more generalized structure so as to represent not only structured properties of network objects but also the best ML model for each network object. We carefully developed MIB-IoT so as to make it compliant with the existing MIB standards. Hence, MIB-IoT profiles can be used for various applications such as abnormal behavior detection, malicious behavior detection, and data source identification. Our experiment results demonstrate that the classification algorithm using MIB-IoT is capable of achieving an accuracy of $99.81\%$ for malicious behavior detection and $78.51\%$ for data source identification respectively. We specifically note that the achieved accuracy ($78.51\%$) using the pre-trained Random Forest model in the MIB-IoT profile is significantly better than the accuracy ($72.63\%$) of the Random Forest model with default settings. In future work, we plan to improve MIB-IoT for autonomous updating, which minimizes the human burden for profiling, and we will also study the possibility of unsupervised ML algorithms.

## ACKNOWLEDGMENTS

This research was partly supported by the NRF of Korea (NRF-2017K1A3A1A17092614), the ICT Consilience Creative support program (IITP-2019-2015-0-00742), and the ITRC support program (IITP-2019-2015-0-00403) supervised by the IITP.

## REFERENCES

[1] A. L. Samuel, "Some studies in machine learning using the game of checkers," *IBM Journal of Research and Development*, vol. 3, no. 3, pp. 210–229, Jul 1959.

[2] H. M. B. D. Cardinaux F., Brownsell S., "Modelling of behavioural patterns for abnormality detection in the context of lifestyle reassurance." *In: Ruiz-Shulcloper J., Kropatsch W.G. (eds) Progress in Pattern Recognition, Image Analysis and Applications. CIARP 2008. Lecture Notes in Computer Science*, vol. 5197, 2008.

[3] M. Rose and K. McCloghrie, "Structure and identification of management information for tcp/ip-based internets," *IETF, STD 16, RFC 1155*, May 1990. [Online]. Available: https://www.rfc-editor.org/info/rfc1155

[4] M. Hildebrandt, S. Gutwirth, M. Hildebrandt, and S. Gutwirth, *Profiling the European citizen: cross-disciplinary perspectives*, 1st ed. Springer Publishing Company, Incorporated, 2008.

[5] S. Madden. (2003) Intel berkeley research lab data. [Online]. Available: http://db.csail.mit.edu/labdata/labdata.html

[6] R. Yan, T. Xu, and M. Potkonjak, "Data integrity attacks and defenses for intel lab sensor network," in *Proceedings of the 2015 IEEE 2Nd World Forum on Internet of Things (WF-IoT)*, ser. WF-IOT '15. Washington, DC, USA: IEEE Computer Society, 2015, pp. 721–726. [Online]. Available: http://dx.doi.org/10.1109/WF-IoT.2015.7389143

[7] Y. Meidan, M. Bohadana, A. Shabtai, J. D. Guarnizo, M. Ochoa, N. O. Tippenhauer, and Y. Elovici, "Profiliot: a machine learning approach for iot device identification based on network traffic analysis," in *Proceedings of the Symposium on Applied Computing*, ser. SAC '17. New York, NY, USA: ACM, 2017, pp. 506–509.

[8] A. Serbanati, C. M. Medaglia, and U. B. Ceipidor, "Building blocks of the internet of things: State of the art and beyond," *Deploying RFID-Challenges, Solutions, and Open Issues, InTech*, 2011.

[9] R. Clarke, "Profiling: a hidden challenge to the regulation of data surveillance," *Journal of Law and Information Science*, vol. 4, p. 403, 1993.

[10] R. Ferrando and P. Stacey, "Classification of device behaviour in internet of things infrastructures: towards distinguishing the abnormal from security threats," in *Proceedings of the 1st International Conference on Internet of Things and Machine Learning*, ser. IML '17. New York, NY, USA: ACM, 2017, pp. 57:1–57:7.

[11] S. Myagmar, A. J. Lee, and W. Yurcik, "Threat modeling as a basis for security requirements." *Symposium on Requirements Engineering for Information Security (SREIS)*, Aug 2005. [Online]. Available: http://d-scholarship.pitt.edu/16516/

[12] P. Ji and M. Szczodrak, "A multivariate model for data cleansing in sensor networks," Mar 2019.

[13] R. Yan, T. Xu, and M. Potkonjak, "Data integrity attacks and defenses for intel lab sensor network," *2015 IEEE 2nd World Forum on Internet of Things (WF-IoT)*, pp. 721–726, Dec 2015.

[14] S. Lee, S. Wi, E. Seo, J. Jung, and T. Chung, "Profiot: Abnormal behavior profiling (abp) of iot devices based on a machine learning approach," in *2017 27th International Telecommunication Networks and Applications Conference (ITNAC)*, Nov 2017, pp. 1–6.

[15] Y. x. Xie, X. g. Chen, and J. Zhao, "Data fault detection for wireless sensor networks using multi-scale PCA method," *2nd International Conference on Artificial Intelligence, Management Science and Electronic Commerce (AIMSEC)*, pp. 7035–7038, Aug 2011.

[16] M. E. Ahmed, S. Ullah, and H. Kim, "Statistical application fingerprinting for ddos attack mitigation," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 6, pp. 1471–1484, Jun 2019.

[17] K. McCloghrie and M. Rose, "Management information base for network management of tcp/ip-based internets: Mib-ii," *STD 17, RFC 1213*, Mar 1991. [Online]. Available: https://www.rfc-editor.org/info/rfc1213

[18] L. Pan and J. Li, "K-nearest neighbor based missing data estimation algorithm in wireless sensor networks," *Wireless Sensor Network*, vol. 2, pp. 115–122, Jan 2010.

[19] G. Kortuem, F. Kawsar, D. Fitton, and V. Sundramoorthy, "Smart objects as building blocks for the internet of things," *IEEE Internet Computing*, vol. 14, no. 1, pp. 44–51, Jan 2010.