Anomaly Detection in Smart Home Operation From User Behaviors and Home Conditions

Masaaki Yamauchi, *Graduate Student Member, IEEE*, Yuichi Ohsita, *Member, IEEE*, Masayuki Murata, *Member, IEEE*, Kensuke Ueda, and Yoshiaki Kato, *Senior Member, IEEE*

***Abstract*—As several home appliances, such as air conditioners, heaters, and refrigerators, were connecting to the Internet, they became targets of cyberattacks, which cause serious problems such as compromising safety and even harming users. We have proposed a method to detect such attacks based on user behavior. This method models user behavior as sequences of user events including operation of home IoT (Internet of Things) devices and other monitored activities. Considering users behave depending on the condition of the home such as time and temperature, our method learns event sequences for each condition. To mitigate the impact of events of other users in the home included in the monitored sequence, our method generates multiple event sequences by removing some events and learning the frequently observed sequences. For evaluation, we constructed an experimental network of home IoT devices and recorded time data for four users entering/leaving a room and operating devices. We obtained detection ratios exceeding 90% for anomalous operations with less than 10% of misdetections when our method observed event sequences related to the operation. In this article, we also discuss the effectiveness of our method by comparing with a method learning users' behavior by Hidden Markov Models.**

***Index Terms*—Anomaly detection, behavior pattern, consumer electronics, cybersecurity, Internet of Things, operation by attackers, smart home.**

I. Introduction

HOME appliances such as refrigerators, heaters, and air conditioners are being increasingly integrated with Internet connections to expand connectivity beyond personal computers and smartphones. These devices are collectively called IoT (Internet of Things) devices. Users can obtain information from IoT devices and operate them using smartphones, tablets, or smart speakers.

Currently, seven billion IoT devices are connected to the Internet, with a substantial increase to 215 billion devices expected by 2025 [1]. As the number of devices connected to the Internet increases, the risk for these devices to be targeted by cyberattacks also increases [2]–[5]. In fact, direct attacks and malware targeting of IoT devices [6], [7] have already been reported.

Most of the current attacks targeting IoT devices aim to create botnets [8], [9]. Such attacks are detectable by methods based on analyses of attacker behaviors [10]–[12] or through usual traffic comparisons [13], [14].

However, as IoT devices are closely included in our routine, attacks may cause immediate and personal harm to users [15]. For instance, the operation of IoT devices by attackers may threaten user safety, potentially even causing physical harm, through actions such as changing the temperature of an air conditioner or the settings of a healthcare device. In addition, simultaneous attacks on high-power IoT devices can suddenly increase energy demands and lead to major power outages [16]. Therefore, methods to detect and prevent cyberattacks are necessary for the widespread adoption of IoT devices.

Intrusion detection systems are the typical countermeasures against attacks targeting IoT devices. Zarpelao *et al.* [17] presented an intrusion detection system to detect anomalous traffic over IoT devices by either comparing the packets to predefined rules or detecting outliers from the observed traffic. Although existing intrusion detection systems assume that legitimate and anomalous traffic patterns are notably different, both attackers and legitimate users send the same types of packets to operate IoT devices. For instance, if an attacker sends packets via the malware-infected smartphone of a legitimate user, even the source IP address is identical to that of the legitimate user. Consequently, existing intrusion detection systems cannot distinguish between packets sent by legitimate users and attackers based on the available information.

To improve user protection, we proposed a method to detect the anomalous operation of home IoT devices by learning user behaviors during operation [18]. Our method learns user behaviors. Then, when a command arrives at a home IoT device, the method verifies whether it matches the learned behavior. When a command does not agree with the learned behavior, it is classified as an anomaly. Therefore, the proposed method can detect anomalous operation that does not fit the user behaviors even if the commands are generated by malware-infected smartphones.

User behaviors may depend on the condition of the room such as time and temperature. Considering the above point,

Manuscript received July 2, 2019; revised October 22, 2019, December 28, 2019, and February 17, 2020; accepted March 14, 2020. Date of publication March 18, 2020; date of current version April 23, 2020. This work was supported by the Mitsubishi Electric Cybersecurity Research Alliance Laboratories. *(Corresponding author: Masaaki Yamauchi.)*

Masaaki Yamauchi, Yuichi Ohsita, and Masayuki Murata are with the Graduate School of Information Science and Technology, Osaka University, Suita 565-0871, Japan (e-mail: m-yamauchi@ist.osaka-u.ac.jp; y-ohsita@ist.osaka-u.ac.jp; murata@ist.osaka-u.ac.jp).

Kensuke Ueda is with the Advanced Technology R&D Center, Mitsubishi Electric Corporation, Amagasaki 661-8661, Japan (e-mail: ueda.kensuke@ce.mitsubishielectric.co.jp).

Yoshiaki Kato is with the Information Technology R&D Center, Mitsubishi Electric Corporation, Kamakura 247-8501, Japan (e-mail: kato.yoshiaki@dh.mitsubishielectric.co.jp).

Digital Object Identifier 10.1109/TCE.2020.2981636

we define user behaviors by sequences of operations according to conditions such as time and sensor measurements in the home network. When we learn the sequences of operations, we should consider the case that a monitored sequence of operations includes operations by multiple users because a smart home may have multiple users. Such operations by the other users have a large impact on learning essential user behavior. To mitigate such impact, our method generates multiple sequences of operations by removing some operations from the monitored sequence and learns sequences that are frequently observed from them.

In this article, we extend our previous work [18] by investigating the effectiveness of our method by comparing it with a method to learn normal activities of a user by using Hidden Markov Models (HMM) [19]. In addition, we compared our method with methods using a part of our leaning method, and discussed the effectiveness of our method.

The rest of this article is organized as follows. We describe the related work in Section II. Then, We describe the proposed method to detect anomalous operations in Section III. After that, we report the evaluation of the method and the corresponding results in Section IV. Finally, we draw conclusions and discuss directions of future work in Section V.

II. Related Work

As several home appliances, such as air conditioners, heaters, and refrigerators, are being connected to the Internet, and they became targets of cyberattacks [6], [7], which can cause serious problems such as compromising safety and even harming users [15]. Therefore, the methods to detect attacks for the home IoT devices have been proposed [10]–[14]. For example, Hodo *et al.* proposed a method to detect intruded IoT devices. This method uses an artificial neural network trained by using the packet traces. By using the artificial neural network, this method classifies the normal and threat patterns on an IoT network and detects DoS attacks [20]. Xu *et al.* proposed the method to detect intruded home IoT devices that become a part of botnets. This method uses a bloom-filter based analytics framework to find anomalous packets and detect intruded home IoT devices [10]. Martin *et al.* proposed a comprehensive home network defense method against attacks to home IoT devices. This method uses honeypot to find attacks by the signatures-based method and changes settings of firewalls to drop the attacking packets [11].

They focus on the intrusion or anomaly on the IoT devices and detect anomalies based on the difference of traffic from/to the IoT devices. However, if an attacker sends packets via the malware-infected smartphone of a legitimate user, even the source IP address is identical to that of the legitimate user. That is, the anomalous operation cannot be detected by the methods based on the difference of traffic from/to the IoT devices. Therefore, we proposed a method to detect anomalous operations by learning user behaviors.

Ramapatruni *et al.* also proposed a method to detect anomalous operations by learning user behaviors. This method uses Hidden Markov Models (HMM) to learn the normal activities of a user. This method uses the information obtained from sensors and/or statuses of the home IoT devices as the observations. By using the observations, this method learns the parameters of HMM. Then, this method detects the anomalous operations if an operation whose probability is low occurs. They demonstrated the accuracy of this method by using the dataset collected at the smart home environment deployed by them. This method focuses on the case of a single user [19]. However, a smart home may have multiple users. Therefore, in this article, we propose a method to detect anomalous operations even in the case of multiple users in a home.

There exist papers on learning user behaviors in other areas. Rashid *et al.* proposed a method to detect behaviors of malicious insiders of organizations by learning behaviors of legitimate users. In this article, the behaviors are defined by a sequence of actions stored in log files of computer systems and modeled by a hidden Markov model. Then, this method detects the malicious insider whose behavior does not match the learned model [21]. Haider *et al.* proposed a method to detect zero-day attacks for cloud servers by modeling the users' usage behavior. This article also defined user behaviors as a sequence of actions. This article modeled the behavior by using nested-arc hidden semi-Markov model (NAHSMM). This article demonstrated that the method detects major attack families such as DoS and worms by learning the model by using traffic data [22].

Methods to monitor people in a home and learn their behavior have also been proposed. For example, Aran *et al.* proposed a method to detect anomalous behavior in elderly daily life. This method monitors locations of a person in a house by using motion sensors, chair sensors, bed sensors and so on. Then, this method uses the time series of the locations to detect anomalous behavior [23].

Similar to our method, they learn the sequence of events, but they cannot be applied to learning the users' operation in a smart home. First, user behaviors depend on the conditions of the home such as time of day, temperature, and humidity, but they do not consider the behavior changing by the condition. Another problem is that a smart home has multiple users. As a result, a monitored sequence may include the event by the other users, which has an impact on learning essential user behaviors. Note that the essential event sequences include only the events from one user. Therefore, we proposed a new method to learn users' behaviors so as to detect anomalous operations.

III. Anomaly Detection in Smart Home Operation

In this section, we introduce a method to detect the anomalous operation of home IoT devices attributed to attackers. The proposed method considers specific patterns of user behavior depending on diverse conditions. For example, when users return home and feel cold, they turn on a heater and then a humidifier, whereas if it is warm, they never turn on the heater. In addition, operation sequences reflect user behavior. For example, a user turns on the heater and then the humidifier, whereas another user first turns on the humidifier. The proposed method learns these types of condition-dependent

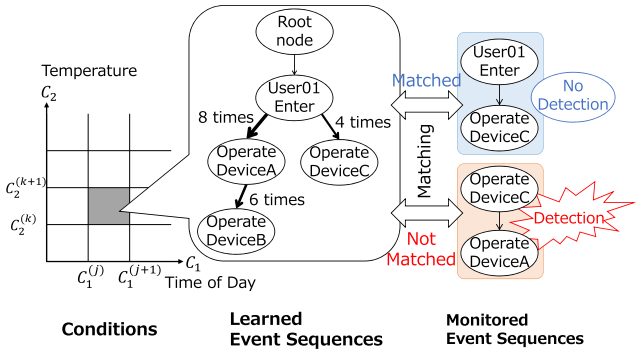

Fig. 1. Overview of detection model.

Generated event sequences	A	B	A→B	C	A→C	B→C	A→B→C
Variable v of Algorithm 1	*001*	*010*	*011*	*100*	*101*	*110*	*111*
Removed events	B, C	A, C	C	A, B	B	A	None

Fig. 2. Generated event sequences when events A, B, and C occur in this order within T seconds.

operation sequences to establish behavior patterns, and it classifies deviations from these learned patterns as anomalies.

In this section, we describe the proposed model of user behavior. Then, we explain the learning of user behaviors and the detection of anomalous operations.

A. Learning Model

Fig. 1 illustrates the model to learn user behaviors. This model is defined by the conditions and the stored user behaviors for each condition. The conditions are represented as a table in which each cell corresponds to each condition. For each condition, we store the learned user behaviors. The rest of this subsection explains the detail of the model of the conditions and the user behaviors for each condition.

1) Conditions: We define a condition as a combination of time of day and sensor measurements such as room temperature, humidity, and noise value. The variables representing the various components of a condition are denoted as c_i, where index i starts from 1 and reaches a maximum i^{max}. An example, as shown in Fig. 1 has two metrics, time of day and temperature to define the condition. That is, i^{max} is 2 and c_1 represents the time of a room, whereas c_2 represents its temperature. For tractability, we discretize continuous data by using multiple thresholds for each type of data. Specifically, a value of c_1 and c_2 satisfying $c_1^{(j)} \leq c_1 \leq c_1^{(j+1)}$ and $c_2^{(k)} \leq c_2 \leq c_2^{(k+1)}$, where $c_1^{(j)}$ is the j-th threshold of the 1st variable and $c_2^{(k)}$ is the k-th threshold of the 2nd variable, is classified into the colored area of Fig. 1.

2) User Behaviors: In this article, we define an *event* as any monitored behavior of a user, including the operation of IoT devices such as operating a heater, opening a refrigerator, and turning a TV's volume up, and any other behavior monitored by sensors, such as entering or leaving a room. Also, we define an *event sequence* as events occurring within a timeframe of T seconds from a previous event. We store the event sequences for each condition. Any model and data structure can be used to store the event sequence, but in this article, we model the sequences by a tree where the children nodes of the root are the initial events and the leaves are the final events. Anomalous event sequences occur when an executed sequence is not contained in any tree. The "Learned Event Sequences" in Fig. 1 shows an example of event sequences. In this example, an event sequence of "User01 Enter", "Operate DeviceA", and "Operate DeviceB" is observed 6 times, that of "User01 Enter" and "Operate DeviceA" is observed 2 times and that of "User01 Enter" and "Operate DeviceC" is observed 4 times. The learned event sequence can be used to detect anomalies. For example, "Operate DeviceA" occurs only after "User01 Enter" occurs. Thus, the "Operate DeviceC" and "Operate DeviceA" are detected as anomalous.

B. Learning User Behaviors

Our method learns user behaviors by storing the monitored event sequences. In the proposed method, we store the sequences at a home gateway that can monitor all the packets from the IoT devices and users' smartphones and other operation devices. However, some event sequences may include events from different users when more than one person inhabits or visits the smart home. To mitigate the impact of these events, which should be treated as noise, and learn only essential event sequences, we remove events from the observed sequences. Note that the essential event sequences include only the events from one user.

Our method learns the event sequences by the following steps. First, our method generates multiple event sequences by removing some events from the observed sequences. Then our method learns the frequent event sequences by using the generated event sequences. The event sequences that occur many times are the essential sequences of events that the users often do. Therefore, we use only the event sequences that occur many times as the learned event sequences of the legitimate users. The rest of this section explains the detail of each step to learn user behaviors.

1) Generation of Event Sequences: Algorithm 1 generates event sequences that are subsets of the monitored event sequence $s^{\text{monitored}}$. In Algorithm 1, we introduce a binary variable v which indicates the events included in the generated sequence; if ith bit of v is 1, the generated sequence s^{tmp} includes ith event of $s^{\text{monitored}}$. Otherwise ith event of $s^{\text{monitored}}$ is not included in s^{tmp}. After generating one event sequence, v is incremented. By continuing the above steps, we generate all subsets of $s^{\text{monitored}}$.

Figure 2 shows an example to generate event sequences when a sequence "A-B-C" is monitored. The first v is set to

Algorithm 1 Generating Event Sequences From the Monitored Event Sequence

Input: $s^{\text{monitored}}$: monitored sequence of events
Output: S: List of generated sequences of events
  function GENERATEEVENTSEQUENCES($s^{\text{monitored}}$)
    $v \Leftarrow$ UnsignedBinaryVariable(1)
    $S \Leftarrow$ EmptyList()
    while $v <$ UnsignedBinaryVariable($2^{|s^{\text{monitored}}|}$) **do**
      $s^{\text{tmp}} \Leftarrow$ EmptySequence()
      for $i = 1 \ldots |s^{\text{monitored}}|$ **do**
        if ith bit of v = 1 **then**
          Add ith event of $s^{\text{monitored}}$ to s^{tmp}
        end if
      end for
      Add s^{tmp} to S
      Increment v
    end while
    return S
  end function

001, and the sequence "A" is generated. Then, v is incremented and becomes 010. The sequence "B" is generated. These steps are continued to generate all patterns of subsets of the monitored event sequence.

2) Selection of Conditions: The proposed method selects the condition based on that corresponding to the initial event in the sequence and updates the model for the selected condition. However, to effectively use event sequences and learn user behaviors even from a few sequences per condition, we update not only the model corresponding to the condition of the initial event but also models with similar conditions. When the condition of the initial event is $\{c_1^{root}, \ldots, c_{i^{max}}^{root}\}$, the sequence is used to update the model for the region on which c_i satisfies $c_i^{root} - \alpha_i \leq c_i \leq c_i^{root} + \alpha_i$ for some value α_i, which can vary according to index i.

3) Updating Tree of Event Sequences: When the home gateway observes an event sequence, nodes and links are created in the tree corresponding to the selected conditions, thus including the event sequence, where the initial and final events of the sequence are the root and leaf in that branch, respectively. Then, we increment a counter for each link on the route corresponding to the event sequence. Algorithm 2 shows the steps to update the tree of an event sequence. We update the tree by calling Update(root, s) where root is the root node for the condition. If the length of sequence s, $|s|$ is 0, this procedure ends. Otherwise, this procedure searches the children of the current node pos whose event is the first event of s, s_1. If a child whose event is s_1 is found, we set next to the found node. Otherwise, we add a new node and set next to the new node. Then, we update the counter for the link between pos and next. Finally, we iteratively call Update() to update the subtree whose root is next by using the sequence $s_{2\ldots|S|}$ which is the sequence generated from s by removing s_1. By repeating this procedure, we update the tree.

We repeat this procedure to each generated sequence. As a result the count for links related to frequent event sequences increases. Thus, we detect anomalies by using the pruned tree that contains only the links whose counters exceed threshold $n_d \times L_{num}$, where n_d is an adjustment parameter, d is the depth

Algorithm 2 Updating the Tree of Each Condition Using Generated Event Sequences

Input: pos: node of the start point of update, s: event sequence
Output:
  function UPDATE(pos, s)
    if $|s| = 0$ **then**
      return
    end if
    if pos has a child whose event is s_1 **then**
      next ←child of pos whose event is s_1
    else
      next ←new node whose event is s_1
      add next to the list of child of pos
    end if
    increment counter for the link (pos, next)
    Update(next, $s_{2..|s|}$)
  end function

Algorithm 3 Comparing a New Event Observed Sequence With Learned Event Sequences

Input: pos: node of learned event sequences, s: observed event Sequence
Output: Matched, Pending, Unmatched
  function SEARCH(pos, s)
    if $|s| = 0$ and pos has no children **then**
      return Matched
    else if $|s| = 0$ **then**
      return Pending
    else if pos has a child whose event is s_1 **then**
      next ←child of pos whose event is s_1
      return Search(next, $s_{2..|s|}$)
    else
      return Unmatched
    end if
  end function

of the links, and L_{num} is the total number of learned operations for the target device. In this manner, we eliminate spurious events (i.e., noise) from the event sequence.

C. Detection

When the home gateway observes operation execution, it generates the corresponding event sequences, which are compared to learned behaviors.

1) Generation of Event Sequences: An executed operation generates multiple event sequences, similar to during learning, by removing events from the sequence within a T seconds timeframe from a previous event and uses these sequences to determine whether the operation is anomalous.

2) Decision-Making: We check whether an executed operation is anomalous by comparing the generated event sequences with the learned behaviors. Algorithm 3 shows the procedure to compare an event sequence with the learned event sequences. We compare the event sequence s with the learned event sequences by calling Search(root, s) where root is the root node for the condition of the initial event of the sequence s. In this procedure, we repeat searching the children until the nodes corresponding to all events in the executed sequence are located or not. If this procedure cannot locate the nodes corresponding to one of the events in the sequence, this procedure returns "Unmatched". If this procedure locates the

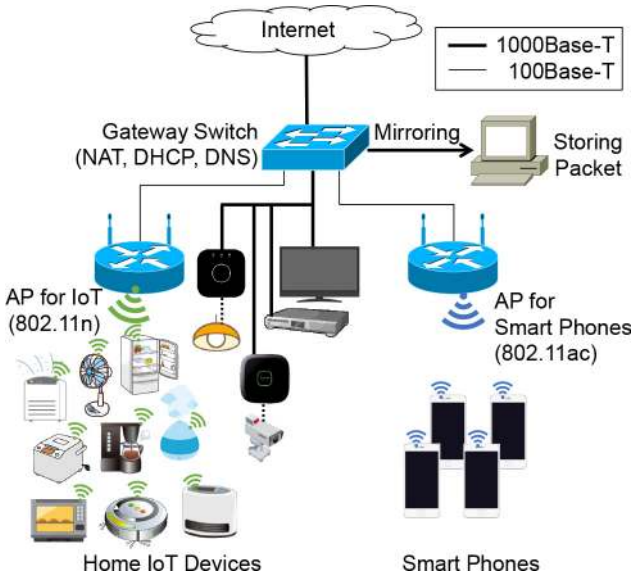

Fig. 3. Experimental home network environment.

nodes corresponding to all events and reaches a leaf node, this procedure return "Matched". If corresponding nodes are found for all events in the sequences, but the node corresponding to the final event is not a leaf, we need to wait for the next event to be executed because the sequence matches only the subset of the learned sequence and it depends on the next event whether the sequence completely matches the learned sequence. In this case, the procedure return "Pending".

This checking is performed for all generated event sequences. If the above procedure returns "Unmatched" for all generated event sequences, the operation is classified as anomalous, whereas if the above procedure returns "Matched" for one of the generated event sequences the operation is classified as legitimate. If the procedure returns "Pending" for one of the generated event sequences, we wait for the next event to be executed. If the next event does not occur within T seconds, searching is terminated. Otherwise, we generate event sequences including the next event and compare the resulting sequences with learned behaviors by using the same procedure described above.

IV. Evaluation and Results

A. Data

To obtain data for the evaluation of the proposed anomaly detection method, we constructed a network of home IoT devices in our laboratory, as shown in Fig. 3. We deployed two access points; one access point is used to connect electronic devices and another access point is used to connect smart phones. Both of the access points are connected to the gateway switch. The gateway switch relays all packets from/to electronic devices and smart phones. By configuring the gateway switch, we captured all packets from/to electronic devices and smart phones without loss.

In this experiment, we deployed 13 types of connectable consumer electronics listed in Table I. These home electronic devices can be connected to the Internet, are commercially available, and can be deployed in our laboratory. Before starting the experiment, we analyzed the packets from/to the deployed electronics when we control the devices and clarified the features of the packets when devices are operated. By using the features, we detect the operations of the electronics devices from the captured packets. In addition to the operations of the devices, we monitored the time when users entering or leaving a room. The time when users entering or leaving a room is obtained by monitoring the packets from smart phone of each user.

TABLE I
Deployed Home IoT Devices in the Experimental Home Network Environment to Evaluate Anomaly Detection

Device	Quantity
Air purifier	2
Automatic cooker	1
Coffee maker	1
Electric fan	4
Heater	1
Humidifier	1
Lighting equipment	6
Microwave oven	1
Monitor camera	1
Recorder	4
Refrigerator	1
Robot vacuum	1
Television	4

In this experimental home network environment, four students from our laboratory participated in the study. We allowed the participants to use the home IoT devices freely every other month for a total duration of 6 months. Additionally, we asked them to fill logs with information including time of operating the deployed electronics and time of entering and leaving the laboratory. We confirmed that these times can be matched with the monitored time of operating the electronics and the time of entering and leaving the laboratory from the captured packets.

We also captured sensor data such as temperature, humidity, and noise. As these sensor measurements remained at stable levels in the laboratory environment, we considered the time of day as the only condition for detection in this study.

We used two datasets, A and B, obtained in April, June, and August, and in May, July, and September 2017, respectively. The same participants used the IoT devices during the acquisition periods of the datasets.

For evaluation, we used the misdetection and detection ratios.

1) Misdetection Ratio: To determine the misdetection ratio, we require sufficient legitimate operations in the test data. Given the limited number of legitimate operations we were able to collect during the study period, we determined the misdetection ratio by using leave-one-out cross-validation [24], which divides the dataset into multiple sets and evaluates one of them after training using the others. After verifying all the combinations from these sets, we can obtain the overall results. We separated data in a daily timeframe. Then, we used the data of one day for testing and those of the other days for learning legitimate behaviors. Finally, we calculated the misdetection ratio from the number of misdetections to the total number of operations that turning on the power of each device.

TABLE II
SETTINGS OF PROPOSED AND ITS VARIANT METHODS

Method	Condition	Sequence	Noise removal
Proposed method	✓	✓	✓
Only condition	✓		—
Without condition		✓	✓
No noise removal	✓	✓	
Only sequence		✓	

2) Detection Ratio: The evaluation also included anomalous operations besides the observed legitimate operations. We used a strategy similar to that for misdetection to obtain the detection ratio. We separated the dataset by day and used the data of one day for testing and the remainder for learning.

We added 100 anomalous operations that resembled legitimate operations of turning on each device into the test data for each day and attempted to detect them. Then, we calculated the detection ratio from the number of detected anomalous operations over the days.

B. Compared Methods

To evaluate the effectiveness of our method we compared our method with the other methods. In this evaluation, we compare our method with the method proposed by Ramapartuni *et al.* [19]. By comparing with this method, we demonstrate the effectiveness of our method. Moreover, we also compare the methods without some part of our method. By comparing them, we evaluate the necessity of each part of our method.

1) HMM Method [19]: This method learns the users' behavior by HMM. This method uses the observations obtained by the sensors and/or statuses of devices. This method learns the parameters of the HMM so as to suit the sequences of the observations. Then, by using the learned model, this method detects anomalous operations if an operation whose estimated probability is less than a threshold occurs. In this evaluation, we use the states of devices shown in Table I and the list of users currently in the room as the observations.

2) Variant of Our Methods: Table II shows the variants of our method used in this section. The details of the variants are as follows.

a) Only condition: One important aspect of the proposed detection method is the use of event sequences and operation conditions. To investigate the effectiveness of using event sequences, we compared the proposed method with a variant using only condition information. For this evaluation, we used only the time of day as a condition, which was stored as part of the training data. Then, we defined an anomaly when the number of operations that occur in the timeframe from $Time - \alpha_1$ to $Time + \alpha_1$ is below $n_1 \times L_{num}$, where *Time* is the time of day of the tested operation.

b) Without condition: In a smart home, user behavior depends on the conditions. To investigate the effectiveness of condition information, we compared the proposed method with a variant using only sequence information with noise removal. This method learns all event sequences in the same way as our method except that it does not use the condition information.

TABLE III
EVALUATION PARAMETERS IN EACH DATASET TO EVALUATE THE PROPOSED ANOMALY DETECTION METHOD

Dataset	T	α_1	$n_1, n_d (d \geqq 2)$
Dataset A	540	$600, 1200, \ldots, 43200$	$0.00, 0.01, \ldots, 1.00$
Dataset B	370	$600, 1200, \ldots, 43200$	$0.00, 0.01, \ldots, 1.00$

c) No noise removal: A smart home may have multiple users and monitored event sequences may include the operation of multiple users. Thus, noise removal when learning the event sequences is also one of important aspects of our method. By removing noise (i.e., spurious events), the method learns essential event sequences. To investigate the effectiveness of noise removal, we also compared the proposed method with its variant without removing noise. Anomaly detection proceeded in the same manner for both compared methods. Specifically, when events A, B, and C are monitored, the comparison variant only learns the sequence A, B, C, whereas the proposed method learns the sequences generated by removing noise shown in Fig. 2.

d) Only sequence: This method simply learns user behaviors based on only sequences of events without noise removal method similar to the existing methods that learn user behaviors. By comparing our method with this method, we demonstrate the advantages of our method over the existing methods.

C. Evaluation Parameters

The proposed method has three types of parameters, T, α_1, n_1, and n_d. We set T according to the procedure in our previous work [18]. We adjusted the other parameters and evaluated the method to obtain the misdetection and detection ratios, whose relations were depicted as receiver operating characteristic curves. Table III lists all the evaluation parameters for this study.

D. Results

Fig. 4 shows the receiver operating characteristic curves of the proposed method and five compared methods, where the detection ratio is plotted according to the misdetection ratio. The proposed method detects more than 90% of attacks, whereas misdetection reaches only 10% of the legitimate operations for most devices.

The detection ratio of the variant using only the condition (i.e., time of day) is much lower than that of the proposed method. This confirms that using the event sequences effectively improves detection.

The detection ratio of the variant without noise removal is much lower than that of our method. Hence, noise removal is necessary to learn essential event sequences. For example, we observed the following sequence: user 2 enters the room, opens the refrigerator, and operates electric fan B, but the refrigerator was operated by another user. Hence, the event sequence was not essential, which corresponded to user 2 entering the room and operating electric fan B, leading to a misdetection for the legitimate fan operation. In addition, the detection ratio of the HMM method is much lower

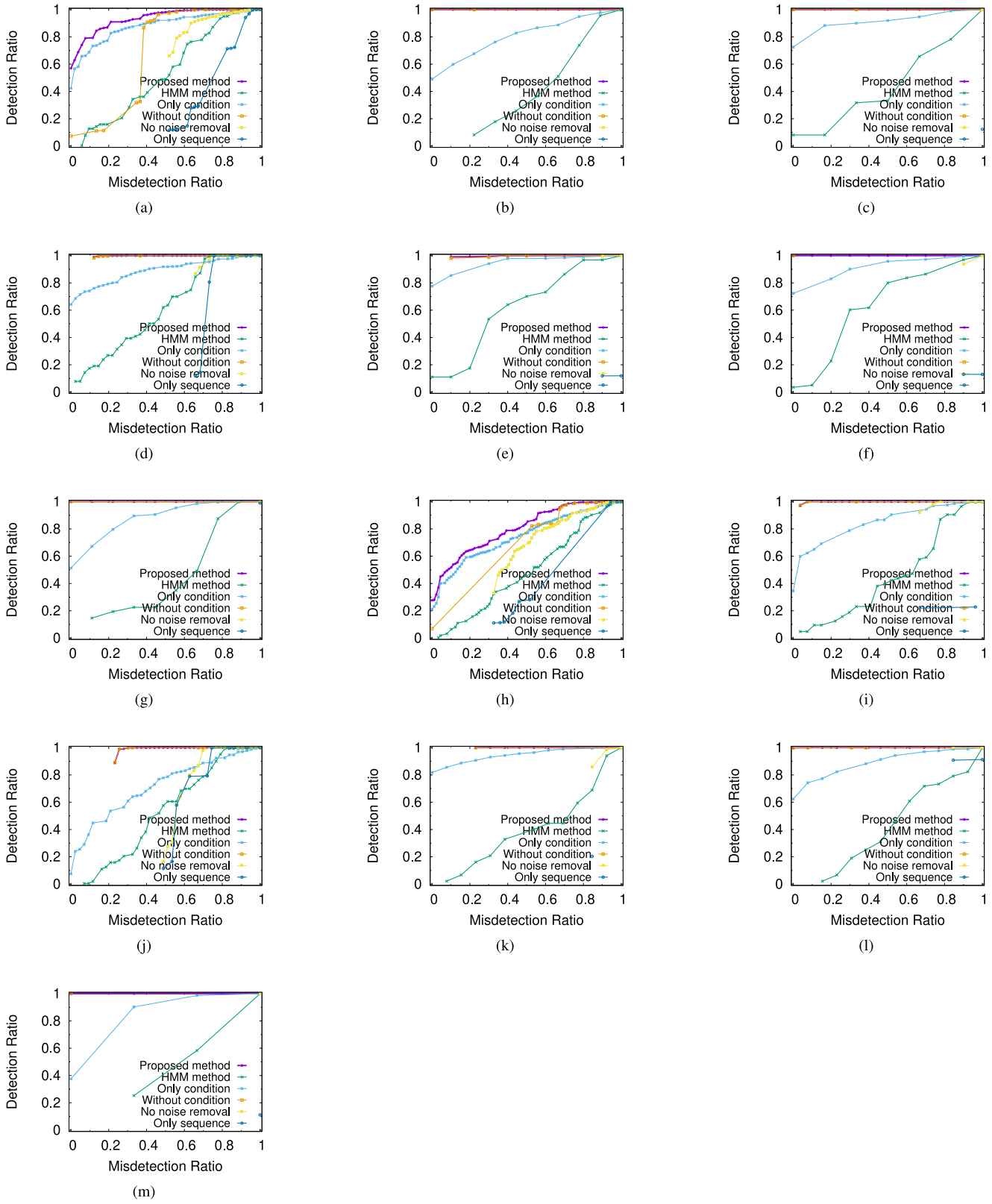

Fig. 4. Receiver operating characteristic curves of detection using the proposed method and comparison variants: (a) coffee maker in dataset A, (b) electric fan A in dataset A, (c) electric fan B in dataset A, (d) TV A in dataset A, (e) TV B in dataset A, (f) TV C in dataset A, (g) TV D in dataset A, (h) coffee maker in dataset B, (i) electric fan A in dataset B, (j) electric fan B in dataset B, (k) TV A in dataset B, (l) TV B in dataset B, and (m) TV C in dataset B.

than that of our method. This is because, the HMM method does not consider that multiple users are in the home. In the case that multiple users are in the home, the number of states required to model the home is significantly large. On the other hand, the number of observed operations on home IoT devices is limited. As a result, we cannot estimate

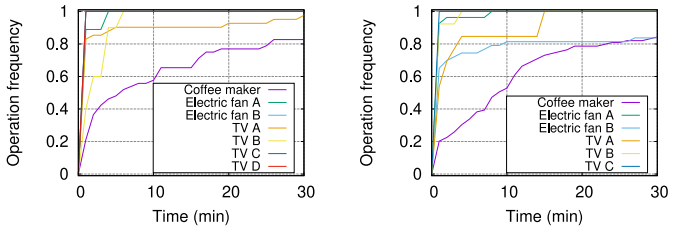

Fig. 5. Distribution of time between an operation of each device and its nearest event: (a) Dataset A, (b) Dataset B.

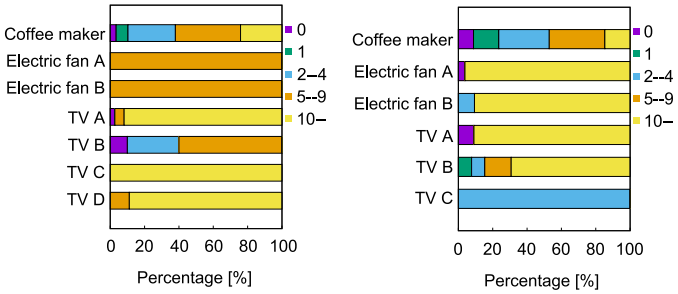

Fig. 6. Frequency of matching behaviors: (a) Dataset A, (b) Dataset B.

the proper parameters of the HMM due to the lack of the observations.

The proposed method achieved similar detection ratios to the method using the sequence information with noise removal except for the results of coffee makers. That is, the effectiveness of considering condition information was not large. This is because we used only time of day to define the conditions. If we use the other sensors to define the conditions, the difference may become larger.

Fig. 4 shows that attacks in some devices are not accurately detected by the proposed method. For example, the detection ratio of attacks for the coffee maker of the proposed method is similar to that of its variant using only the time condition. Moreover, the proposed method cannot take a misdetection ratio for TVs A and B in dataset A and electric fan B and TV A in dataset B below 10%.

To investigate the difference between devices whose attacks are accurately detected and those whose attacks cannot be accurately detected, we further investigated the usage of these devices. Fig. 5 shows whether the operation of each device is included in event sequences, including multiple events. Specifically, we plot the distribution of time between an operation of each device and its nearest event. Next, we determine the frequency of the monitored event sequences. We first stored the event sequences in the training dataset. Then, we compared the event sequences in the test dataset with the stored event sequences and generated the event sequences per operation in the test dataset by removing noise, as explained in Section III-B1. Finally, we determined the number of stored event sequences matching each generated event sequence and obtained the maximum count for each operation in the test dataset. Fig. 6 shows the distribution of the number of matching event sequences.

From the figures above, we identified the devices whose attacks cannot be accurately detected. Figure 5 clarifies the devices that tend to be used alone. The elapsed time between an operation of the coffee maker and another event is large, compared with those of other devices. 44% of coffee maker's operations in dataset A and 62% of those in dataset B have no previous or next events within T seconds. For devices that tend to be used alone, the event sequence approach does not function properly. Consequently, the detection ratio of the proposed method for such devices is similar to that of the variant using only the condition. Due to the similar reason, the detection ratio of attacks for the electronic fan B in dataset B becomes also similar to that of the method using only the condition when the parameters set so as to make the misdetection ratio smaller than 0.28. Fig. 5 shows more than 25% of the operations of the electronic fan B in dataset B have no previous or next events within T seconds. As a result, parameters in our method are set to avoid misdetection of single operations if the misdetection ratio should be less than 0.28. One approach to prevent anomalous operations in such devices is to deploy sensors to thoroughly monitor their operation. Another approach to improve the detection accuracy of legitimate operations is to use more information to define the conditions.

Another kind of devices that cannot be detected accurately includes TVs A and B in dataset A and electric fan B and TV A in dataset B. Our method detects more than 90% of the attacks for these devices but cannot make the misdetection ratio less than 10%. This is caused by rare event sequences. Fig. 6 shows that some event sequences generated for such devices do not match any previously stored event sequences. Consequently, such rare event sequences are detected. Storing more data can avoid misdetection in such devices, but only a limited number of operations are monitored in each home. Hence, using data from several homes can help provide more training data.

E. Discussion

Our method can be applied to a system like an intrusion prevention system. This system monitors the commands to home IoT devices and detects anomalous operations by our method. If an anomalous operation is detected, the system avoids anomalous operations by dropping the packets related to the operations. In this system, the detection ratio is more important than the misdetection ratio, because if an attack is not detected, the home IoT devices operated by an attacker may cause immediate and personal harm to users. If a legitimate operation by a user is mistakenly detected as an anomalous operation, the command from the user is dropped. However, we can implement a mechanism to enable users to operate the IoT devices even in the case of misdetections. For example, we can send notification of the detection to the users' smartphone and allow operations temporally after authentication by different factors. Therefore, the parameters should be set so as to achieve high detection ratios.

In the evaluation in Section IV, if we set parameters so as to detect more than 99% of anomalous operations, our method

misdetects about 6 legitimate operations per month except for the operations of the coffee maker. However, the operations of the coffee maker are misdetected many times. As discussed in Section IV, it is difficult to distinguish the legitimate operations of the coffee maker from anomalous operations, because most of the operations of the coffee makers are single operations. As discussed above, the impact of the misdetection is smaller than the attacks that cannot be detected. However, the frequent misdetection makes users uncomfortable. Therefore, we need to improve the accuracy of the detection, which is one of our future work.

We should also discuss the possibility that attackers change their attacks so as to avoid detection by our method. The attackers might mimic the behaviors of legitimate users. In this case, our method cannot detect the command sent by the attackers. To mimic the behavior of legitimate users, the attackers require the knowledge of the behavior of legitimate users and the ability to control the devices included in the legitimate event sequence. Especially if many users use a device in the same manner, the attackers may easily infer the behavior of legitimate users. To avoid such attacks, we need to (1) protect learned user behaviors so that attackers cannot obtain the information, (2) make the model complicated so that the attacker cannot infer the behavior by adding more sensors, including more events and so on, and (3) include the devices that are difficult to be operated by the attackers in the model. The robust method against such attacks is also one of our future work.

Another point to be discussed is the number of users in a home. Our method considers the case that multiple users in a home, and avoids misdetection by learning essential behaviors of all users. By doing so, our method avoids misdetections even if many users are in a home and behave differently from each other. However, as the number of users who behave differently from each other increases, more event sequences are stored as legitimate sequences. Even if a large number of event sequences are stored, it is difficult for attackers to operate the home IoT devices so that the event sequence including the attack matches the learned event sequences, if the model of the event sequence is complicated enough to avoid the inference of the legitimate behavior.

V. Conclusion

We validate the effectiveness of our method for detecting anomalous operations of home IoT devices by comparing it with an existing method and some of its variants. The proposed method can learn sequences of user behaviors according to conditions such as time of day, temperature, and humidity. Then, when an operation command arrives, the method compares the current sequence with learned sequences for the current condition. If the sequences do not match, the operation is considered as anomalous. We constructed a network of home IoT devices in our laboratory and allowed four subjects to operate the devices for 3 months. We recorded the times at which the devices were operated along with sensor data. Using these data, we evaluated the detection and misdetection rates of the proposed method and its variants considering only the condition or without removing spurious events. The proposed method can detect over 90% of anomalous operations with less than 10% of misdetections if the events related to legitimate operations can be monitored. Therefore, we found that the most effective way to learn user behaviors in homes for the detection of anomalous operation is by learning event sequences and user habits when entering and leaving rooms. In addition, noise (i.e., spurious event) removal is necessary for improved detection. When single operations that do not correspond to observed sequences occur, the proposed method achieves a higher accuracy by learning sequences executed multiple times than by using only condition information.

However, the proposed detection method can produce a high rate of misdetections, especially when single or rare operations occur. In fact, as the method achieves accuracy by comparing event sequences, the anomaly detection of isolated and rare events depends only on the operation condition. To improve the detection accuracy for such operations, we can deploy sensors that monitor events related to these operations. Alternatively, we can use more information to define conditions. In our evaluation, we used only the time of day to define conditions. However, defining more representative conditions to distinguish legitimate operations can lead to improved detection accuracy. We will explore methods to improve legitimate single operation detection in future research.

Mitigating the misdetection of rare legitimate operations is another challenge, as we cannot obtain sufficient training data to accurately identify such rare operations in each home. To obtain more training data, we will use data from several homes in future work. However, some problems remain to be solved before achieving this type of data collection. For instance, different homes and environments and varying user behaviors may render the collected data useless. Thus, we need to gather data from several homes whose users exhibit similar behaviors. Moreover, as privacy is a major concern, we should use anonymized data from different homes to preserve information privacy.

References

[1] K. L. Lueth. (Aug. 2018). *State of the IoT2018: Number of IoT devices Now at 7B–Market Accelerating*. [Online]. Available: https://iot-analytics.com/state-of-the-iot-update-q1-q2-2018-number-of-iot-devices-now-7b/

[2] I. Lee and K. Lee, "The Internet of Things (IoT): Applications, investments, and challenges for enterprises," *Bus. Horizons*, vol. 58, no. 4, pp. 431–440, Jul. 2015.

[3] M. U. Farooq, M. Waseem, A. Khairi, and S. Mazhar, "A critical analysis on the security concerns of Internet of Things (IoT)," *Int. J. Comput. Appl.*, vol. 111, no. 7, pp. 1–6, Feb. 2015.

[4] B. L. R. Stojkoska and K. V. Trivodaliev, "A review of Internet of Things for smart home: Challenges and solutions," *J. Clean. Prod.*, vol. 140, no. 3, pp. 1454–1464, Jan. 2017.

[5] M. Capellupo, J. Liranzo, M. Z. A. Bhuiyan, T. Hayajneh, and G. Wang, "Security and attack vector analysis of IoT devices," in *Proc. Int. Conf. Security Privacy Anonymity Comput. Commun. Stor.*, Dec. 2017, pp. 593–606.

[6] M. Antonakakis *et al.*, "Understanding the Mirai botnet," in *Proc. 26th USENIX Security Symp.*, Vancouver, BC, USA, Aug. 2017, pp. 1093–1110.

[7] D. Palmer. (May 2017). *120,000 IoT Cameras Vulnerable to New Persirai Botnet Say Researchers*. [Online]. Available: https://www.zdnet.com/article/120000-iot-cameras-vulnerable-to-new-persirai-botnet-say-researchers/
[8] Y. M. P. Pa, S. Suzuki, K. Yoshioka, T. Matsumoto, T. Kasama, and C. Rossow, "IoTPOT: A novel honeypot for revealing current IoT threats," *J. Inf. Process.*, vol. 24, no. 3, pp. 522–533, May 2016.
[9] M. Lyu, D. Sherratt, A. Sivanathan, H. H. Gharakheili, A. Radford, and V. Sivaraman, "Quantifying the reflective DDoS attack capability of household IoTdevices," in *Proc. 10th ACM Conf. Security Privacy Wireless Mobile Netw. (WiSec)*, New York, NY, USA, Jul. 2017, pp. 46–51.
[10] K. Xu, F. Wang, and X. Jia, "Secure the Internet, one home at a time," *Security Commun. Netw.*, vol. 9, no. 16, pp. 3821–3832, Jul. 2016.
[11] V. Martin, Q. Cao, and T. Benson, "Fending off IoT-hunting attacks at home networks," in *Proc. 2nd Workshop Cloud-Assist. Netw.*, Dec. 2017, pp. 67–72.
[12] S. Shirali-Shahreza and Y. Ganjali, "Protecting home user devices with an SDN-based firewall," *IEEE Trans. Consum. Electron.*, vol. 64, no. 1, pp. 92–100, Feb. 2018.
[13] K. Xu, F. Wang, R. Egli, A. Fives, R. Howell, and O. Mcintyre, "Object-oriented big data security analytics: A case study on home network traffic," in *Proc. Int. Conf. Wireless Algorithms Syst. Appl.*, Jun. 2014, pp. 313–323.
[14] K. Xu, F. Wang, L. Gu, J. Gao, and Y. Jin, "Characterizing home network traffic: An inside view," *Pers. Ubiquitous Comput.*, vol. 18, no. 4, pp. 967–975, Apr. 2014.
[15] N. Komninos, E. Philippou, and A. Pitsillides, "Survey in smart grid and smart home security: Issues, challenges and countermeasures," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 4, pp. 1933–1954, 4th Quart. 2014.
[16] S. Soltan, P. Mittal, and H. V. Poor, "BlackIoT: IoTbotnet of high wattage devices can disrupt the power grid," in *Proc. 27th USENIX Security Symp.*, Baltimore, MD, USA, Aug. 2018, pp. 15–32.
[17] B. B. Zarpelao, R. S. Miani, C. T. Kawakani, and S. C. de Alvarenga, "A survey of intrusion detection in Internet of Things," *J. Netw. Comput. Appl.*, vol. 84, pp. 25–37, Apr. 2017.
[18] M. Yamauchi, Y. Ohsita, M. Murata, K. Ueda, and Y. Kato, "Anomaly detection for smart home IoT based on users' behavior," in *Proc. IEEE Int. Conf. Consum. Electron.*, Las Vegas, NV, USA, Jan. 2019, pp. 1–6.
[19] S. Ramapatruni, S. N. Narayanan, S. Mittal, A. Joshi, and K. Joshi, "Anomaly detection models for smart home security," in *Proc. IEEE 5th Int. Conf. Big Data Security Cloud IEEE Int. Conf. High Perform. Smart Comput. IEEE Int. Conf. Intell. Data Security*, May 2019, pp. 19–24.
[20] E. Hodo *et al.*, "Threat analysis of IoT networks using artificial neural network intrusion detection system," in *Proc. IEEE Int. Symp. Netw. Comput. Commun.*, May 2016, pp. 1–6.
[21] T. Rashid, I. Agrafiotis, and J. R. Nurse, "A new take on detecting insider threats: Exploring the use of hidden Markov models," in *Proc. 8th ACM CCS Int. Workshop Manag. Insider Security Threats (MIST)*, New York, NY, USA, 2016, pp. 47–56.
[22] W. Haider, J. Hu, Y. Xie, X. Yu, and Q. Wu, "Detecting anomalous behavior in cloud servers by nested-arc hidden semi-Markov model with state summarization," *IEEE Trans. Big Data*, vol. 5, no. 3, pp. 305–316, Sep. 2019.
[23] O. Aran, D. Sanchez-Cortes, M.-T. Do, and D. Gatica-Perez, "Anomaly detection in elderly daily behavior in ambient sensing environments," in *Proc. Int. Workshop Human Behav. Understand.*, Sep. 2016, pp. 51–67.
[24] C. M. Bishop, *Pattern Recognition and Machine Learning* (Information Science and Statistics). New York, NY, USA: Springer, Feb. 2006.

Masaaki Yamauchi (Graduate Student Member, IEEE) received the B.E. and M.E. degrees in information science and technology from Osaka University, Japan, in 2017 and 2019, respectively, where he is currently pursuing the Ph.D. degree with the Graduate School of Information Science and Technology.

His research interest includes network security.

Yuichi Ohsita (Member, IEEE) received the M.E. and Ph.D. degrees in information science and technology from Osaka University, Japan, in 2005 and 2008, respectively.

Since January 2019, he has been an Associate Professor with the Institute for Open and Transdisciplinary Research Initiatives, Osaka University. His research interests include traffic engineering, traffic prediction, and network security.

Dr. Ohsita is a member of IEICE and the Association for Computing Machinery.

Masayuki Murata (Member, IEEE) received the M.E. and D.E. degrees in information and computer science from Osaka University, Japan, in 1984 and 1988, respectively.

Since April 2004, he has been a Professor with the Graduate School of Information Science and Technology, Osaka University. His research interests include information network architecture, performance modeling, and evaluation.

Prof. Murata is a member of ACM and IEICE.

Kensuke Ueda received the B.E. and M.E. degrees in information science and technology from Osaka University, Japan, in 2004 and 2006, respectively.

In April 2006, he joined the Advanced Technology R&D Center, Mitsubishi Electric Corporation, Japan. His research interests include home network system and services.

Mr. Ueda is a member of IPSJ.

Yoshiaki Kato (Senior Member, IEEE) received the B.E. and M.E. degrees in applied physics and the Ph.D. degree in information science and technology from Osaka University, Japan, in 1984, 1986, and 2010, respectively.

Since April 2016, he has been the Chief Researcher with the Information Technology R&D Center, Mitsubishi Electric Corporation and the Deputy Director of the Mitsubishi Electric Cybersecurity Research Alliance Laboratories.

Dr. Kato is a Senior Member of IEICE.