2019 IEEE International Conference on Consumer Electronics
(ICCE)

Anomaly Detection for Smart Home Based on User Behavior

Masaaki Yamauchi*, Yuichi Ohsita*, Masayuki Murata*, Kensuke Ueda†, Yoshiaki Kato‡
* *Graduate School of Information Science and Technology, Osaka University.*
Email: {m-yamauchi, y-ohsita, murata}@ist.osaka-u.ac.jp
† *Advanced Technology R&D Center, Mitsubishi Electric Corporation.*
Email: Ueda.Kensuke@ce.MitsubishiElectric.co.jp
‡ *Information Technology R&D Center, Mitsubishi Electric Corporation.*
Email: Kato.Yoshiaki@dh.MitsubishiElectric.co.jp

***Abstract*—Many devices, such as air conditioners and refrigerators, are now being connected to the Internet and, as a consequence, have become targets of cyberattacks. Especially, the operations by attackers can cause serious problems, which may harm users. However, such attacks are difficult to detect because they use the same protocol as legitimate operations by users. In this paper, we propose a method to detect such attacks based on user behavior. We model user behavior as a sequence of events, which includes the operation of IoT devices and other behavior monitored by any sensors. Our method learns sequences of events for each one of a predefined set of conditions and detects attacks by comparing the sequences of the events including the current operation with the learned sequences. We evaluate our method by using data collected by monitoring the behavior of four users. Based on the results of this evaluation, we demonstrate the accuracy of our method and discuss the limitations of our method.**

***Index Terms*—Anomaly Detection, IoT, Security, Smart Home, Behavior Pattern, Operation by Attackers, Consumer Electronics**

I. Introduction

Recently, consumer electronics, such as refrigerators, air conditioners, and pacemakers, have started to be connected to the Internet in addition to personal computers and smartphones. These devices are called "IoT (Internet of Things)" devices. Users can obtain information from the IoT devices and can operate the IoT devices with using a smartphone, a tablet, or an AI speaker, such as Google Home [1] or Amazon Echo [2].

As the number of devices connected to the Internet increases, the risk that these devices become the target of cyberattacks is increasing [3]–[6] and, in fact, direct attacks and malware targeting IoT devices [7], [8] have already been observed. Such attacks may be detectable by methods based on an analysis of the behavior of the attackers [9]–[11] or in comparison with legitimate usage [12], [13]. Most of current attacks targeting IoT devices are designed to compromise IoT devices to create botnets [14], [15]. However, because IoT devices are closely related to everyday life, there is a risk of attacks having an immediate and personal effect on users [16]. In particular, the operation of IoT devices by attackers may make the users unsafe, and may even harm them, by changing the set temperature of an air conditioner, the setting of a healthcare device, or similar. Therefore, methods to detect and prevent operations initiated by attackers are necessary.

Conventionally, security software and intrusion detection systems are used to detect cyberattacks. They detect attacks by pattern matching, comparing the packets with predefined rules. However, packets sent by attackers to operate IoT devices are the same as the packets sent by legitimate users. In particular, if an attacker sends packets via the compromised smartphone of a legitimate user, the packets cannot be distinguished from packets sent by legitimate users.

In this paper, we propose a method to detect the anomalous operation of home IoT devices at a home gateway that can monitor all packets between the home network and the Internet. In this method, the gateway learns the behavior of users for each condition defined by the time of day and the information observed by the sensors in the home network. Then, when a command arrives, the gateway checks whether it matches the learned behavior. If the command does not match the learned behavior, the gateway classifies the command as an anomaly.

To evaluate our method, we constructed a network of home IoT devices in our laboratory. We then selected four students as subjects and let them use the IoT devices as they liked. We monitored the home network and recorded the times of control commands given to the IoT devices. To evaluate our method using the data collected; we use the monitored commands as the legitimate commands and investigate whether our method can detect anomalous commands added to the monitored data.

The rest of this paper is organized as follows. We describe our method to detect anomalous operations in section II and explain how to set the parameters in section III. Then, we evaluate our method in section IV. Finlay, we conclude this work and discuss future issues in section V.

II. Anomaly Detection for Smart Home

We propose a method to detect the anomalous operation of home IoT devices by attackers. It is based on that users have their own patterns of behavior depending on conditions. For example, when a user returns home and feels cold, they turn on a heater first and then turn on a humidifier; on the other hand, when a user feels warm, they never turn on a heater. In addition, the sequence of operations reflects the characteristics

978-1-5386-7910-4/19/$31.00 ©2019 IEEE

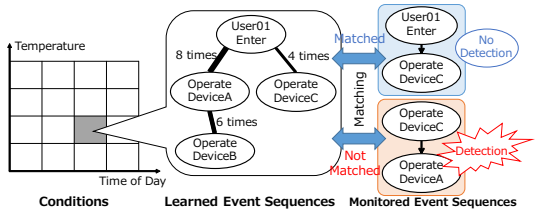

Fig. 1: Overview of the detection model.

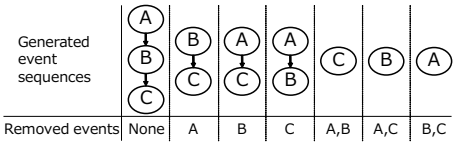

Fig. 2: Generated event sequences when events A, B, and C occur.

of users: one user first turns on a heater and then turns on a humidifier, while another user first turns on a humidifier.

Our method learns such condition-depend operation sequences as behavior patterns and classifies deviations from these learned patterns as anomalous operations.

In this section, we first explain the model of user behavior used in our method. Then, we explain how user behavior is learned and how anomalous operations are detected.

A. Model

Fig. 1 shows the model of user behavior in our method.

1) Condition: We define a condition as a combination of the time of day and values obtained from sensors. The variables representing the various components of a condition will be denoted c_i, where the index i runs from 1 to some maximum value i^{max}. For example, as in Fig. 1, c_1 could represent time of day, while c_2 represents the temperature of a room, and so on. Of course, continuous data have to be discretized for tractability. So, we use multiple thresholds for each type of value. A value of c_i that satisfies $c_i^{(j)} \leq c_i \leq c_i^{(j+1)}$ where $c_i^{(j)}$ is the jth threshold for the ith variable, is classified into jth region for that variable.

2) User Behavior: We define an *event* as any monitored behavior of a user, including the operation of IoT devices and any other behavior monitored by the sensors, such as entering or leaving a room. We define an *event sequence* to be a sequence of events that occurs within T seconds of a previous event. Our method learns a user's behavior by learning the event sequences for each condition. The set of event sequences corresponding to a condition is modeled as multiple trees whose roots are the first events and whose leaves are the last events. We detect anomalous event sequences by checking whether the sequence is included in trees. If the sequence is not included in the tree, we classify the sequence as an anomaly.

B. Learning Method

Our method learns user behavior from observed information by monitoring events in the home and generating event sequences. Then, the condition corresponding to an event sequence is selected from the predefined set. Finally, the observed sequence is used to update the model for the corresponding condition.

1) Generation of Event Sequences: Our method regards events observed within T seconds of one another as belonging to the same event sequence, constructing the sequence at the home gateway. However, this means that some event sequences may include events from more than one user because a typical smart home has multiple users. To remove these events, which should be treated as noise, and learn the essential event sequence, we generate multiple event sequences by removing some events from the observed sequence. Fig. 2 shows an example of the generation of event sequences. By considering these event sequences, we discover the essential event sequences that include only the events generated by a single user.

2) Selection of Condition: Our method selects the condition whose corresponding model is to be updated based on the condition of the first event in a sequence. However, to make effective use of event sequences and learn users' behaviors even when only a small number of event sequences for each condition are observed, we update not only the model corresponding to the condition of its first event but also the models for similar conditions. When the condition of the first event is $\{c_1^{root}, \ldots, c_{i^{max}}^{root}\}$, the sequence is used to update the model for the region for which the variable c_i satisfies $c_i^{root} - \alpha_i \leq c_i \leq c_i^{root} + \alpha_i$ for some value α_i, which can be different for each i.

3) Updating the Tree of Event Sequences: When the home gateway observes an event sequence, we add nodes and links to the tree corresponding to the selected conditions so that the event sequence is included in the tree, the first event of the sequence is the root, and the final event of the sequence is the leaf. Then, we increment a counter for each link on the route corresponding to the event sequence. By repeating this procedure, the count for links related to frequent event sequences increases. Thus, we detect anomalies by using the pruned tree which contains only the links whose counters exceed a threshold $n_d \times L_{num}$ where n_d is a parameter, d is the depth of the link, and L_{num} is the total number of learned operations for the device targeted for detection. By doing so, we eliminate the noise included in the event sequence.

C. Detection Method

When the home gateway observes a new operation, it generates event sequences related to the operation. Then, it compares these sequences with learned behaviors.

1) Generation of Event Sequence: When the home gateway observes a new operation, it generates multiple event sequences, in the same way as during learning, by removing some events from the sequence of events that occurred within T seconds after previous event and uses these to check whether the operation is anomalous.

2) Decision making: We check whether the new operation is anomalous or not by comparing the generated event sequences with the learned behaviors. We first select the condition corresponding to the first operation of the generated sequences. Then, we compare each of the generated sequences with the learned sequences for the selected condition by finding the events in the tree. First, we find the tree whose root is the first event of the event sequence. Then, we find the node that is connected to the found root and is the second event of the event sequence. We repeat searching for nodes until nodes corresponding to all events in the sequence are found or no corresponding nodes are found. We repeat the check for all generated event sequences using the above steps. If all sequences have events whose corresponding nodes are not found in the learned behavior, we classify the operation as an anomaly. On the other hand, if corresponding nodes are found for all events in one of the sequences and the node corresponding to the final event is the leaf, we decide that all the events in the sequence are legitimate. If corresponding nodes are found for all events in one of the sequences but the node corresponding to the final event is not the leaf, we wait for the next event. If the next event does not occur for T seconds, we finish searching for the sequence. If the next event occurs, we generate event sequences including the next event and compare the sequences with learned behaviors using the same steps as described above.

III. Parameter Setting

In this section, we describe the method used to set parameters. Our method has three kinds of parameter, T, α_i, and n_d. T affects the generation of event sequences, while α_i and n_d affect the model used for the detection. Thus, we set the parameter T independently from the other parameters so as to generate suitable event sequences. To set parameters, we use events monitored for P weeks. The information on the monitored events includes the kind of event, the time, and the sensor values defining the conditions corresponding to the events.

1) Setting the Parameter T: First, we set the parameter T. A user may operate multiple devices simultaneously or within a small interval of time. In this paper, we set T so that the operations of such related devices are included in the event sequences.

Algorithm 1 shows the steps used to set T. In these steps, T is incremented until the generated event sequences become stable. We check whether the generated event sequences are stable by counting the number of pairs of events included in the same event sequence using Algorithm 2. The number of pairs of events included in the same event sequence (*NumOfPair*(T)) initially increases with T. However, the number becomes stable when T is large enough that the operations of related devices are included in the same sequences. Algorithm 2 regards the generated event sequences as stable when the difference between *NumOfPair*(T') and *NumOfPair*$(T' + T_{inc})$ is less than N for all T' satisfying $T \leq T' < T + T_{inc}^{stable}$, where T_{inc}^{stable} is a preset parameter.

Algorithm 1 Setting the parameter T

Input: P: Period of the parameter setting (weeks)
Output: T
  $T \Leftarrow T_{init}$
  while IsNumberOfSequencesStable(T) == *False* **do**
    $T \Leftarrow T + T_{inc}$
  end while
  return T

Algorithm 2 Whether generated event sequences are stable

Input: T
Output: *True* | *False*
  function IsNumberOfSequencesStable(T)
    $T' \Leftarrow T$
    while $T - T' \leq T_{inc}^{stable}$ **do**
      $T' \Leftarrow T' + T_{inc}$
      if *NumOfPair*$(T' + T_{inc})$ - *NumOfPair*$(T') \geq N$ **then**
        return *False*
      end if
    end while
    return *True*
  end function

2) Setting the Parameters α_i and n_d: After setting the parameter T, we set the parameters $\boldsymbol{\alpha}$ and $\boldsymbol{n}$, where $\boldsymbol{\alpha} = \{\alpha_1, \cdots, \alpha_{i^{max}}\}$ and $\boldsymbol{n} = \{n_1, \cdots, n_{d^{max}}\}$. i^{max} is the number of values used to define the condition, and d^{max} is a predefined value indicating the depth of the tree considered when setting parameters. For the links whose depth d' is larger than d^{max}, we use $n_{d^{max}}$ as $n_{d'}$.

The parameters $\boldsymbol{\alpha}$ and $\boldsymbol{n}$ have impacts on the detection ratio (D), that is the proportion of anomalous operations detected as anomalies, and misdetection ratio (M), that is the proportion of legitimate operations detected as anomalies. In this paper, we set these parameters to obtain the highest detection ratio from among the values that given an acceptable misdetection ratio, $M \leq M^{goal}$. In order to calculate the detection and misdetection ratio, we perform the "Detection Test" shown as Algorithm 4. The detection test calculates the detection and misdetection ratio when anomalous operations of IoT devices are added to event datasets by using cross-validation. At first, we use event datasets to learn behaviors except for a certain day. Then, we add 100 anomalous operations in the omitted day and count the number of detected anomalous operations and misdetected legitimate operations. After doing this for each day, we sum up the numbers of both and calculate the detection and misdetection ratio.

Algorithm 3 shows the steps followed to set parameters $\boldsymbol{\alpha}$ and $\boldsymbol{n}$, and we obtain optimal parameters by calling Opt$\boldsymbol{\alpha n}$(*Null*, *Null*, $k = 1, M^{goal}$). This function finds optimal parameter values by recursively calling itself with k incremented by 1. This function finds optimal parameters by fixing

the k'th parameters for $k' < k$. As α_i becomes large, the misdetection ratio becomes small, because the observed event sequences are used to update a larger range of conditions and similar event sequences are regarded as legitimate even if the condition is slightly different from that of the learned sequence. Therefore, if we cannot achieve a misdetection ratio less than M^{goal} by setting $\alpha_i = \alpha'_i$, an acceptable misdetection ratio cannot be achieved for $\alpha_i \leq \alpha'_i$. Similarly, as n_d becomes small, more event sequences are used as legitimate behavior, and the misdetection ratio becomes small. That is, if we cannot achieve a misdetection ratio less than M^{goal} by setting $n_d = n'_d$, an acceptable misdetection ratio cannot be achived for $n_d \geq n'_d$. Therefore, to reduce the calculation time, we avoid considering such parameters in Algorithm 3.

If no parameters satisfy the target misdetection ratio, our algorithm cannot find suitable parameters. When such cases occurred in our evaluation, we increased the misdetection ratio until suitable parameters could be found.

IV. Evaluation

A. Evaluation Environment

1) Evaluation Dataset: To evaluate our method, we constructed a network of home IoT devices in our laboratory. We deployed 15 kinds of connectable home appliances and sensors: a heater, a coffee maker, a refrigerator, electric fans, TVs, and temperature sensors. We let four subjects use the appliances as they liked. We captured all packets in the home network and recorded the times of operations of the home IoT devices and sensor data. The sensor data includes information on whether smartphones are connected to the home network. From this information, we generate the events that a user enters or leaves the home. The sensor data also includes the temperature, humidity, and noise values; however, these do not change significantly in our environment. Therefore, in this evaluation, only the time of day is used to define the condition.

2) Evaluation Procedure: In this section, we use two metrics: the detection ratio and the misdetection ratio.

a) Misdetection Ratio: To evaluate the misdetection ratio in a straightforward way, we would need a sufficient number of legitimate operations in the test data. However, we have only a limited number of legitimate operations in our collected data. Therefore, we evaluate the misdetection ratio by Leave-One-Out Cross-Validation (LOOCV) [17]. LOOCV separates the dataset into multiple small datasets and verifies one of them after training using the others. After verifying all patterns of the test dataset, we summarize the results.

For this paper, we separated the data by day and used the data for one day as the test data and used the data for the other days to learn legitimate behaviors.

b) Detection Ratio: In our evaluation, we added anomalous operations into observed legitimate operations to obtain the detection ratio. We used a strategy similar to that for the evaluation of misdetection to obtain the detection ratio: we separated the dataset by day and used the data for one day for the test and the remaining data for learning.

Algorithm 3 Optimal $\boldsymbol{\alpha}$, $\boldsymbol{n}$, and Detection-Misdetection Ratio

Input: $\boldsymbol{\alpha^{in}}, \boldsymbol{n^{in}}, k, M^{goal}$
Output: $D, M, \boldsymbol{\alpha}^{best}, \boldsymbol{n}^{best}$
  function OPT$\boldsymbol{\alpha n}(\boldsymbol{\alpha}^{in}, \boldsymbol{n}^{in}, k, M^{goal})$
    for $0 < k' < k$ **do**
      if $k' \leq i^{max}$ **then**
        $\alpha_{k'} \Leftarrow \alpha^{in}_{k'}$
      else
        $n_{k'-i^{max}} \Leftarrow n^{in}_{k'-i^{max}}$
      end if
    end for
    $(D^{best}, M^{best}) \Leftarrow (0.0, 1.0)$
    if $k \leq i^{max}$ **then**
      $\alpha_k \Leftarrow \alpha^{max}_k$
    else
      $n_{k-i^{max}} \Leftarrow n^{min}_{k-i^{max}}$
    end if
    while *True* **do**
      if $k < i^{max} + d^{max}$ **then**
        $(D', M', \boldsymbol{\alpha'}, \boldsymbol{n'}) \Leftarrow$OPT$\boldsymbol{\alpha n}(\boldsymbol{\alpha}, \boldsymbol{n}, k+1, M^{goal})$
      else
        $(D', M') \Leftarrow$ DETECTIONTEST$(\boldsymbol{\alpha}, \boldsymbol{n})$
        $(\boldsymbol{\alpha'}, \boldsymbol{n'}) \Leftarrow (\boldsymbol{\alpha}, \boldsymbol{n})$
      end if
      if $M' > M^{goal}$ **then**
        Break
      else if $D' > D$ **then**
        $(D^{best}, M^{best}, \boldsymbol{\alpha}^{best}, \boldsymbol{n}^{best}) \Leftarrow (D', M', \boldsymbol{\alpha'}, \boldsymbol{n'})$
      end if
      if $k \leq i^{max}$ **then**
        $\alpha_k \Leftarrow \alpha_k - \alpha^{dec}_k$
        if $\alpha_k < \alpha^{min}_k$ **then**
          Break
        end if
      else
        $n_{k-i^{max}} \Leftarrow n_{k-i^{max}} + n^{inc}_{k-i^{max}}$
        if $n_{k-i^{max}} > n^{max}_{k-i^{max}}$ **then**
          Break
        end if
      end if
    end while
    return $(D^{best}, M^{best}, \boldsymbol{\alpha}^{best}, \boldsymbol{n}^{best})$
  end function

We added 100 anomalous operations into the test data for each day and tried to detect them. Then, we obtain the detection ratio by counting the total number of detected anomalous operations for all days and dividing by $100 \times$ *number of days of observations*.

B. Evaluation Results

We evaluated three datasets; the dataset obtained for one month in January 2017, and the datasets obtained for the three months of April, June, and August 2017 and the three months of May, July, and September 2017. In each period, the same

Algorithm 4 Detection Test

Input: $\boldsymbol{\alpha}, \boldsymbol{n}$
Output: D, M
  function DETECTIONTEST($\boldsymbol{\alpha}, \boldsymbol{n}$)
    for each *Day* $\in$ Period of parameter setting **do**
      Insert 100 anomaly operations into the *Day*
      Learn users' behavior without the *Day* with $\boldsymbol{\alpha}$
      Detect the *Day*'s operations with $\boldsymbol{n}$
    end for
    Sum up the results of each *Day*
    return Calculate D, M
  end function

TABLE I: Detection Results for January 2017

	Detection Ratio	Detected /Total	Misdetection Ratio	Misdetection /Total
Coffee Maker	0.157	346/2200	0.000	0/48
Heater	0.959	2110/2200	0.182	2/11
Humidifier	0.080	176/2200	0.000	0/38
TV A	1.000	2200/2200	1.000	8/8
TV B	1.000	2200/2200	0.000	0/2

subjects were using the devices. The first dataset was obtained in the winter, and the subjects used the heater. The other datasets were obtained from the spring to the summer, and the subjects did not use the heater but used the electronic fans.

1) Results from the Data for 1 Month: First, we used the dataset obtained in January 2017. In this evaluation, we set the parameters by using the data monitored in the first week. When determining optimal parameter values, we set $M^{goal} = 0.10$, $\left\{T_{init}, T_{inc}, T_{inc}^{stable}\right\} = \{0, 10, 120\}$, $N = 10 \times P$, $i^{max} = 1$, $\left\{\alpha_1^{min}, \alpha_1^{dec}, \alpha_1^{max}\right\} = \{600, 600, 43200\}$, $d^{max} = 2$, and $\left\{\left\{n_1^{min}, n_2^{min}\right\}, \left\{n_1^{inc}, n_2^{inc}\right\}, \left\{n_1^{max}, n_2^{max}\right\}\right\} = \{\{0.00, 0.00\}, \{2/P_{num}, 1/P_{num}\}, \{1.00, 1.00\}\}$, where P_{num} is the number of operations used for parameter setting for the device targeted for detection.

After setting parameters, we evaluate our method by LOOCV using the data monitored in the last three weeks. In this evaluation, we also use the data monitored in the first week as training data to learn the legitimate behaviors.

Table I shows the results of our evaluation in this case. Over 95% of anomalous operations of the heater were detected. The high detection rate is achieved by using the learned event sequences. Our method learned three patterns of event sequences related to the heater; the patterns that a user operates a humidifier before operating a heater, a user operates a heater before operating a coffee maker, and the user01 leaves the room after operating a heater on the afternoon. Even if an anomalous operation command arrives, we detect anomaly unless the operation matches the above-learned behaviors.

The number of misdetection of legitimate operations of the heater is two, which is slightly larger than the predefined target ratio M^{goal}. The misdetected operations are related to the event sequence in which a user operates the heater before operating the humidifier. Such event sequences occurred only a few times in one month and our method did not learn such a sequence as legitimate.

TABLE II: Detection Results for Apr., June, and Aug. 2017.

	Detection Ratio	Detected /Total	Misdetection Ratio	Misdetection /Total
Coffee Maker	0.611	3908/6400	0.058	3/52
Electric Fan A	0.998	6384/6400	0.000	0/9
Electric Fan B	0.999	6399/6400	0.000	0/6
TV A	0.996	6377/6400	0.171	7/41
TV B	1.000	6400/6400	0.400	4/10
TV C	0.999	6397/6400	0.000	0/10
TV D	0.999	6398/6400	0.111	1/9

TABLE III: Detection Results for May, July, and Sept. 2017.

	Detection Ratio	Detected /Total	Misdetection Ratio	Misdetection /Total
Coffee Maker	0.057	368/6500	0.000	0/89
Electric Fan A	0.991	6439/6500	0.074	2/27
Electric Fan B	0.986	6409/6500	0.302	13/43
TV A	0.998	6485/6500	0.231	3/13
TV B	0.999	6497/6500	0.077	1/13
TV C	0.999	6496/6500	0.000	0/3

Table I shows that the detection ratios for operations of the coffee maker or the humidifier are small. This is caused by *single operation*: for most of the operations of the humidifier or the coffee maker, previous or subsequent events were not observed. In addition, they were operated at various times of day. We set parameters so that such single operations at various times of day cannot be detected as anomalies. As a result, most of such operations are regarded as legitimate.

Table I also indicates that all anomalous operations of TVs are detected, but the misdetection ratio for TV A is quite high. This is caused by the low number of operations used to learn legitimate behavior. Furthermore, the monitored event sequences related to TV A vary quite a lot. As a result, an event sequence cannot be learned as legitimate by its similarity to other event sequences. The misdetection ratio of TV B is small because many event sequences are used for detection.

2) Results for Data from 3 Months: We also evaluated our method by using datasets for two sets of three months: April, June, and August 2017 and May, July, and September 2017. We used data monitored in the first week of each month to set parameter values and to learn legitimate behaviors, and then performed LOOCV using the remaining data in a similar way and with same parameters as for the 1-month data.

Tables II and III show the results of the evaluation in these cases. Our method detected more than 98% of all anomalous operations, except for those of the coffee maker. This is because our method successfully learns legitimate event sequences that are repeated many times, such as the sequence that electronic fans are often turned on immediately after entering the room.

Tables II and III show that the misdetection ratios for operations of the TVs are smaller than for the dataset for one month. This is because we used more event sequences to learn legitimate behavior. As a result, when a new legitimate operation arrives, the operation can be regarded as legitimate, because similar event sequences have been learned. However, even when we use more data, we cannot detect the anomalous

operations of the coffee maker accurately. In addition, the misdetection ratios for TV B in Table II and for Electronic Fan B and TV A in Table III are also large. This is caused by their single operation. Therefore, to improve the detection of anomalous operations and reduce the misdetection of the legitimate operations, we need a method to accurately identify such single operations as legitimate, which is one of our future research topics.

3) Discussion: Our method could detect around 95–100% of anomalous operations if the events related to legitimate operations can be monitored. However, our method also produced several misdetections. These are either single operations, for which related events are not observed, or rare operations whose corresponding event sequences are rare.

Our method achieves accurate detection of anomalous operations by comparing event sequences. However, we cannot utilize a sequence to check a single operation and the condition of the operation is the only information we have to determine whether such an operation is legitimate.

One approach to improving the accuracy of identification for single operations is to deploy sensors that can monitor events related to these operations. For example, single operations of the humidifier occurred after the water tank became empty. If we can monitor the water tank of the humidifier, we can obtain events related to its operation. Consequently, such operations of the humidifier are no longer single operation. Another approach to improving the accuracy of identification of legitimate single operations is to use more information to define the conditions. In our evaluation, we used only the time of day to define the condition. However, if we define a condition in such a way as to distinguish conditions where legitimate operations are performed from other conditions, we can identify legitimate single operations accurately. Methods to identify the legitimate single operations are included in our future research topics.

The mitigation of misdetection of rare legitimate operations is another challenge, as we cannot obtain a sufficient amount of training data to accurately identify such rare operations in each home. One approach to obtaining more training data is to use data monitored at other homes. However, several issues to be solved before this could be done. One issue is the difference between homes; information about users whose behaviors when using the devices are different may be useless. Thus, we need a method to obtain data from other homes that have users with similar behaviors. Privacy is another issue. That is, we need a method that uses the data from other homes without exchanging private information. This is also one of our future research topics.

V. Conclusion and Future Work

In this paper, we proposed a method to detect the anomalous operation of home IoT devices. This method learns sequences of user behaviors for each condition that is defined by time of day, sensed temperatures, humidity and so on. Then, when an operation command arrives, the method compares the current sequence with the learned sequences for the condition corresponding to the current condition. If they do not match, our method classifies the operation as an anomaly.

We constructed a network of home IoT devices in our laboratory and let four subjects use the devices. We recorded the times at which the devices were operated along with sensor data. Using this data, we evaluated the detection and misdetection ratios for our method. The results demonstrate that our method detects around 95–100% of anomalous operations if events related to legitimate operations are observed. However, our method cannot accurately identify single operations for which related events are not observed. Rare legitimate operations are also difficult to identify, which results in misdetection. The development of methods to accurately identify rare or single operations is something we will address in future research.

Acknowledgment

This research was supported by Mitsubishi Electric Cybersecurity Research Alliance Laboratories.

References

[1] "Google Home." [Online]. Available: https://home.google.com
[2] "Amazon Echo." [Online]. Available: https://www.amazon.com/echo
[3] I. Lee and K. Lee, "The Internet of Things (IoT): Applications, investments, and challenges for enterprises," *Business Horizons*, vol. 58, no. 4, pp. 431–440, 2015.
[4] M. U. Farooq *et al.*, "A critical analysis on the security concerns of internet of things (IoT)," *International Journal of Computer Applications*, vol. 111, no. 7, 2015.
[5] B. L. R. Stojkoska and K. V. Trivodaliev, "A review of Internet of Things for smart home: Challenges and solutions," *Journal of Cleaner Production*, vol. 140, pp. 1454–1464, 2017.
[6] M. Capellupo *et al.*, "Security and Attack Vector Analysis of IoT Devices," in *International Conference on Security, Privacy and Anonymity in Computation, Communication and Storage*. Springer, 2017, pp. 593–606.
[7] M. Antonakakis *et al.*, "Understanding the Mirai Botnet," in *Proceedings of the 26th USENIX Security Symposium*, Aug. 2017, pp. 1093–1110.
[8] "120,000 IoT cameras vulnerable to new Persirai botnet say researchers," May 2017. [Online]. Available: https://www.zdnet.com/article/120000-iot-cameras-vulnerable-to-new-persirai-botnet-say-researchers/
[9] V. Martin, Q. Cao, and T. Benson, "Fending off IoT-hunting attacks at home networks," in *Proceedings of the 2nd Workshop on Cloud-Assisted Networking*. ACM, 2017, pp. 67–72.
[10] K. Xu, F. Wang, and X. Jia, "Secure the Internet, one home at a time," *Security and Communication Networks*, vol. 9, no. 16, pp. 3821–3832, Jul. 2016.
[11] S. Shirali-Shahreza and Y. Ganjali, "Protecting Home User Devices with an SDN-Based Firewall," *IEEE Transactions on Consumer Electronics*, vol. 64, no. 1, pp. 92–100, Feb. 2018.
[12] K. Xu *et al.*, "Object-oriented big data security analytics: A case study on home network traffic," in *International Conference on Wireless Algorithms, Systems, and Applications*. Springer, 2014, pp. 313–323.
[13] K. Xu *et al.*, "Characterizing Home Network Traffic: An Inside View," *Personal Ubiquitous Comput.*, vol. 18, no. 4, pp. 967–975, Apr. 2014.
[14] Y. M. P. Pa *et al.*, "IoTPOT: A Novel Honeypot for Revealing Current IoT Threats," *Journal of Information Processing*, vol. 24, no. 3, pp. 522–533, May 2016.
[15] M. Lyu *et al.*, "Quantifying the Reflective DDoS Attack Capability of Household IoT Devices," in *Proceedings of the 10th ACM Conference on Security and Privacy in Wireless and Mobile Networks*, ser. WiSec '17. ACM, Jul. 2017, pp. 46–51.
[16] N. Komninos, E. Philippou, and A. Pitsillides, "Survey in Smart Grid and Smart Home Security: Issues, Challenges and Countermeasures," *IEEE Communications Surveys Tutorials*, vol. 16, no. 4, pp. 1933–1954, Apr. 2014.
[17] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Feb. 2006.