Anomalous Operation Detection for Smart Home based on In-home Behaviors of Users

Submitted to
Graduate School of Information Science and Technology
Osaka University

January 2022

Masaaki YAMAUCHI

List of publication

Journal papers

1. Masaaki Yamauchi, Yuichi Ohsita, Masayuki Murata, Kensuke Ueda, and Yoshiaki Kato, "Anomaly Detection in Smart Home Operation from User Behaviors and Home Conditions," *IEEE Transactions on Consumer Electronics*, vol. 66, no. 2, pp. 183–192, May 2020.

2. Masaaki Yamauchi, Yuichi Ohsita, and Masayuki Murata, "Platform Utilizing Similar Users' Data to Detect Anomalous Operation of Home IoT without Sharing Private Information," *IEEE Access*, vol. 9, pp. 130615–130626, September 2021.

Refereed Conference Papers

1. Masaaki Yamauchi, Yuichi Ohsita, Masayuki Murata, Kensuke Ueda, and Yoshiaki Kato, "Anomaly Detection for Smart Home Based on User Behavior," in *Proceedings of 37th IEEE International Conference on Consumer Electronics 2019 (ICCE 2019)*, pp. 1–10, January 2019.

2. Masaaki Yamauchi, Yuichi Ohsita, and Masayuki Murata, "Platform Utilizing Others' Behavior Data to Detect Anomalous Operation Hiding Private Information," in *Proceedings of 7th IEEE International Conference on Consumer Electronics - Taiwan*, pp. 1–2, September 2020.

3. Masaaki Yamauchi, Masahiro Tanaka, Yuichi Ohsita, Masayuki Murata, Kensuke Ueda, and Yoshiaki Kato, "Modeling Home IoT Traffic using Users' in-Home Activities for Detection of

Anomalous Operations," in *Proceedings of 32nd International Teletraffic Congress (ITC32) - PhD workshop*, pp. 1–2, September 2020.

Non-Refereed Technical Papers

1. Masaaki Yamauchi, Yuichi Ohsita, Masayuki Murata, Kensuke Ueda, and Yoshiaki Kato, "Anomaly Detection for Smart Home IoT based on Users' Behavior," *Technical Reports of IEICE (IN2017-58)*, vol. 117, no. 353, pp. 73–78, December 2017 (in Japanese).

2. Masaaki Yamauchi, Yuichi Ohsita, Masayuki Murata, Kensuke Ueda, and Yoshiaki Kato,"[Invited Talk] Anomaly Detection for Smart Home Based on User Behavior -ICCE2019 Report-," *Technical Reports of ITE (ME2019-46)*, vol. 43, no. 5, pp. 249–254, February 2019 (in Japanese).

3. Masaaki Yamauchi, Yuichi Ohsita, and Masayuki Murata, "Framework to Utilize Others' Behavior without Sharing Privacy Information," *Technical Reports of IEICE (IN2019-114)*, vol. 119, no. 461, pp. 213–218, March 2020 (in Japanese).

Preface

Anomalous operation attack that an attacker operates home Internet of Things (IoT) devices by sending packets via an intruded IoT device or a smartphone is an important problem. Because the home IoT devices are physically close to users, this attack may make users unsafe and could even harm users physically. For example, an attacker operates to turn on and off the room light, change the temperature of an air conditioner, or unlock a smart home lock. Furthermore, simultaneous attacks on high-power IoT devices can suddenly increase energy demands, which could lead to major power outages. Therefore, the detection and prevention of packets of the attack are paramount importance.

Unfortunately, existing intrusion detection systems are difficult to detect packets of the anomalous operation attack. The existing intrusion detection systems assume that legitimate and anomalous traffic patterns are notably different. However, 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. It is necessary to grasp whether the users intended to perform the operation of the home IoT devices or not.

Therefore, we first propose 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 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. However, this method modeled users' home states based on the time of day; hence, attackers can exploit the system to maximize attack opportunities.

Thus, we second propose a behavior-modeling method that combines home state estimation and event sequences of IoT devices within the home to enable a detailed understanding of long- and short-term user behavior. The state estimation is that estimating the home state using not only the time of day but also the observable values of home IoT sensors and devices. We compared the proposed model to the first method using data collected from real homes. Compared with the estimation-based method, the proposed method achieved a 15.4% higher detection ratio with fewer than 10% misdetections. Compared with the sequence-based method, the proposed method achieved a 46.0% higher detection ratio with fewer than 10% misdetections.

Learning the behavior of users sufficiently may require a large amount of data, but the amount of data collected in each home is limited. When the data is not enough, the detection of the anomalous operation would be difficult. The data shortage will also occur in the case that a user introduces a new home IoT device.

However, anomalous operations still need to be detected regardless of whether the data amount is sufficient or not. Thus, we consider the cooperation between trained models that learned behaviors of users in the homes. However, an attacker who tries anomalous operations may participate in the cooperation; and gets hints about the timing that the anomalous operations tend to be achieved by watching the cooperation traffic that includes information of behaviors in the past. Therefore, the cooperation should be taken with hiding personal identification. Furthermore, to achieve accurate detection, the trained models that learned the same behaviors of users have to cooperate. Therefore, we need a method that the users having similar lifestyles cooperate without identification of users.

We also propose a framework to utilize data of similar users without sharing private information. We introduce an agent that learns behaviors of users to detect anomalous operations in each home and cooperates with other agents. In this framework, an agent requiring cooperation with other

agents sends a question to the other agents, attaching identifiers of past questions that are similar to the behaviors learned. The receivers decide whether the question is from a similar agent by using the attached information. If the question is from a similar agent, the agent answers the question. We evaluate our framework by using behavior datasets collected from real homes. We simulate two cases: (1) sequences of operations are monitored, and (2) home IoT devices are used alone but sequences cannot be used for detection. The results show that our framework has a 50.5% higher detection ratio for case (1) when using the behavioral data of each user. For case (2), our framework has a 13.4% higher detection ratio when using all the behavioral data of users. However, if an attacker participates in this framework and send fake questions and answers, there are risks that users receive false data via the framework.

Therefore, we propose a countermeasure to suppress the impact of such negative impact while maintaining the advantage of the framework. In this countermeasure, we check the message is sent from the similar agent by checking the attached IDs to the message. If the agent who received the message answered “Yes” to the same questions of the attached IDs, the agent accepts the message. By doing so, the agent avoids accepting the messages that sent from attackers because it is difficult for attackers to attach the IDs that the agent answered “Yes” in the past. We discuss the effects of this countermeasure with numerical examples.

Acknowledgments

This thesis could not have been accomplished without the assistance of many people in various ways, and I would like to acknowledge all of them.

First of all, I express my deepest gratitude to my supervisor, Professor Masayuki Murata, for his generous guidance and insightful comments throughout my Ph.D. His creative suggestions and patient encouragement have been essential for my research activity.

I am heartily grateful to the members of my thesis committee, Professor Takashi Watanabe, Professor Toru Hasegawa, and Professor Hirozumi Yamaguchi of Graduate School of Information Science and Technology, Osaka University, and Professor Morito Matsuoka of Cyber Media Center, Osaka University, for their multilateral reviews and perceptive comments.

Furthermore, I would like to show my greatest appreciation to Associate Professor Yuichi Ohsita of Osaka University. He devoted a great deal of time to me and gave me a lot of advice about my research. Without his continuous support, my work would not be accomplished.

Moreover, I would like to show my appreciation to Associate Professor Shin'ichi Arakawa, Assistant Professor Daichi Kominami, Assistant Professor Naomi Kuze, and Assistant Professor Tatsuya Otoshi of Osaka University for their appreciated comments and support. Their kindnesses on my behalf were invaluable, and I am forever in debt.

I am grateful to Dr. Yoshiaki Kato, Dr. Akira Ishihara, Dr. Takeshi Yoneda, Dr. Shotaro Miwa, Dr. Isao Otsuka, Mr. Tsutomu Yoshikawa, Mr. Masao Noguchi, and Mr. Kensuke Ueda of Mitsubishi Electric Corporation for providing me with many valuable comments. The opportunity of discussion at the Mitsubishi Electric Cybersecurity Research Alliance Laboratories improved my explanation capability.

I am also grateful to Professor Toru Fujiwara, Associate Professor Naoto Yanai of Osaka University, and Assistant Professor Yuya Tarutani of Okayama University, for their advice on my research and experimental environment. Their advice from different perspective points of view was always informative and helpful.

I would like to thank the subjects who participated in the experiment. Also, I sincerely appreciate a lot of support of Mr. Masahiro Tanaka of Osaka University for deploying and operating the experimental environment.

I thank all the members of the Advanced Network Architecture Research Laboratory at the Graduate School of Information Science and Technology, Osaka University, for their inciting discussions and fellowship.

I also thank my parents and my sister for their constant encouragement for my studies.

Finally, I cannot conclude my acknowledgments without expressing my thanks to my fiance, Ms. Ari Koya. Thank you for your invaluable and ongoing supports throughout my Ph.D. studies and for promising me unwavering support throughout my life.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

The Internet of Things (IoT) technology is spread all over the world and 12.3 billion IoT devices are connecting to the Internet in 2021 [1]. In particular, as multiple home appliances connect to the home network, smart homes that can be managed via the network are constructed. Users can operate the home IoT devices and collect information from the IoT sensors via smartphones, AI speakers, and smartwatches. Because of the convenience of these applications, multiple devices will connect to the network and the home IoT devices continue to improve our life.

Along with the growth of home IoT, the security risk of cyberattacks targeting home IoT increases [2–4]. In recent years, a major type of attack is to be intruded on by exploiting the inherent vulnerabilities of devices. The intruded home IoT devices are abused as step servers for the Distributed Denial of Service (DDoS) attack [5, 6]. One of the countermeasures of the intrusion, to update the application and remove the vulnerabilities. However, when the number of devices is large, managing the software of all appliances is costly [7, 8]. Thus, against such intrusion and abuse for the DDoS attack, comprehensive anomaly detection systems are proposed. For example, the detection systems detect the wrong packets by analyzing the behavior of the attacker [9–11] and by comparing the behavior to the usual ones displayed by the home occupants [12, 13].

These attacks are similar to that of personal computers and smartphones, however, due to their characteristics, there are attacks specific to the IoT devices. One of such attacks is the anomalous operation attack that an attacker operates the home IoT device by sending packets via an intruded IoT device or a smartphone. Because the home IoT devices are physically close to users, this attack may make users unsafe and could even harm users physically [14]. For example, an attacker operates to turn on and off the room light, change the temperature of an air conditioner, or unlock a smart home lock. CIA verified that it is possible to hijack a television via the Internet [15]. Furthermore, simultaneous attacks on high-power IoT devices can suddenly increase energy demands, which could lead to major power outages [16]. Therefore, the detection and prevention of packets of the attack are of paramount importance.

Unfortunately, the intrusion detection systems are difficult to detect packets of the anomalous operation attack. The existing intrusion detection systems assume that legitimate and anomalous traffic patterns are notably different. However, 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. It is necessary to grasp whether the users intended to operate the home IoT devices or not.

To grasp the users' intention to detect the anomalous operation, we propose a detection method focusing on user behavior. We construct the model to learn the user behaviors. When a command arrives at a home IoT device, the method verifies whether it matches with the legitimate behaviors. If the command does not match with the learned behavior, it is classified as an anomaly and dropped. Thus, we have to construct the model to learn the user behaviors.

Learning the behavior of users sufficiently may require a large amount of data, but the amount of data collected in each home is limited. When the data is not enough, the detection of the anomalous operation would be difficult. The data shortage will also occur in the case that a user introduces a new home IoT device.

However, anomalous operations still need to be detected regardless of whether the data amount is sufficient or not. Thus, we consider the cooperation between trained models that learned behaviors of users in the homes. However, an attacker who tries anomalous operations may participate in the cooperation; and gets hints about the timing that the anomalous operations tend to be achieved by watching the cooperation traffic that includes information of behaviors in the past. Therefore, the cooperation should be taken with hiding personal identification. Furthermore, to achieve accurate detection, the trained models that learned the same behaviors of users have to cooperate. Therefore, we need a method that the users having similar lifestyles cooperate without identification of users.

In this thesis, we also propose a framework to utilize data of similar users without sharing private information. We introduce an agent that learns the behaviors of users to detect anomalous operations in each home and cooperates with other agents. In this framework, an agent requiring cooperation with other agents sends a question to the other agents, attaching identifiers of past questions that are similar to the behaviors learned. The receivers decide whether the question is from a similar agent by using the attached information. If the question is from a similar agent, the agent answers the question. In addition, an attacker participating in the framework may send fake information to make anomalous operations easier to succeed. We also introduced a countermeasure for the framework to suppress the negative impact of such fake information. The countermeasure is that an agent requires solving a quiz that only agents of users with similar behaviors can understand to avoid accepting the fake information.

1.2 Anomaly detection

In this section, we explain the positioning of this thesis in anomaly detection. We describe the classification of anomaly detection and the difference from related work based on the three guiding principles.

In this thesis, a home gateway learns the behaviors of users based on the observable information via the packets. The packets include the time of operations of IoT devices and the entering or leaving home; the sensed values of IoT sensors. Then, the home gateway detects the operations out of the learned behavior model as anomalous.

Here, it should be noted that the anomalous operation includes malicious operations and unexpected operations. The former are operations performed by attackers with malicious intent; the latter are unexpected operations out for users. In this thesis, the target of the detection is anomalous operations; our methods cannot distinguish the malicious and unexpected ones. However, almost anomalous operations are malicious operations. This is because there is a small number of unexpected operations, such as an operation from a guest of the home and a misoperation by pets. In addition, a user can distinguish them by receiving a detection notice via smartphones and checking the circumstance of the targeted device physically. When the detected operation is legitimate, the user can temporarily allow the operation to the home IoT devices. In this thesis, we unify the detection target as an anomalous operation.

Hence, we list three guiding principles for our anomaly detection below.

1. We have to detect home IoT devices' operations that are deviations from the usual behaviors of users living in the home.

2. The home gateway detects anomalous operations utilizing the information that can be collected from the packets from the home network; it does not use surveillance cameras and many motion sensors that can grasp the physical movement of the users.

3. The anomaly detection method must be applicable to the smart home that multiple users live.

The first guiding principle is to detect the anomalous operations that are different from the usual behaviors of users. Even if the home IoT devices are not affected by malware, we can detect the anomalous operation traffic sent from outside the home or via intruded smartphones and AI speakers. The second principle is that the home gateway learns behaviors and detects anomalous operation without observing physical movement using surveillance cameras or motion sensors. The home gateway can obtain information from the traffic of home IoT devices, sensors, and smartphones; can drop the packets of anomalous operations. The reasons not to use the surveillance cameras and motion sensors are that the users can operate home IoT in any room and users want to avoid excessive monitoring. The third principle is to consider the multiple users living in the smart home because multiple users live in a home. We cannot grasp who operated the IoT devices because users use IoT devices via a shared tablet or an AI speaker at home.

Anomaly detection can be classified into outlier detection, changing point detection, and abnormal behavior detection based on the detection target and the learning model. The outlier detection constructs an independent model as the probability model for multidimensional vectors and detects outlier data from the model [17]. For example, there are detection methods for abnormal communications between IoT nodes based on the usual communication packets of the service where IoT nods cooperate [18, 19] and for abnormal traffic by comparing the legitimate packet payload and statistics to abnormal ones [20, 21]. They detect anomalous communications of intruded IoT devices by comparing them to usual traffic. However, our thesis aims to detect anomalous operations that are unmatched with the usual behaviors of users at the first guideline. In particular, a packet of an anomalous operation via an intruded AI speaker is the same as a legitimate packet including the source IP address. Therefore, we cannot apply these methods to our purpose.

The changing point detection finds the sudden changes that appear in a time series, assuming a time series model as the probability model for multidimensional series data [17]. For instance, there is a changing detection method for the error of the hijacked IoT device based on the time series of electricity consumed [22]. However, this method does not match the first guideline. Specifically, even if the anomalous operation is performed on a home IoT device, it works the same as the legitimate. Thus, the method cannot detect anomalous operations.

The abnormal behavior detection generates a behavioral model as a probabilistic model for a series of behavioral data as a unit and detects abnormal sessions relative to the model [17]. In this thesis, we train a behavioral model by a series of users' actions and detect the operations included in the abnormal behaviors that are deviated from the trained model. Therefore, our detection methods are the abnormal behavior detection. As an example of the abnormal behavior detection, there is a detection method of a user's abnormalities [23]. This method aims to find anomalies of an elderly person, such as a user being down for a long time. It learns the time length and the time of day of actions of a user as a behavior model. The actions are analyzed by the data of motion sensors, surveillance cameras, and IoT devices installed in the home. Then, it detects the cases that the user does not take a particular behavior and something wrong during the behavior. However, this method does not coincide with the second guideline, not to use surveillance cameras and motion sensors, and the third one, adaptable to multiple users at home. Furthermore, this method detects anomalies of the user itself, but, our method detects anomalies of the operations of home IoT devices. It causes a difference in the timing to detect the target. Therefore, this method is difficult to apply our detection. Another method is to detect abnormal behaviors from a hidden Markov (HMM) model [24]. This method models behaviors of the user based on the information about the operation on consumer electronics and location at home with focusing on the single user. Then, it detects anomalous behaviors that deviated from the trained HMM model. The evaluation scenario of this paper included the detection of anomalous operation. However, this method does not satisfy the second and third guiding principles because it uses motion sensors and focuses on the single user. To confirm that this method cannot apply to our anomaly detection guidelines, we compared our method to the HMM model in Chapter 2.

Therefore, we propose anomalous operation detection methods for smart home IoT devices with satisfying the guiding principles.

1.3 Privacy-preserved cooperation

When an agent does not have a sufficient amount of behavior data, we consider the cooperation between trained models that learned behaviors of users in the homes. However, an attacker who tries anomalous operations may participate in the cooperation; and gets hints about the timing that the anomalous operations tend to be achieved by watching the cooperation traffic that includes information of behaviors in the past. To prevent such problems, in this thesis, we described three requirements for the cooperation of the agents in the homes.

1. The framework must not use any information to identify the individual users. The framework must not assign any identifiers to the users and agents. In addition, the agents must not share the personal information of the users including the historical behavior of the users and their personal information, such as ages, genders, and jobs.

2. The agents must avoid cooperating with agents of users who have different lifestyles, which may cause inaccurate detections of anomalous operations.

3. The framework prevents the negative impact of the fake information sent from the attacker. The anomalous operation would become easier to succeed by receiving the fake information via the framework.

Here, the attacker is a user who tries anomalous operations and can perform the anomalous operation on a home IoT device. This framework prevents the attackers from grasping hints of the anomalous operation.

The first requirement is to avoid identifiers of users and agents. This is because we consider the risk of data leakages. In the case that the agents send the behavior data in the home to outside, we learn the relationship between behaviors that are performed by the same users by adding an identifier of users to the behaviors. If the series of behaviors of users are leaked, attackers can understand the timing that anomalous operation is likely to succeed and we cannot prevent the anomalous operation. Therefore, the agents hide the personal information or identifier and communicate to prevent the risk of leaking hints for the attackers. An approach that satisfies the requirement is to train each model on each dataset and to construct a general model by sharing the learned gradients [25, 26]. However, this general model constructed via the cooperation between agents may not match the lifestyle of each user because each user has different behaviors. Therefore, this approach cannot achieve accurate detection of anomalous operations and satisfy the requirements 2.

The second requirement is to avoid cooperation with agents who have different lifestyles. If an agent uses behavior data of different lifestyles, the agent overlooks the anomalous operations. Thus, agents need to cooperate with the agents that their users have similar lifestyles to prevent false negatives. An approach to analyzing the data of other users by collecting the behavioral datasets of many users [27] satisfies this requirement. This method uses the privacy information of the users, including the in-home activities of the users. However, because this method has to relate the users and behaviors, there is a risk of data leakage [28]. Furthermore, in the circumstance of each agent learning its users' behavior data in its home, some users feel uncomfortable transmitting all their activities at home to the outside. Therefore, this method cannot satisfy the first requirement and cannot apply to our purpose.

Furthermore, as a requirement 3, it is necessary to prevent attackers from sending false information to facilitate the success of anomalous operations. The false information may cause the agent to receive information about the wrong behavior, resulting in undetected or false positives. Therefore, we needed an anonymous cooperation framework with agents of similar users that would not be adversely affected by such false information.

Thus, it was difficult to achieve the requirements with existing cloud-based and decentralized data analysis methods to cover the data shortage. It is also difficult to completely protect the privacy of users and to use the data of users having completely the same lifestyles. Therefore, we propose a framework that the agents of users with similar lifestyles can cooperate to detect anomalous operations as anonymously as possible. In Chapter 4, we first propose an anonymous cooperation framework with agents of similar users that satisfies requirements 1 and 2. In Chapter 5, to achieve the requirements 3, we second propose a countermeasure that prevents the negative impact of the fake information sent by the attackers to the framework.

1.4 Outline of thesis

The overview of this thesis is shown in Figure 1.1.

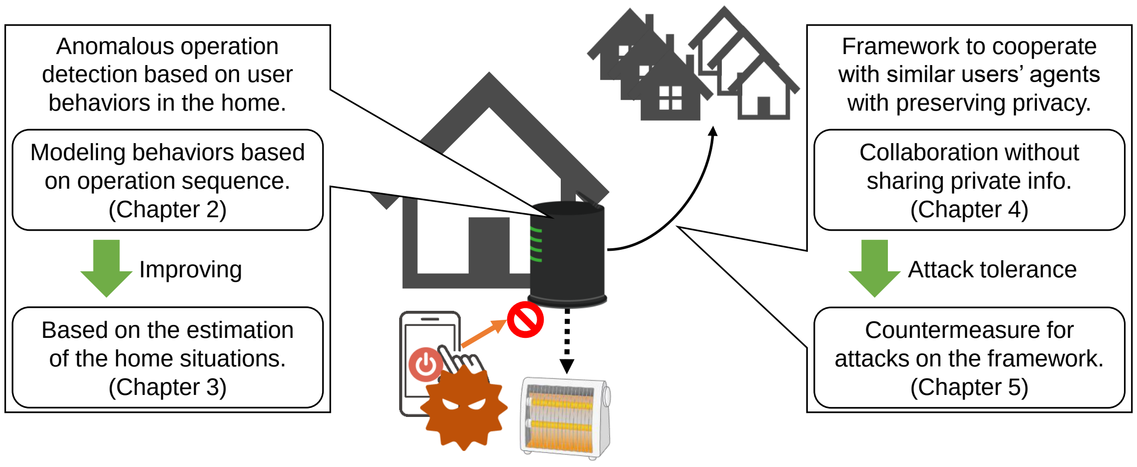

Anomalous operation detection based on user behaviors in the home.
Modeling behaviors based on operation sequence. (Chapter 2)
Improving
Based on the estimation of the home situations. (Chapter 3)
Framework to cooperate with similar users' agents with preserving privacy.
Collaboration without sharing private info. (Chapter 4)
Attack tolerance
Countermeasure for attacks on the framework. (Chapter 5)

Figure 1.1: Overview of this thesis.

Anomaly Detection Method using User Behavior Sequences [29–32]

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 propose 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 thesis, we also discuss the effectiveness of our method by comparing with a method learning users' behavior by Hidden Markov Models.

Improving Anomaly Detection Method by Estimating In-home Situation [33–35]

Internet-of-things (IoT) devices are vulnerable to malicious operations by attackers, which can cause physical and economic harm to users; therefore, we previously proposed a sequence-based method that modeled user behavior as sequences of in-home events and a base home state to detect anomalous operations. However, that method modeled users' home states based on the time of day; hence, attackers could exploit the system to maximize attack opportunities. Thus, we propose a behavior-modeling method that combines home state estimation and event sequences of IoT devices within the home to enable a detailed understanding of long- and short-term user behavior. The state estimation is to estimated the home state using not only the time of day but also the observable values of home IoT sensors and devices. We compared the proposed model to our previous methods using data collected from real homes. Compared with the estimation-based method, the proposed method achieved a 15.4% higher detection ratio with fewer than 10% misdetections. Compared with the sequence-based method, the proposed method achieved a 46.0% higher detection ratio with fewer than 10% misdetections.

Privacy-Preserved Cooperation Framework for Anomaly Detection without using Private Information [36–38]

To mitigate the risk of cyberattacks on home IoT devices, we have proposed a method for detecting anomalous operations by learning the behaviors of users based on the operation sequences of their home IoT devices and home conditions. While this method requires a sufficient amount of training data, achieving accurate detection is still possible by utilizing the data of users with similar lifestyles. However, users are unwilling to share their private information with others. In this study, we propose a platform to utilize data of similar users without sharing private information. We introduce an agent that learns behaviors of users to detect anomalous operations in each home and cooperates with other agents. In this framework, an agent requiring cooperation with other agents sends a question to the other agents, attaching identifiers of past questions that are similar to the behaviors learned. The receivers decide whether the question is from a similar agent by using the attached information. If the question is from a similar agent, the agent answers the question. We evaluate our platform by using behavior datasets collected from real homes. We simulate two cases: (1) sequences of operations are monitored, and (2) home IoT devices are used alone but sequences cannot be used for detection. The results show that our framework has a 50.5% higher detection ratio for case (1) when using the behavioral data of each user. For case (2), our framework has a 13.4% higher detection ratio when using all the behavioral data of users.

Improving Attack Tolerance of Privacy-Preserved Cooperation Framework

Network services can be tailored to preferences of users by analyzing the users' data. To collect the data, we should consider its privacy; collecting data in a cloud includes a risk of data leakage. However, a privacy-preserving method that generates a general model by collecting gradients of each trained model is difficult to reflect personal preferences. Thus, we proposed a privacy-preserved-cooperation (PPC) framework that users can send questions to ask information about data and receive answers from similar users without telling who senders are. In PPC framework, users collaborated with others who sent same answers to the same questions in the past. However, if an attacker participates in PPC framework and send fake questions and answers, there are risks that users receive false data via PPC framework. In this work, we propose a countermeasure, PPC with Countermeasure (PPCwC), to suppress the impact of such negative impact while maintaining the advantage of PPC. In this PPCwC framework, we check the message is sent from the similar agent by checking the attached IDs to the message. If the agent who received the message answered “Yes” to the same questions of the attached IDs, the agent accepts the message. By doing so, the agent avoids to accept the messages that sent from attackers because it is difficult for attackers to attach the IDs that the agent answered “Yes” in the past. We discuss the effects of PPCwC framework with numerical examples.

Chapter 2

Anomaly Detection Method using User Behavior Sequences

2.1 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 21.5 billion devices expected by 2025 [39]. As the number of devices connected to the Internet increases, the risk for these devices to be targeted by cyberattacks also increases [40–43]. In fact, direct attacks and malware targeting of IoT devices [3, 4] have already been reported.

Most of the current attacks targeting IoT devices aim to create botnets [5, 6]. Such attacks are detectable by methods based on analyses of attacker behaviors [9–11] or through usual traffic comparisons [12, 13].

However, as IoT devices are closely included in our routine, attacks may cause immediate and personal harm to users [14]. 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. [44] 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 [31]. 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, 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 chapter, we investigate the effectiveness of our method by comparing it with a method to learn normal activities of a user by using Hidden Markov Models (HMM) [24]. 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 chapter is organized as follows. We describe the related work in section 2.2. Then, We describe the proposed method to detect anomalous operations in section 2.3. After that, we report the evaluation of the method and the corresponding results in section 2.4. Finally, we draw conclusions and discuss directions of future work in section 2.5.

2.2 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 [3, 4], which can cause serious problems such as compromising safety and even harming users [14]. Therefore, the methods to detect attacks for the home IoT devices have been proposed [9–13]. 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 [45]. 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]. Martine 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 [9].

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 [24]. However, a smart home may have multiple users. Therefore, in this chapter, 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 paper, 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 [46]. Haider et al. proposed a method to detect zero-day attacks for cloud servers by modeling the users' usage behavior. This paper also defined user behaviors as a sequence of actions. This paper modeled the behavior by using nested-arc hidden semi-Markov model (NAHSMM). This paper demonstrated that the method detects major attack families such as DoS and worms by learning the model by using traffic data [47].

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 [48].

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.

2.3 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 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.

2.3.1 Learning Model

Fig. 2.1 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.

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. 2.1 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

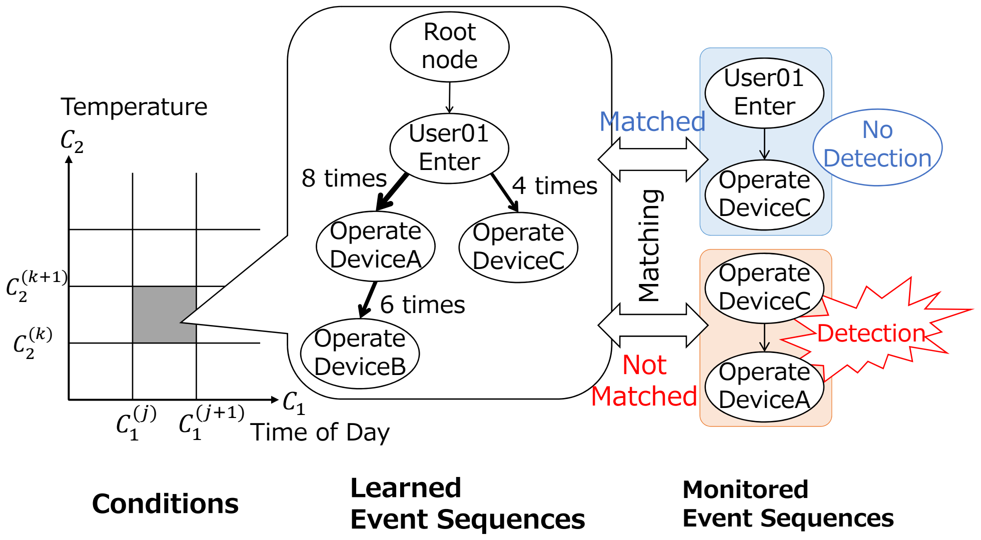


Figure 2.1: Overview of detection model.

colored area of Fig. 2.1.

User Behaviors

In this chapter, 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 chapter, 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. 2.1 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.

2.3.2 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 paragraph explains the detail of each step to learn user behaviors.

Generation of Event Sequences

Algorithm 1 generates event sequences that are subsets of the monitored event sequence $s^{\mathrm{monitored}}$. 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 $s^{\mathrm{monitored}}$. Otherwise ith event of $s^{\mathrm{monitored}}$ is not included in s^{tmp}. After generating one event sequence, v is incremented. By continuing the above steps, we generate all subsets of $s^{\mathrm{monitored}}$.

Figure 2.2 shows an example to generate event sequences when a sequence "A-B-C" is monitored. The first v is set to 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.

Algorithm 1 Generating event sequences from the monitored event sequence.

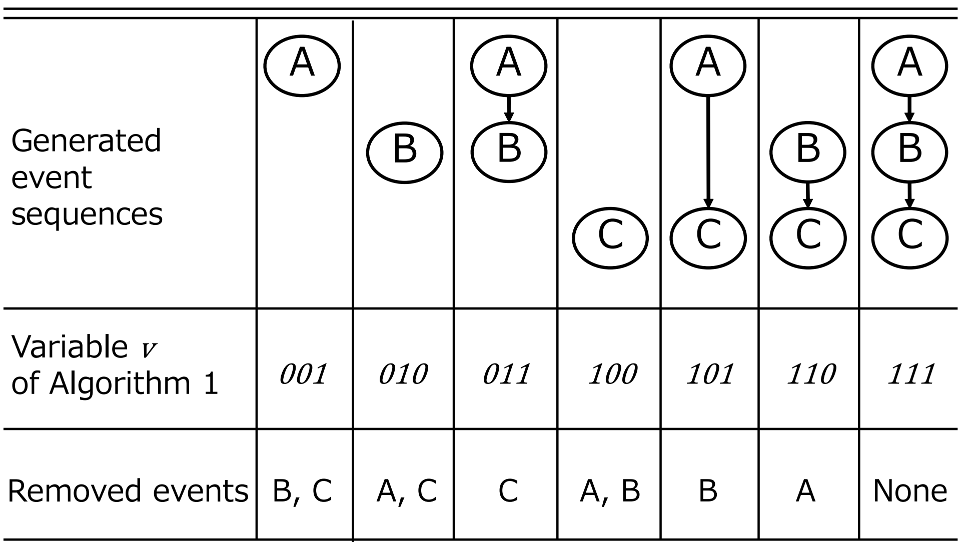
```
Require: $s^{\mathrm{monitored}}$: monitored sequence of events
Ensure: $S$: List of generated sequences of events
  function GenerateEventSequences($s^{\mathrm{monitored}}$)
      $v \Leftarrow \mathrm{UnsignedBinaryVariable}(1)$
      $S \Leftarrow \mathrm{EmptyList}()$
      while $v < \mathrm{UnsignedBinaryVariable}(2^{|s^{\mathrm{monitored}}|})$ do
          $s^{\mathrm{tmp}} \Leftarrow \mathrm{EmptySequence}()$
          for $i = 1 \ldots |s^{\mathrm{monitored}}|$ do
              if $i$th bit of $v$ = 1 then
                  Add $i$th event of $s^{\mathrm{monitored}}$ to $\mathrm{s}^{\mathrm{tmp}}$
              end if
          end for
          Add $\mathrm{s}^{\mathrm{tmp}}$ to $S$
          Increment $v$
      end while
      return $S$
  end function
```

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 $\{c_1^{root}, \ldots, c_{imax}^{root}\}$, 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.

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



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

Figure 2.2: Generated event sequences when events A, B, and C occur in this order within T seconds.

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 $s_{2..|S|}$ 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 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.

2.3.3 Detection

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

Algorithm 2 Updating the tree of each condition using generated event sequences.

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

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.

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 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

Algorithm 3 Comparing a new event observed sequence with learned event sequences

```
Require: pos: node of learned event sequences, $s$: observed event Sequence
Ensure: 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
```

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.

2.4 Evaluation and Results

2.4.1 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. 2.3. We deployed two access

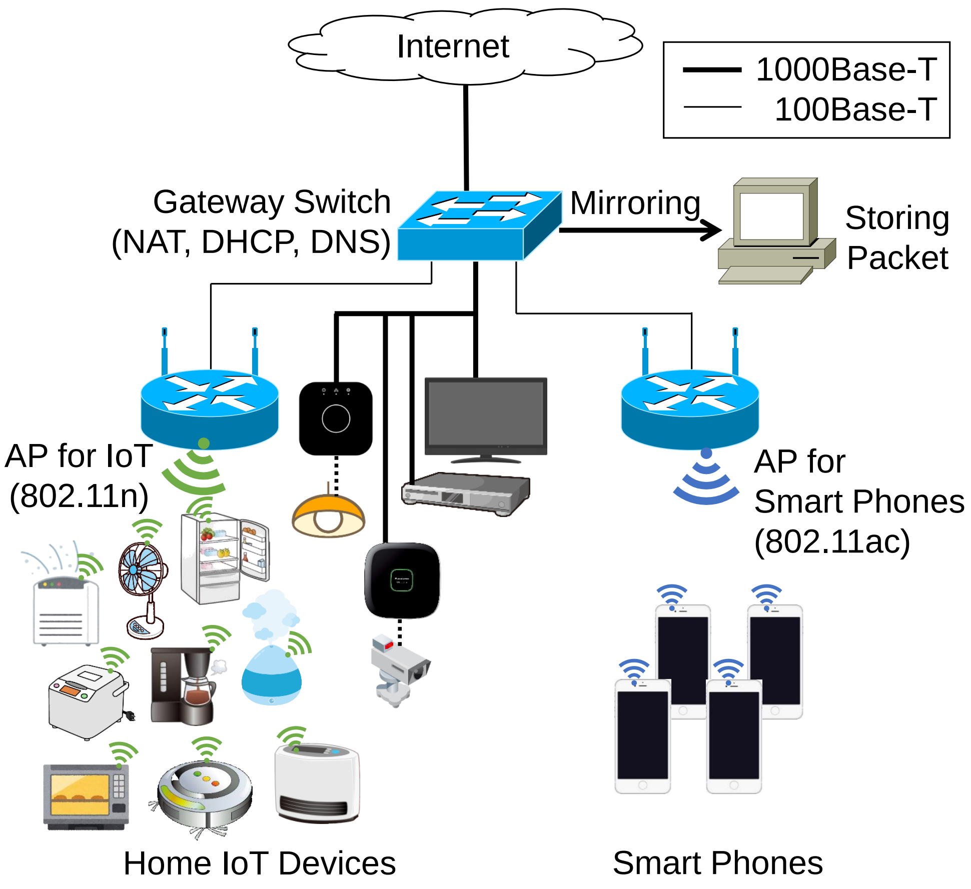


Figure 2.3: Experimental home network environment.

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 2.1. 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

Table 2.1: 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

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.

In this experimental home network environment, four students from our laboratory participated in the chapter. 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 chapter.

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.

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 [49], 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.

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.

2.4.2 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. [24]. 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.

HMM Method [24]

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

Table 2.2: Settings of proposed and its variant methods.

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

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 2.1 and the list of users currently in the room as the observations.

Variant of our methods

Table 2.2 shows the variants of our method used in this section. The details of the variants are as follows.

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 $\mathit{Time} - \alpha_1$ to $\mathit{Time} + \alpha_1$ is below $n_1 \times L_{\mathit{num}}$, where *Time* is the time of day of the tested operation.

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.

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

Table 2.3: 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	$600, 1200, \ldots, 43200$	$0.00, 0.01, \ldots, 1.00$

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. 2.2.

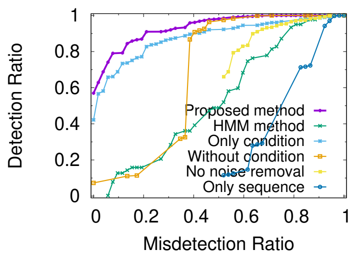
Only sequence This method simply learns user behaviors based on only sequences of events without noise removal method similar to the existing methods that learn user behaviors. By comparing our method with this method, we demonstrate the advantages of our method over the existing methods.

2.4.3 Evaluation Parameters

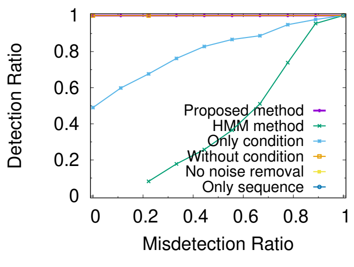
The proposed method has three types of parameters, T, α_1, n_1, and n_d. We set T according to the procedure in our previous work [31]. We adjusted the other parameters and evaluated the method to obtain the misdetection and detection ratios, whose relations were depicted as receiver operating characteristic curves. Table 2.3 lists all the evaluation parameters for this chapter.

2.4.4 Results

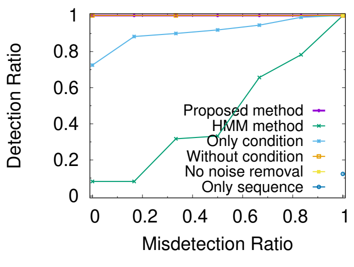
Fig. 2.4 and 2.5 show the receiver operating characteristic curves of the proposed method and five compared methods, where the detection ratio is plotted according to the misdetection ratio. The proposed method detects more than 90% of attacks, whereas misdetection reaches only 10% of the legitimate operations for most devices.



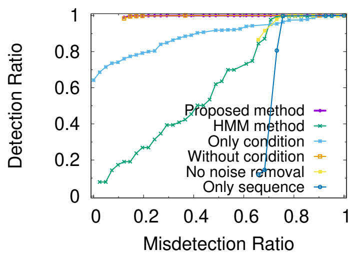
(a) Coffee maker in dataset A



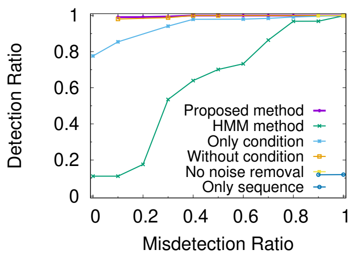
(b) Electric fan A in dataset A



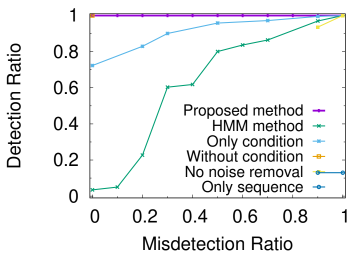
(c) Electric fan B in dataset A



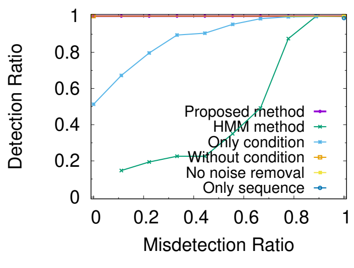
(d) TV A in dataset A



(e) TV B in dataset A



(f) TV C in dataset A

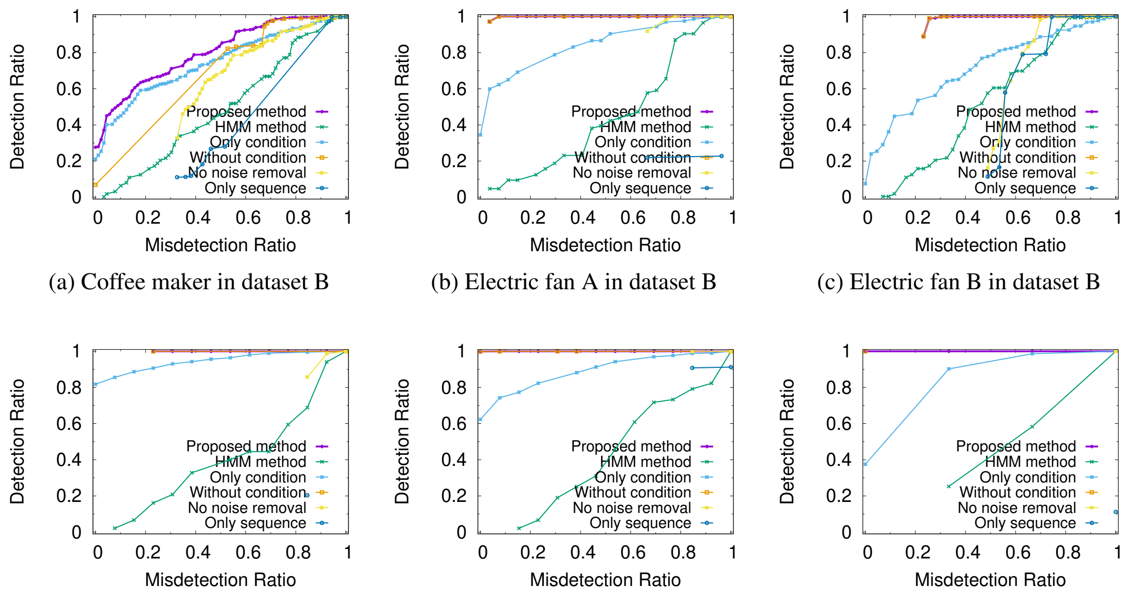


(g) TV D in dataset A

Figure 2.4: Receiver operating characteristic curves of detection using the proposed method and comparison variants in dataset A.

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



(a) Coffee maker in dataset B (b) Electric fan A in dataset B (c) Electric fan B in dataset B

(d) TV A in dataset B (e) TV B in dataset B (f) TV C in dataset B

Figure 2.5: Receiver operating characteristic curves of detection using the proposed method and comparison variants in dataset B.

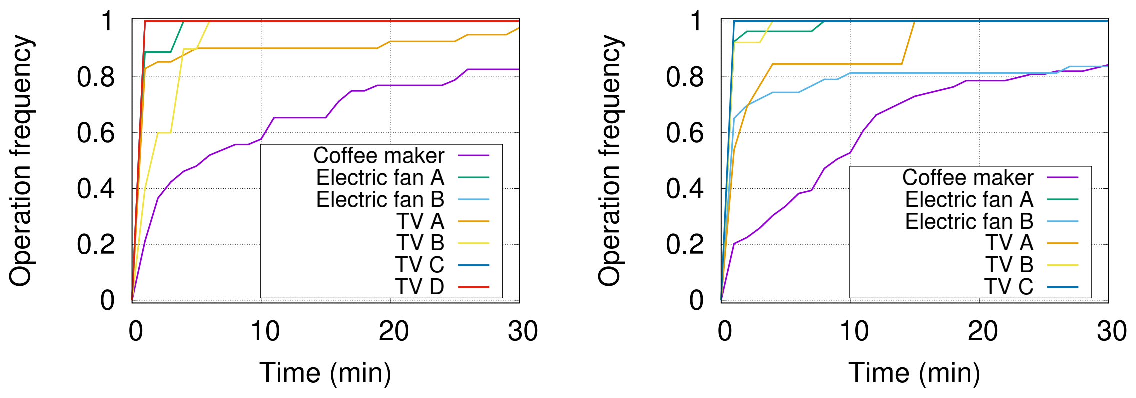
corresponded to user 2 entering the room and operating electric fan B, leading to a misdetection for the legitimate fan operation. In addition, the detection ratio of the HMM method is much lower than that of our method. This is because, the HMM method does not consider that multiple users are in the 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 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. 2.4 and 2.5 show 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. 2.6 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 2.3.2. 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. 2.7 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 2.6 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. 2.6 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



(a) Dataset A (b) Dataset B

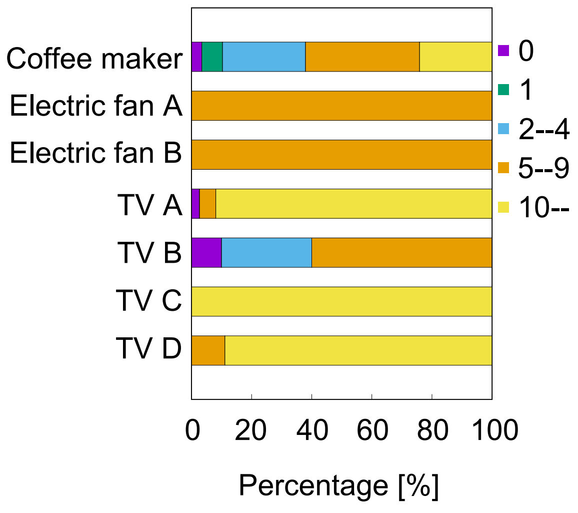
Figure 2.6: Distribution of time between an operation of each device and its nearest event.

define the conditions.

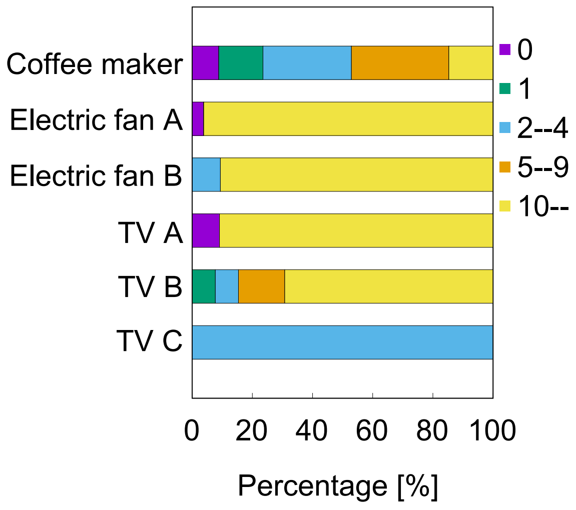
Another kind of devices that cannot be detected accurately includes TVs A and B in dataset A and electric fan B and TV A in dataset B. Our method detects more than 90% of the attacks for these devices but cannot make the misdetection ratio less than 10%. This is caused by rare event sequences. Fig. 2.7 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.

2.4.5 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) Dataset A



(b) Dataset B

Figure 2.7: Frequency of matching behaviors.

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, 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 2.4, 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.

2.5 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.

Chapter 3

Improving Anomaly Detection Method by Estimating In-home Situation

3.1 Introduction

Smart homes with multiple internet-connected home appliances have become widespread as part of the internet of things (IoT). More than 12 billion IoT devices were deployed in 2020, and it is estimated that the number of IoT appliances now surpasses the number of non-IoT versions [50]. Users can connect to their IoT devices (e.g., washing machines, home sensors, and cooking stoves) via smartphones and smartwatches. The growth of this trend is expected to continue indefinitely [50].

However, with this growth, the risk of cyberattacks targeting home IoT devices increases [2]. A major type of cyberattack on home IoT devices is the distributed denial-of-service attack, which affects multiple IoT devices simultaneously based on the devices' inherent vulnerabilities [5, 6]. Fortunately, countermeasures exist [9–11].

Notably, it is very difficult to maintain the boundary security of IoT devices [7] because they employ many different communication protocols and connect to many different platforms. Moreover, proper boundary security would be exceedingly expensive [8]. Therefore, anomaly detection systems that comprehensively monitor a smart home or a smart factory to detect abnormal (out-of-the-ordinary) IoT behaviors (e.g., signals, operating status, and error reporting) [12, 13, 51] are needed. For example, Sivanathan et al. proposed a monitoring system that analyzed legitimate behaviors of IoT devices by classifying their traffic flows [51]. Distributed denial-of-service attacks on smart homes have been detected by comparing suspicious traffic with usual behaviors based on home occupancy [12, 13].

Notably, cyberattacks on IoT devices create significant additional human risks [14]. In particular, attacks that take control of home IoT devices are considered dangerous not only in cyberspace but also in the physical world. For example, simultaneous attacks on high-power IoT devices can suddenly increase energy demands and lead to power outages [16]. As a discrete example, it has also been shown that in-home IoT televisions can be hijacked from the internet [15]; similar attacks have been shown to affect smart phones and smart watches [52].

To address attacks on home IoT devices leading to anomalous operations, we previously proposed a detection method [32] that modeled the behavior of users from sequences of events in their homes to assess normal behaviors. This sequence-based method trained its model by storing event sequences based on the time of day so that deviations from operations could be detected. However, this sequence-based method was too simplistic, and the home state was not studied in detail; hence, it was noted that an attacker could optimize attacks by studying the time-of-day behaviors.

Subsequently, we proposed another anomaly detection method [34] that modeled home states by estimating the sensed values and operating statuses of IoT devices. That estimation-based method calculated the operating probability of the IoT device and assessed anomalies based on a baseline threshold. The estimation-based method achieved a better detection accuracy than a method to detect anomalous operation based on only the time of day information. As the estimation was based on the current home situation, it was difficult for attackers to exploit the system because they could not easily estimate the timing when an attack would be likely to succeed. However, this estimation-based method could not grasp user activities in detail over short periods.

Therefore, in this chapter, we propose a detection method that models user behavior by combining state estimation and behavior sequences of in-home activities performed over short periods. Hence, our sequence-based method can grasp the short-term activities of users in detail, whereas the estimation-based method grasps the long-term transitions of the home state. The proposed method stores the sequences in the estimated home states. Then, the proposed method calculates the occurrence probabilities of sequences, including detection target operations, and it detects anomalous operations when the probability is lower than a threshold value.

We simulated the proposed method and compared the results to those of the previous sequence-based and estimation-based methods using datasets of behaviors and sensor values collected from real homes.

The remainder of this article is organized as follows. We describe anomaly detection methods for operations of home IoT devices in Section 3.2. The proposed method, including the estimation of the in-home situation, storage of behavior sequences, and their combinations, is described in Section 3.3. Then, we report on the evaluation of the proposed method and the corresponding results in Section 3.4. Finally, we conclude the chapter and discuss possible avenues for future research in Section 3.5.

3.2 Related Works

Here, we explain detection methods of anomalous operations that learn user behaviors based on their usage of home IoT devices.

Ramapatruni et al. proposed a method to detect anomalous operations. Their method used hidden Markov modeling (HMM) to learn a single user's normal activities. HMM parameters were then trained with information obtained from IoT sensors. Then, the trained HMM detected anomalous operations when the probability of that operation occurring was lower than a baseline threshold. The accuracy of this method was demonstrated using a dataset collected from a smart-home environment. The authors collected detailed activity information on the user entering and leaving the home and the operations of the consumer electronics therein. Additionally, IoT activities from the living room, bedroom, bathroom, and closet devices were recorded. This method learned the behaviors of a single user in detail. However, the method could not be applied to a home containing multiple users [24]; it was examined in our previous work [32]. It is difficult to deploy this method in real homes because most involve multiple users, which greatly increases the difficulty. In contrast, our proposed method models the situation while focusing on the states of the home instead of the states of the user. Furthermore, the proposed method uses information that can be easily collected from commercially available IoT sensors and home gateways. Therefore, our proposed method can be applied to real home environments.

We previously proposed a method to detect anomalous operations even in cases of multiple users by utilizing their sequence of behaviors [32]. This sequence-based method detected anomalous operations at the home gateway, which was connected to all home IoT devices, sensors, and smartphones. The home gateway collected two types of information: the state information of the operations of devices (e.g., time of day, room temperature, and humidity) and the presence or absence of users in the home based on the statuses of their smartphones. The home gateway subsequently classified the states of the home by constructing a table of sensed values, storing the sequences of operations of IoT devices and data on the entry and exit of users in each cell. Finally, the home gateway judged whether legitimate or anomalous operations occurred by comparing sequences of current operations to the stored sequences of the current state. This sequence-based method handled cases of multiple users by constructing their sequences from the monitored operations. However, the sequence-based method used only the time-of-day information in the table to classify the states. Therefore, an attacker could estimate the optimal attack times based on the time of day. Owing to the large impact of sequence information utilization, the sequence-based method achieved high accuracy. However, the detailed analysis of state learning was deficient.

Hence, we proposed another anomaly detection method that estimated the states of a home based on the sensed values and operating statuses of IoT devices. This estimation-based method calculated the operating probability of each state and detected anomalies when the probability was low. The study compared this estimation-based method to one that used only time-of-day information, confirming that the estimation-based method was more accurate [34]. Therefore, an attacker could not exploit the system based only on time-of-day information. However, the method could not learn the short-term behavior patterns of users. Additionally, some details needed to be corrected, owing to the inadequate observed data.

In this chapter, we propose a more accurate detection method that combines the estimation- and sequence-based methods. The proposed method determines the current state in the home via estimation and the state transitions by learning the behavior patterns of users. Furthermore, we improve the estimation-based method [34] to achieve higher accuracy by fixing details and correcting the observed data.

3.3 Anomaly Detection Method based on In-home Situation and Behavior Sequence

We propose a new model that learns the behaviors of IoT users in a home to detect anomalous device operations as a safeguard against cyberattacks. The model learns sequences of user behaviors alongside corresponding states of the home. When an operation does not match the trained model, the model flags the operation as anomalous.

3.3.1 Models used for detection

The proposed method estimates home states and stores behavior sequences collected over time. First, it defines a timeslot scheme and updates the home status in each slot throughout the day. During training, the model calculates the state transition probability, a, and the operation probability, b, using the labeled home state as the training data. The method then estimates the home state by calculating the state probability of the training data using a and b. Following the calculation, the operation sequences of the home IoT devices are stored according to the estimated home state. High-probability sequences are considered legitimate behaviors. The overview of the learning model is described in Fig. 3.1. After storing the sequence, the proposed method calculates the probability b' of each behavior sequence that occurs in real time.

Next, we describe the components of the proposed model.

State of the home

The proposed method labels and estimates the current home state, $s_{u,d}$, which is combined with the activity states of the users, u, and the usage states of the devices, d, obtained using the sensor values of IoT operations.

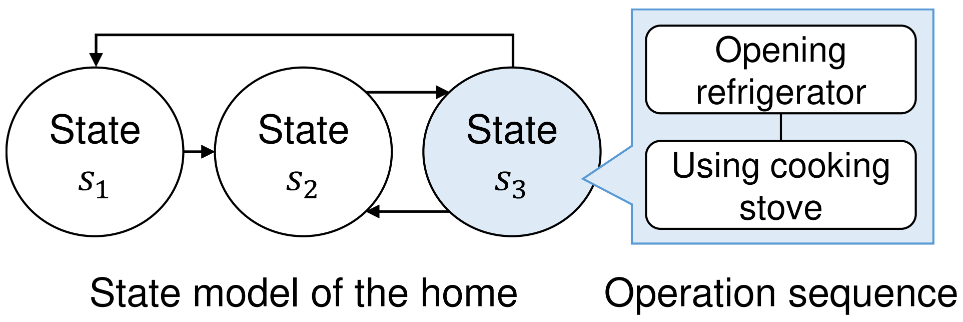


Figure 3.1: Overview of the training model of the proposed method. According to the estimated home state "State s_3", a cooking stove is used after opening a refrigerator.

The activity state of users, u, reflects the situation of users in the home (e.g., all users are away, at least one user is home and active, or all users are sleeping). Variable u is thus defined several ways according to the home environment and the number and attributes of users. For example, "active," "out," "sleeping," etc. can be considered; thus, we can set u as $u \in \{\mathit{active}, \mathit{out}, \mathit{sleep}, \dots\}$.

There are four types of usage states, d, related to how home IoT devices are used. The state prior to use is *before*, the state after use is *after*, the state during use is *use*, and others are *none*. Thus, the device state, d, is defined as $d \in \{\mathit{use}, \mathit{before}, \mathit{after}, \mathit{none}\}$.

Furthermore, the proposed method calculates the state transition probability a to forecast the changes in the home state over time. Because user behaviors differ greatly during day and night, the state transition probability, a, in the home varies depending on the time of day. Therefore, in our model, the state transition probabilities are defined for each timeslot of the day. The transition probability $a_k(i,j)$ from state i to state j in the k-th timeslot of each day is defined by Equation (3.1), where s_k is the state in the k-th timeslot.

$$a_k(i,j) = P(s_k = j | s_{k-1} = i). \tag{3.1}$$

Additionally, the proposed method calculates the operation probability, b, to reflect the activity of the users to the home states as devices are operated. The operation probability b differs for each state i and for each operation x. Therefore, in this model, the operation probabilities are defined for each state. The operation probability $b(i,x)$ of the operation x in state i is defined by the following

Equation (3.2):

$$b(i,x) = P(x|s=i). \tag{3.2}$$

Sequence of events in the home

The event sequences in the home are stored and used for detection. The information is obtained from the home network via IoT operation packets, including information of the connection and disconnection of smartphones.

As with the sequence-based method, an event sequence is defined as a series of events performed within T_{seq} s, where T_{seq} is a parameter determining whether or not events are considered as a sequence. We consider actions A and B to be a series if they satisfy Equation (3.3), where time_A and time_B are the times when actions A and B are performed and $\mathit{Diff}(\mathit{time}_A, \mathit{time}_B)$ is a function that obtains differences in seconds between time_A and time_B.

$$\mathit{Diff}(\mathit{time}_A, \mathit{time}_B) \leq T_{seq}. \tag{3.3}$$

Furthermore, after storing the sequences in the estimated states, the proposed method calculates the behavior sequence probability b' for detection. When the proposed method identifies an event sequence including the operations of the detection target device, the probability of occurrence of sequences is calculated by multiplying the state probabilities by the behavior sequence probability b'. The behavior sequence probability $b'(i,y)$ in state i of the sequences y is defined by Equation (3.4).

$$b'(i,y) = P(y|s'=i), \tag{3.4}$$

where $s'=i$ means that the estimated state is i.

3.3.2 Training the model

The proposed method trains the model using data collected from the home divided into timeslots. We assign the observed values and labels of the home states $s_{u,d}$ to the timeslots. Then, the proposed method calculates the state transition probability, $a_k(i,j)$, according to the labeled states, $s_{u,d}$. The

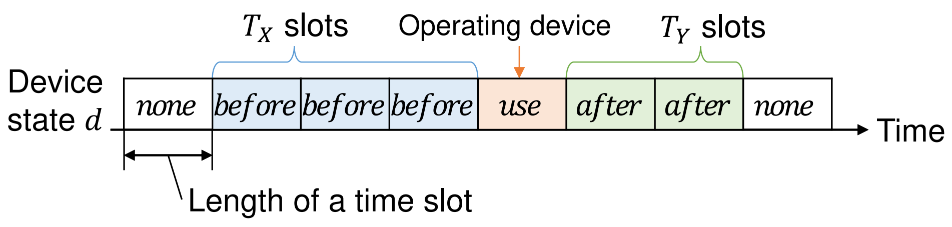


Figure 3.2: Labeling rule of device state d, where T_X is 3 and T_Y is 2.

proposed method also calculates the operation probability $b(i, x)$ that the device is operated in each state. Next, the proposed method calculates the state probability $\alpha(s)$ with a and b. The proposed method generates multiple sequences assuming that the homes have multiple users. Based on the calculated state probability $\alpha(s)$ and the generated sequences, event sequences are stored for each estimated home state. Then, the proposed method calculates the operation probability $b'(i, y)$ in each estimated state. The rest of this subsection explains the details.

Labeling the training dataset

To create the training data, we divided the observed data into multiple parts using timeslots and labeled the states accordingly. The state s is set by combining the activity state of the users u and the usage state of the device d as defined.

The state of the users u is set using IoT sensor data according to predefined rules. Because these rules vary depending on the target device, the type of IoT sensors, and the number of users, we set the labeling rule according to the scenario.

The device state d is determined based on the time the target device is operated. As shown in Fig. 3.2, we define the four states of the target device d as follows. *use* indicates that the device is in use, and *before* indicates that the device will be used within T_X timeslots. Similarly, *after* indicates that the device has been used within T_Y timeslots, while *none* denotes other states. Variables T_X and T_Y are parameters.

Furthermore, to change the state for each device operation, we update them as observed. An example of the learning data is described in Table 3.1.

Table 3.1: Learning data samples include the observed information, the labeled states, and variable characters. The labels change even in the same timeslot according to the operations of the devices. In this table, we set the length of the timeslot to 1 min. As a sample rule for the state of users u, we set *sleep* when the CO2 value is higher than 35 and the noise value is lower than $1,500$. As a sample rule of the device state d, we set T_X as 1 and T_Y as 1.

	Date information		Observed information			Labeled states		
ID	Date	k-th timeslot of the day / t-th timeslot of the data	CO2	Noise	Operation	Users u	Device d	Home $s_{u,d}$
4350	2020/1/3 23:56:00	1438/4318	34	1520	—	*active*	*none*	$s_{active,none}$
4351	2020/1/3 23:57:00	1439/4319	34	1520	—	*active*	*before*	$s_{active,before}$
4352	2020/1/3 23:58:00	1440/4320	34	1520	—	*active*	*before*	$s_{active,before}$
4353	2020/1/3 23:58:20	1440/4320	34	1520	Refrigerator _Open	*active*	*before*	$s_{active,before}$
4354	2020/1/3 23:58:35	1440/4320	34	1520	Cooking oven _On	*active*	*using*	$s_{active,cooking}$
4355	2020/1/3 23:59:00	1/4321	34	1520	—	*active*	*after*	$s_{active,after}$
4356	2020/1/4 00:00:00	2/4322	41	1480	—	*sleep*	*none*	$s_{sleep,none}$

Calculating state transition probability and the operation probability

Based on the labeled home states for changing timeslots, we calculate state transition probabilities $a_k(i,j)$ from state i to state j at the k-th timeslot during the day. This is used to calculate the probability $\alpha_t(s)$ that the home state s is in timeslot t. Although the time of the state transition varies daily, similar state transitions occur in similar timeslots. Therefore, $a_k(i,j)$ is calculated by Equation (3.5) by considering the data from the $k-T_Z$-th timeslot to the $k+T_Z$-th timeslot of each day.

$$a_k(i,j) = \begin{cases} \dfrac{\displaystyle\sum_{m=k-T_Z}^{k+T_Z} N_{m+1,j}}{\displaystyle\sum_{m=k-T_Z}^{k+T_Z} N_{m,i}} & \left(\displaystyle\sum_{m=k-T_Z}^{k+T_Z} N_{m,i} \neq 0\right) \\ 0 & \left(\displaystyle\sum_{m=k-T_Z}^{k+T_Z} N_{m,i} = 0\right). \end{cases} \tag{3.5}$$

The variable $N_{m,i}$ represents the number of timeslots in the training data at the m-th timeslot of the day, the state of which is labeled as i. T_Z denotes the number of similar timeslots around the target. Parameter T_Z has a different value for each k; thus, we set the minimum value that satisfies $\sum_{m=k-T_Z}^{k+T_Z} N_{m,i} \neq 0$ for all states i. Even if $\sum_{m=k-T_Z^{max}}^{k+T_Z^{max}} N_{m,i} \neq 0$ is not satisfied for all states i, where T_Z^{max} is the maximum value for T_Z, we set T_Z as T_Z^{max}.

We next explain how to calculate the operation probability b which is used to correct the state probability α. Operation probability $b(i,x)$ denotes the probability of the number of the operations x of the IoT device in state i. $b(i,x)$ is calculated using Equation (3.6):

$$b(i,x) = \begin{cases} \dfrac{\sum_k N_{k,i}^{(x)}}{\displaystyle\sum_k N_{k,i}} & \displaystyle\sum_k N_{k,i} \neq 0 \\ 0 & \displaystyle\sum_k N_{k,i} = 0. \end{cases} \tag{3.6}$$

Note that $N_{k,i}^{(x)}$ represents the number of occurrences of operation x in the state i in the k th timeslot of the day. If there are no operations x in the training data, $b(i,x)$ is set to 1 for all states i to avoid incorrect transitions.

Calculating state probability

The proposed method calculates the state probability α for each timeslot of the training data by the calculated a and b. Then, the proposed method stores the sequences of home events using the training data by the estimated home states because the proposed method stores sequences not only in the current home state but also in similar states. To determine the similar states, we use the calculated state probability. By storing sequences in the states that satisfy the conditional probability expression, we can store sequences in the similar states.

When the timeslot changes, the state transitions from the state of the previous timeslot using the learned state transition probability, a. First, the proposed method calculates $\hat{\alpha}_t(i)$, the probability of state i when the timeslot is changed to t, using the learned state transition probability $a_k(i, j)$.

$$\hat{\alpha}_{T(t)}(i) = \sum_j a_{K(t)}(j, i)\alpha_{(T(t)-\Delta T)}(j). \tag{3.7}$$

Variable $K(t)$ is a function that returns the corresponding $K(t)$-th timeslot of the day with timeslot t; $T(t)$ is a function that returns the time corresponding to timeslot t, and ΔT indicates a very small time. By considering the case that, in the previous timeslot, the state probability, α, is updated by using the operation probability, b, Equation (3.7) uses the state probability at $T(t) - \Delta T$. Then, the proposed method calculates α based on $\hat{\alpha}$ so that the sum of the state probabilities of each state is 1 using Equation (3.8):

$$\alpha_{T(t)}(i) = \frac{\hat{\alpha}_{T(t)}(i)}{\sum_j \hat{\alpha}_{T(t)}(j)}. \tag{3.8}$$

When we observe an operation x of a home IoT device, the proposed method updates the state probability α using the operation probability $b(i, x)$. First, the proposed method calculates $\hat{\alpha}_{T(x)}(i)$ according to Equation (3.9).

$$\hat{\alpha}_{T(x)}(i) = b(i, x)\alpha_{(T(x)-\Delta T)}(i), \tag{3.9}$$

where $T(x)$ represents the time when the proposed method observed operation x. Then, the proposed method calculates the state probability, α, after the operation of the home IoT device using Equation (3.8).

Storing sequences

Based on the calculated state probability α, the proposed method stores the behavior sequences to estimated states. First, we must generate the sequences based on the observed operations and the users entering and leaving. This will account for multiple users operating devices within T_{seq} s of each other. When the users operate devices from their respective smartphones, we can identify correct behavior sequences by classifying those who operated which home IoT device based on the IP address of the operating smartphones. There are many cases where it is impossible to distinguish which user performs each operation. Thus, as with the sequence-based method [32], we generate multiple types of sequences from a simple series of events by removing some of them for training. For example, when actions A, B, and C are performed within T_{seq} s, equations $\mathit{Diff}(\mathit{time}_A, \mathit{time}_B) \leq T_{seq}$, $\mathit{Diff}(\mathit{time}_A, \mathit{time}_C) \leq T_{seq}$, and $\mathit{Diff}(\mathit{time}_B, \mathit{time}_C) \leq T_{seq}$ are satisfied. time_A, time_B, and time_C represent the times when actions A, B, and C are performed, respectively. In this example case, we generate and use all seven types of event sequences: A-only, B-only, C-only, A-B, B-C, A-C, and A-B-C. If actions A and B are performed by the same user, and action C is performed by another, the correct event sequences, A-B and C-only, are learned. However, incorrect sequences, such as A-only, B-only, A-C, B-C, and A-B-C are also stored. If sequences A-B and C-only are frequently performed by users, the correct sequences will be stored multiple times. Therefore, by using only the sequences that are greater than or equal to a given threshold, we can identify frequent behaviors.

After generating the sequences, event sequence y, which is related to the operation of the detection target home IoT device, is stored for each state in which the sequences are performed. We can determine the states for which the proposed method stores the sequences from the calculated probability, $\alpha_t(i)$, in state i at timeslot t. We select either Equation (3.10) or (3.11) and store the sequences into all states satisfying the selected one.

$$\alpha_{(T(y)-\Delta T)}(i) \leq L_\alpha, \tag{3.10}$$

$$\mathit{Rank}(\alpha_{(T(y)-\Delta T)}(i)) \leq L_{\mathit{Rank}}. \tag{3.11}$$

Note that $T(y)$ represents the time during which sequence y occurs. Here, $\mathit{Rank}(\alpha_{(T(y)-\Delta T)}(i))$ is a function that returns the number from the top of the state probability of i of all states, such as 1st, 2nd, etc. L_α and L_{Rank} are the parameters. When there are no states satisfying the selected equation, the proposed method does not store the sequence.

After storing the sequences, we calculate the behavior sequence probability $b'(i, y)$ in estimated state i of the sequence y for detection. We can then calculate b' using Equation (3.12):

$$b'(i, y) = \begin{cases} \dfrac{M(i, y)}{N'_i} & N'_i \neq 0 \\ 0 & N'_i = 0, \end{cases} \tag{3.12}$$

where $M(i, y)$ is a function that returns the occurrence times of sequence y in the estimated state i from the training data, and N'_i is a function that outputs the number of timeslots of the training data estimated as state i.

3.3.3 Detection using the learned model

After training the model, when an event sequence, y, includes operations of the detection target device, the proposed method calculates the state probability $\alpha_{(T(y)-\Delta T)}(i)$ using Equations (3.7), (3.8), and (3.9). The proposed method calculates the probability of occurrence $\delta_{(T(y)-\Delta T)}(y)$ of the sequence y by multiplying the state probabilities, $\alpha_{(T(y)-\Delta T)}(i)$, by the behavior sequence probability, $b'(i, y)$, as described in Equation (3.13):

$$\delta_{(T(y)-\Delta T)}(y) = \sum_i b'(i, y)\alpha_{(T(y)-\Delta T)}(i). \tag{3.13}$$

When the calculated occurrence probability, δ, satisfies Equation (3.14), the proposed method detects the operation as an anomalous operation.

$$\delta_{(T(y)-\Delta T)}(y) < n_{L(y)}. \tag{3.14}$$

Function $L(y)$ returns the length of sequence y; the length of the sequence reflects the number of events comprising the sequence. $n_{L(y)}$ is a parameter of the sequence constructed by $L(y)$ events. We set multiple thresholds for each length of the sequence because long sequences are rare.

3.4 Evaluation

To evaluate the proposed method, we simulated anomaly detection using data from two real homes. We evaluated the effectiveness of each part by comparing the detection results to the results of alternative methods.

In this evaluation, we chose the operations of a cooking stove as the detection target device. We prepared the proposed method according to the target device and the home environments.

3.4.1 Evaluation environment

Here, we describe the details of the detection simulation of the proposed and compared methods. First, we explain how the datasets were collected. Then, we set the proposed anomaly detection method suitable for each home by defining the states of the home and the labeling rules. Thereafter, we describe the metrics of the comparison and present the results.

Data collection in real homes

We collected data of user behaviors and observed the values of the installed home IoT sensors from two real houses, A and B[1]. Home A had two users who operated devices, and home B had one. We used monthly data of each home as one case for the simulation, resulting in 20 cases. We describe the case using the data of home A as $A_1, A_2, \ldots, A_{10}$ and home B as $B_1, B_2, \ldots, B_{10}$.

First, we collected the date information of events, including operations of consumer electronics and user entry/exit statuses, as shown in Table 3.2. Because each home included home appliances that were not connected to the internet, we collected their information by asking users to record their device use times. For the simulation, we assumed that each home appliance was an IoT appliance, and the recorded operation logs were used for the purposes described. Logs were compiled as buttons were pressed on the home appliances and when users entered and left the home. Because there were several omissions in the collected logs, we corrected them via labeling rules, as described in Section 3.4.1.

[1]The collection experiment of data on the in-home activities of users and sensor values in real homes received approval from the Research Ethics Committee of the Graduate School of Information Science and Technology, Osaka University.

Table 3.2: Collected operations and events by our experimental system deployed in real homes.

Device or event	Action
User position	Entry / Exit
Room light	On / Off
Air conditioner	Cooling / Heating / Turning up / Turning down / Off
Electric fan	On / Off
Heater	On / Off
Washing machine	On
Refrigerator	Opening
TV	On / Off
Cooking stove	On / Off
Microwave	On
Toaster oven	On
Rice cooker	On

Table 3.3: Collected sensor data from installed IoT sensors in real homes.

Sensor data	Range of sensor values
Room temperature	0 - 50°C
Humidity	0 - 100%
Atmosphere	260 - 1,260 mbar
CO2	0 - 5,000 ppm
Noise	30 - 130 dB

Then, we installed IoT sensors in each home and collected the sensor values shown in Table 3.3 in 5 min intervals.

Settings of anomaly detection method

To simulate anomaly detection for a cooking stove, we set up the state of the home and labeling rules. For this evaluation, the timeslot was assumed to be 1 min for capturing state transitions.

Setting home states We set the usage state of the devices d based on cooking states: $d \in \{use, before, after, none\}$ because cooking stoves are frequently used during cooking. State *use* refers to the cooking state, *before* and *after* indicate times before and after cooking, respectively, and *none* implies other states. Note that to grasp the cooking state exactly, we also used operations of the cooking appliances other than the cooking stove to label the states d. Specifically, when the cooking stove, microwave, toaster oven, or rice cooker was operated, we set d as *use*; the details are described in Section 3.4.1.

We set the activity state of the users as $u \in \{\mathit{active}, \mathit{out}, \mathit{sleep}\}$. Hence, at least one person was active, everyone in the home was out, or everyone was sleeping, respectively.

We set the home states $s_{u,d}$ by combining u and d. However, states $s_{\mathit{out,use}}$ and $s_{\mathit{sleep,use}}$ did not exist because users cannot cook while they are sleeping or out of the home. Hence, we set 10 states excluding the above for detection.

Labeling rule Using the defined states from Section 3.4.1, we labeled each timeslot of the training data. In consideration of privacy concerns, we labeled the home states from the observed information taken from the IoT devices and sensors. In particular, because there were several omissions in the collected logs, we corrected them based on the rules.

The activity states of the users u are labeled as follows.

out: The timeslots that the home was empty were tabulated by counting the number of users in the home based on their entry and exit time information. However, when we observed an operation of a home IoT device, we changed the number of users in the home to 1 and set the states of the timeslot after the time corresponding to the change. This is because the logs included some omissions of entries and exits. In this case, we excluded logs of the day from the calculation of a and b.

sleep: The timeslots at night containing noise values were lower than a threshold, and the CO2 concentration value was higher than a threshold; the installed IoT sensors in each home sensed the values. We defined the thresholds by the sleeping time that we asked of the subjects, including the noise and CO2 values of the sleeping time. Concretely, we defined the night from 22:00 to 9:59, the noise threshold as 35 dB, and the CO2 threshold as 1,500 ppm in home A and 400 ppm in home B. When two *sleep* timeslots existed within 90 min, we labeled the timeslots between the two as *sleep*, because the indicators were temporarily lowered during sleep. When we observed an operation in *sleep* states, we corrected the states to *active* because more than one user was awake and active. Concretely, when a user operated devices in the time frame of 22:00 to 4:59, we changed the user states to *active* before 5 h from the time of operation; when a user operated devices at the time from 5:00 to 9:59, we changed the user states to *active* after 4 h from the timeslot during which the device was operated.

active: This refers to states other than *out* and *sleep*.

Then, the usage states of device d (i.e., cooking or not) in this evaluation are labeled as follows.

use: This refers to timeslots in which a user operates a cooking appliance, including the cooking stove, microwave, toaster oven, and rice cooker. Because the cooking continues for a certain time, we set the T_C timeslots after the operating cooking appliances as *use*, where the T_C is a parameter of cooking time. We did not include the refrigerator in the cooking appliances because it is used frequently even when users are not cooking. Furthermore, when there are two *use* states within 15 min, we labeled the timeslots between the two as *use*.

before: This indicates the T_X timeslots before *use*.

after: This indicates the T_Y timeslots after *use*.

none: This indicates states other than the *use*, *before*, and *after*.

We labeled the home states, $s_{u,d}$, by combining the labeled states of the users u and those of the devices d.

Metrics

We evaluated the proposed method using two metrics: detection and misdetection ratios. For the simulation, we mixed 100 anomalous operations of the cooking stove at random times during the day. Furthermore, we considered the actual operations of the home IoT devices originally included in the recorded log as legitimate operations. The detection ratio and misdetection ratio was calculated using Equation (3.15) and (3.16).

$$\text{Detection ratio} = \frac{TP}{TP + FN} \tag{3.15}$$

Here, *TP* is the number of true positives of detected anomalous operations; *FN* is the number of false negatives; and $TP + FN$ equals $100N^{days}$, where N^{days} indicates the number of days included in the detection data.

$$\text{Misdetection ratio} = \frac{FP}{FP + TN.} \tag{3.16}$$

Here, the *FP* is the number of false positives that are legitimate operations the methods could not determine as legitimate; the *TN* is the number of true negatives.

For the evaluation, we used cross-validation. First, we trained the models with data for one month excluding one day. Then, we simulated the detection of the trained model using the excluded data. By changing the excluding day and summarizing the detection results, we obtained a detection result from the monthly data.

We changed the parameter values in each combination respectively and collected the combinations of detection and misdetection ratios. We describe the detection results as figures with the misdetection ratio on the horizontal axis and the detection ratio on the vertical axis. Thus, we only plotted the results having the highest values on the vertical axis among the results that were less than or equal to the values on the horizontal axis.

Note that when the operation of the target device occurred, a decision was made based on the sequence that was generated up to and just before the operation. Hence, the operations subsequent to the target operation were not used for the detection of the target operation.

Compared methods

To evaluate the effectiveness of the proposed method, we compared it to the other methods. Thus, we demonstrated the improvements gained by combining the sequence information. By comparing with the sequence-based method, we confirmed the effectiveness of estimation of the in-home situation. The differences between the proposed method and the compared methods are described in

Table 3.4: Differences between the proposed and compared methods.

Method	State	Sequence
Proposed	Estimating situation	✓
Estimation-based [34]	Estimating situation	
Sequence-based [32]	Time of day	✓

Table 3.4.

Estimation-based anomaly detection method We compared our new method to the estimation-based anomaly detection method. This method estimates the states of the home based on the sensed values and operating statuses of IoT devices. As with the proposed method, the estimation-based method calculated the b and α using Equations (3.5-3.9). We calculated the probability that the operation was legitimate by multiplying b to α and by summarizing them. If the value was higher than a threshold θ the equation was regarded as legitimate, as shown in Equation 3.17.

$$\sum_i \alpha_{T(x_c-\Delta T)}(i) b(i, x_c) > \theta, \tag{3.17}$$

where x_c denotes the operation of the cooking stove.

Sequence-based anomaly detection method We compared our new method to the sequence-based anomaly detection method [32]. This method models the behaviors of users from sequences of events in the home at each time of day. The sequence includes operations performed within T_{seq} s. When this method observes a sequence related to the detection target operation, it counts the number of stored equivalent sequences that occurred during the time of day within α_{seq} s of the observed sequence. When the ratio of the counted number of all stored operations of the target device was greater than or equal to the threshold, n_d^{seq}, the target operation included in the sequence was judged as legitimate. n_d^{seq} is the parameter, and the d denotes the length of the sequence.

Anomalous operations of the cooking stove must be detected immediately because such operations present higher risks to users compared to other devices such as TVs, air conditioners, etc.

Table 3.5: Values of parameters for the proposed and compared methods.

Parameter	Set values
T_X	$15, 30, 60, 100.$
T_Y	$15, 30, 60, 100.$
T_C	$10, 15, 20, 30, 45, 60.$
$n_l(l = 1)$	$0.0, 1.0 \times 10^{-7}, 2.0 \times 10^{-7}, \ldots, 1.0.$
$n_l(l \geq 2)$	$0.0, 1.0 \times 10^{-7}, 2.0 \times 10^{-7}, \ldots, 1.0.$
T_{seq}	600
L_α	$1, 2, \ldots, 10.$
L_{Rank}	$0.00, 0.05, 0.10, \ldots, 1.00.$
α_{seq}	$0, 900, 3600, 10800, 32400, 43200.$
$n_d^{seq}(d = 1)$	$0.00, 0.02, 0.05, 0.1, 0.15, 0.20, 0.25,$ $0.30, 0.35, 0.40, 0.45, 0.50, 1.00.$
$n_d^{seq}(d \geq 2)$	$0.00, 0.02, 0.05, 0.1, 0.15, 0.20, 0.25,$ $0.30, 0.35, 0.40, 0.45, 0.50, 1.00.$
T_Z^{max}	720

Therefore, for this evaluation, the proposed and sequence-based methods could only use the sequences leading up to the target operations. Additionally, the cooking stove was often operated as the first event of a sequence when users wished to cook. These points differ from the evaluation of the previous sequence-based method [32].

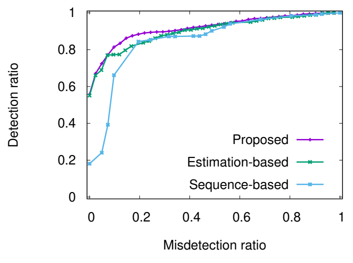
Parameter values

Training and detection were performed for each combination of values set in Table 3.5. We simulated all combinations with each value set for each parameter and evaluated the detection results.

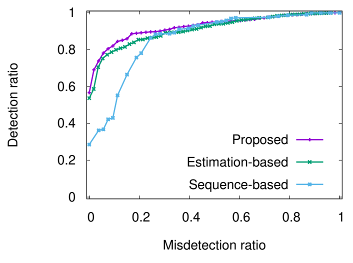
3.4.2 Evaluation results

The evaluation results of the proposed and compared methods for each month are shown in Fig. 3.3 and 3.4.

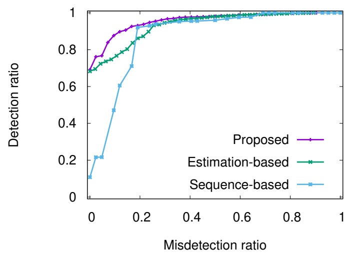
The proposed method achieved a higher detection ratio with the same misdetection ratio of the sequence-based method in the case of home A_1, A_2, ..., A_{10}. In particular, compared with the highest detection ratios having less than 10% misdetections, the proposed method achieved a 46.0% higher detection ratio than the sequence-based method in home A_{10}. This occurred because



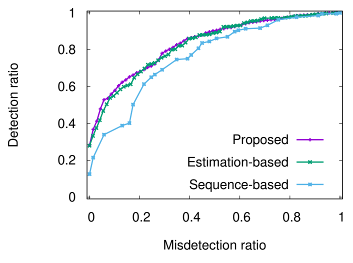
(a) A_1



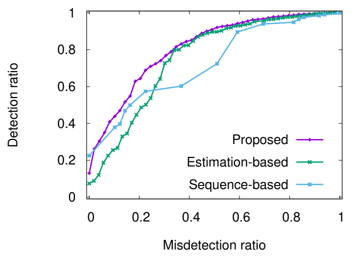
(b) A_2



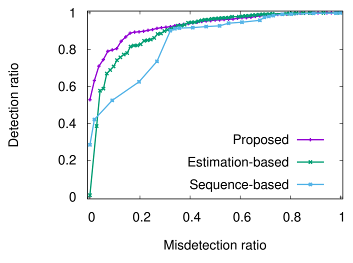
(c) A_3



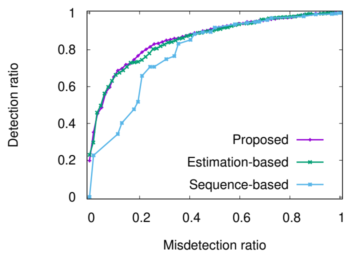
(d) A_4



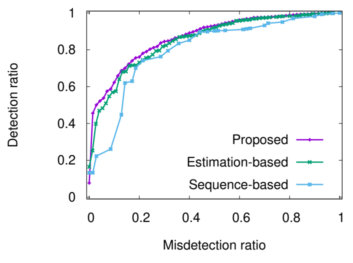
(e) A_5



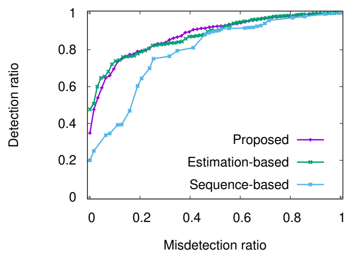
(f) A_6



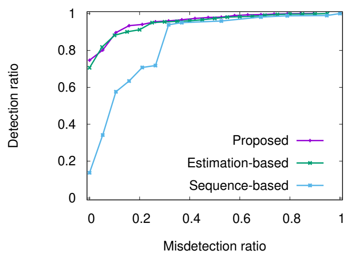
(g) A_7



(h) A_8



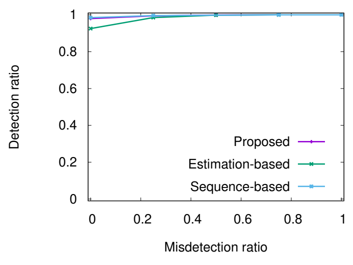
(i) A_9



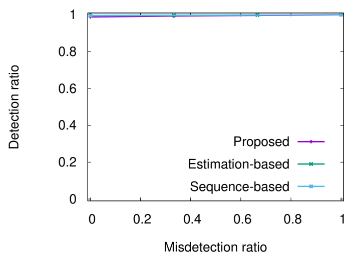
(j) A_{10}

Figure 3.3: Detection results for each month's data of home A.

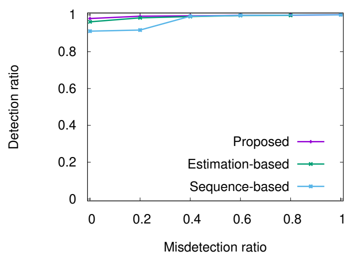
the proposed method has a narrower time range regarded as legitimate than the sequence-based method. When the sequence-based method tries to reduce misdetections, the legitimate range is



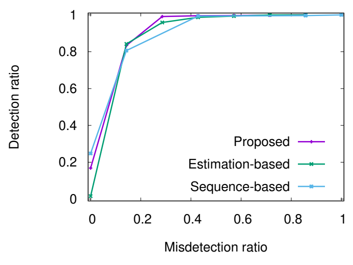
(a) B_1



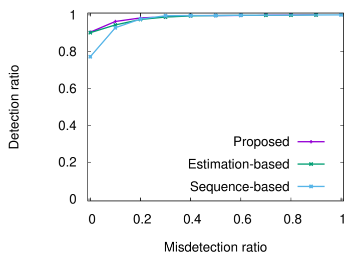
(b) B_2



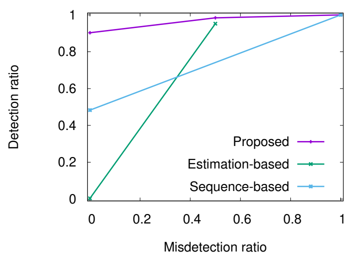
(c) B_3



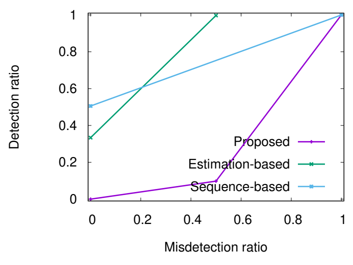
(d) B_4



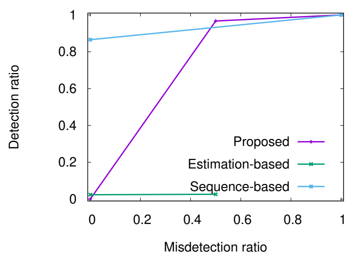
(e) B_5



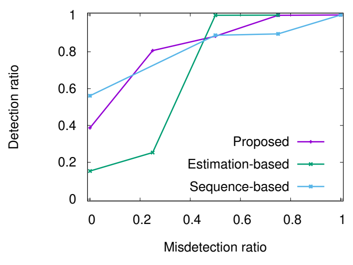
(f) B_6



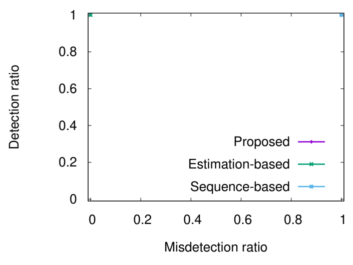
(g) B_7



(h) B_8



(i) B_9



(j) B_{10}

Figure 3.4: Detection results for each month's data of home B.

expanded as the learning data increases because the devices are operated at various times of day. In contrast, because the proposed method estimates the state according to the state of each day, the legitimate time range can be narrowed. Furthermore, the detection of single operations by the proposed method is another reason for its improved performance; the single operation means that there are no other operations before T_{seq} s. When the sequence-based method cannot use short-term information, it cannot determine legitimate/anomalous operations because it must learn from only the time-of-day information. Because the proposed method learns from the short- and long-term information, it can determine whether single operations are legitimate or anomalous from the long-term information. However, the detection results of the methods were almost the same in homes B_1, B_2, B_3, B_4, and B_5. In these cases, the operations were performed at the same time of day and were included in the same sequences. Thus, the sequence-based method learned the behaviors accurately.

The proposed method also achieved a higher detection ratio with the same misdetection ratio of the estimation-based method in the case of homes A_1, A_2, A_3, A_5, and A_6. In particular, compared with the highest detection ratios having less than 10% misdetections, the proposed method achieved a 15.4% higher detection ratio than did the estimation-based method in home A_3. This occurred because the proposed method can learn the relations between operations of the cooking stove and the operations of other frequently used devices, which included air conditioners, heaters, room lights, washing machines used in the morning, and refrigerators. As an example of a legitimate behavior, a user might turn off a heater before using the cooking stove in order to regulate the ambient temperature. The estimation-based method only determines whether the users are about to cook. When users operated non-cooking devices, the probability of cooking was only slightly increased, and the estimation-based method could not determine the state. However, the proposed method can learn the behavior sequence including the operations of such devices to grasp the legitimate operations of the cooking stoves. In contrast, when there were fewer operations related to non-cooking devices, such as in homes A_4, A_7, A_8, A_9, A_{10}, B_1, B_2, B_3, B_4, and B_5, the detection results of the proposed method were slightly improved. During the recorded months, devices such as heaters and air conditioners were not used, and their operations were almost always single-use or used with cooking equipment. Therefore, nearly all operations of the cooking stoves could be determined as legitimate or not by estimating the home states.

The detection results of all methods were not stable in homes B_6, B_7, B_8, B_9, and B_{10}. The numbers of operations included in these cases were too small to train the behavior models sufficiently. However, the misdetections in those homes were not significant because there were only a small number of operations.

3.5 Conclusion and Future Works

To detect anomalous operation attacks on IoT devices in a home, we proposed a detection method that estimates the home state based on the observed values of IoT sensors and device operations and learns the event sequences of users in the home in each estimated state. After training, when a device operation is observed to determine whether it is legitimate or anomalous, the proposed method calculates the occurrence probability of the sequence related to the target operation. If the occurrence probability is lower than the threshold, the operation is detected as anomalous. For this evaluation, we simulated anomaly detection using behavioral logs and sensor data obtained from real homes for one month. We evaluated the improvements of the proposed method and the effectiveness of each part by comparing the proposed method to other methods, one of which did not use sequence information and the other did not estimate the in-home situation. We found that the proposed method achieved a 15.4% higher detection ratio with fewer than 10% misdetections by using the sequence information, and it achieved a 46.0% higher detection ratio with fewer than 10% misdetections by using the estimation of the in-home situation. Thus, the proposed approach can analyze the legitimate behavior of users and legitimate usages of the IoT devices comprehensively by using long- and short-term information, that is, by estimating the home state transition and using the sequence of behaviors. However, a certain amount of data was required to learn the behaviors of users in the home.

In this chapter, we simulated the proposed method by setting a cooking stove as the target device. Evaluating the proposed method when other devices are used as detection targets remains as a future task. Furthermore, although we used data for one month for this evaluation, another future task will involve collecting data for a longer period of time and from many actual homes to verify the utility of the method.

Chapter 4

Privacy-Preserved Cooperation Framework for Anomaly Detection without using Private Information

4.1 Introduction

4.1.1 Motivation

Consumer electronics such as electric fans and refrigerators have recently been connected to the Internet; these devices are called Internet of Things (IoT) devices. Users can operate these IoT devices by using smartphones and AI speakers via the Internet. Owing to the usefulness of IoT devices, many IoT devices have been installed in homes.

Caused by the popularity of home IoT devices, risks of cyberattacks targeting home IoT devices [40–43] and smart homes [53, 54] have increased. Cyberattacks targeting home IoT devices have been previously observed [3,4]. Currently, these attacks mainly aim to intrude the IoT devices to construct botnets and abuse the IoT devices as step devices for DDoS attacks [5,6]. Nevertheless, these attacks can be detected by analyzing the behavior of the attacker [9–11] and comparing it to the usual behaviors displayed by the home occupants [12, 13].

However, these attacks differ from the attacks that target personal computers and smartphones. This is because home IoT devices are physically close to users [14]. For example, attackers would operate home IoT devices by sending packets via the intruded IoT devices to change the temperature of an air conditioner or unlock a smart home lock. These attacks may make users unsafe and could even cause physical harm. Furthermore, simultaneous attacks on high-power IoT devices can suddenly increase energy demands, which could lead to major power outages [16]. Therefore, the detection and prevention of these attacks are of paramount importance.

To mitigate the risk of attacks on home IoT devices, we proposed a method that could detect attacks that targeted home IoT devices [32]. This method focused on the daily behaviors of the users at home. The method learned their behaviors by studying a sequence of events and the conditions of the home. The observed sequence of events included events on the home network such as the operations of the IoT devices and the entry and exit of the users. The condition of the home was a combination of recorded sensor values when IoT devices were being operated; these include the time of day, temperature, and humidity. When an operation of a home IoT device differed from the learned behavior, this method detected the operation as an anomalous operation.

The simulation results indicated that the method achieved a 95–100% detection ratio of anomalous operations with less than 20% of misdetections when using a sufficient amount of training data. Nonetheless, when we did not have a sufficient amount of training data, the method could not correctly learn the behaviors of the users. In this case, the method recorded false negatives of anomalous operations and continued to do so until a sufficient amount of operations had been monitored.

However, anomalous operations still need to be detected regardless of whether the data amount is sufficient or not. An intuitive approach is to analyze the data of other users by collecting the behavioral datasets of many users [27]. This method uses the privacy information of the users, including the in-home activities of the users; thus, the privacy of the user should be held in high regard. Another approach, which does not rely on the sharing of the behavioral datasets, is to train each model on each dataset and to construct a general model by sharing the learned results [25, 26]. However, this general model constructed via the cooperation between agents may not match the lifestyle of each user. Therefore, this approach cannot achieve accurate detection of anomalous operations.

4.1.2 Problem statement

Owing to the issues mentioned above, a cooperation framework for agents to detect anomalous operations of home IoT devices that satisfy the following requirements is needed.

1. The framework must not use any information to identify the individual users. The framework must not assign any identifiers to the users and agents. In addition, the agents must not share the personal information of the users including the historical behavior of the users and their personal information, such as ages, genders, and jobs.

2. The agents must avoid cooperating with agents of users who have different lifestyles, which may cause inaccurate detections of anomalous operations.

4.1.3 Contribution and organization

In this chapter, we propose a cooperation platform to utilize the data of similar users for anomaly detection of home IoT devices without sharing private information. In this platform, each home has an agent that learns and detects anomalous operations in the home. When an agent cannot decide whether a current operation is legitimate or anomalous, it sends requests to the other agents via the platform. Only similar agents reply to the requests with only one-bit information that is legitimate or anomalous without sharing personal information. In our platform, an identifier is set to a request and is used to judge the similarity. When an agent receives a request that includes behaviors that are similar to the behaviors learned by the agent, the received agent stores the ID of the request. When the agent wants to send a request, it attaches the stored IDs to the request. When an agent receives the request, the agent answers the request that includes IDs that have been stored by the agent. By doing so, the other agents identify the similarity between themselves and the agent sending the request. That is, in our framework, agents can cooperate without sharing the identity of the users. In addition, each agent can choose to answer the request or not. That is, the agents can choose to cooperate with others or not to avoid sharing information that the user may be unwilling to share.

A key idea used in our platform is judging the similarity of each agent from the past answers without using the identifiers of the users. Hence, we can apply our platform to other systems such as shopping recommendation systems.

In summary, our main contributions of this chapter are as follows:

- We propose a new method to cooperate with similar agents without sharing private information including the identifiers and personal information of the agents and users.
- We simulate the proposed framework for the detection of anomalous operations of home IoT devices.
- We also demonstrate that the cooperation between similar agents by our framework improves the accuracy of the detection of anomalous operations targeting home IoT devices [32].

The rest of this article is organized as follows. We discuss related works, including anomaly detection methods, an anonymous communication method, and cooperative learning methods in Section 4.2. The proposed platform, which does not share private information but utilizes the dataset of similar users to detect anomalous operations via the cooperation of similar users, is described in Section 4.3. Then, we report the evaluation of our framework and the corresponding results in Section 4.4. Finally, we conclude and discuss possible future work in Section 4.5.

4.2 Related work

Related work are discussed in this section. We explain anomaly detection methods of home IoT devices in section 4.2.1. Then we explain an anonymous communication method, which can be used in our framework, in section 4.2.2. Finally, we explain the methods that train machine learning models via cooperation and discuss the difference between our platform and these methods in section 4.2.3.

4.2.1 Anomaly detection method

Ramapatruni et al. proposed a method to detect anomalous operations by learning user behaviors. This method used Hidden Markov Models (HMM) to learn the normal activities of a user and collected the information obtained from the sensors and/or statuses of the home IoT devices as observations. By using the observations, this method learned the parameters of the HMM. Then, the trained HMM detected an anomalous operation when the probability of that operation occurring was low. They demonstrated the accuracy of this method by using the dataset collected in a smart home environment. This method focused on the case of a single user [24]. However, a smart home may have multiple users.

Therefore, we have proposed a method to detect anomalous operations even in the case of multiple users [32]. This method detects anomalous operations at the home gateway, which is connected to all home IoT devices, home IoT sensors, and smartphones. First, the home gateway collects two kinds of information. One is the condition information of the operations of home IoT devices, such as the time of day, room temperature, and humidity from the connected home sensors. The other information is the presence/absence of the users in the home from the attaching/detaching information of their smartphones. The home gateway subsequently classifies the conditions of the home by constructing a table of sensed values and stores the sequences of operations of IoT devices and the leaving/entering of users in each cell of the condition table. Finally, the home gateway judges whether legitimate or anomalous operations have occurred by comparing sequences of current operations with the stored sequences of the current condition. This method can handle the case with multiple users by constructing the sequences from the monitored operations and considering the case with multiple users.

We have also proposed a method to define the condition of the home for the detection of anomalous operations [34]. In this method, we defined the conditions of the home by the in-home activities. This method modeled the in-home activities of the users as a state transition model. We defined the state of the home as a combination of the state of the users and the state of the devices. The state of the users was defined by the multiple thresholds of sensor values, such as room temperature, noise, and pressures. The state of devices was defined by the time before or after their operation. This method calculated the transition probabilities. After the calculation, when an operation occurred, the method estimated the current condition by using the modeled transition probabilities. By using the estimated current condition, we could detect anomalous operations.

The above methods accurately detected anomalous operations when the amount of training data was sufficient. However, these methods overlooked a large number of anomalous operations until a sufficient amount of operations was monitored.

Therefore, in this chapter, we propose an anomaly detection platform to utilize the behaviors of similar users to achieve accurate detection, even if there is an insufficient amount of training data.

4.2.2 Anonymous communication

In our platform, agents need to hide their sender information, such as IP addresses and user identifiers, when they communicate with other agents. In this chapter, we used Tor [55] as an anonymization tool. Tor is a famous anonymous communication tool. By communicating via the Tor network, agents can hide their sender IP address information. In the Tor network, only the IP address information of the sender is hidden but the data of the sending packets are not encrypted. Our platform does not need the data file to be encrypted; thus, our platform uses Tor for the anonymization of communications.

4.2.3 Collaborative learning

There are some kinds of learning methods using multiple datasets.

One of the cooperative learning methods generates big data by collecting data from many clients in one place, such as a datacenter [27]. In this method, the collected data is used to train a machine learning model. When a new client that requires the model joins the service, the client receives the trained model. A new client receives one of the models trained by datasets of similar attributions and trains the model by using data collected by the clients. This method is vulnerable to attacks that may target the data center; the collected data may be leaked if the service on the data center is vulnerable. An example is when an attacker steals user data on the Internet cloud through a misconfigured web application firewall [28].

Privacy-preserving machine learning is an approach used to train machine learning models hiding private information [25]. One of the methods of privacy-preserving machine learning is to use differential privacy [56]. The differential privacy preserves the data privacy of users by adding random noise to the data. By collecting a large amount of the noised data, we can train a machine learning model to preserve privacy (the influence of noise can be eliminated statistically).

However, this approach is difficult to apply to the detection method of anomalous operations on home IoT devices. This is because the normal behavior of the users depends on their lifestyle, and accurate private information that can identify the lifestyles is required to achieve accurate detections.

Federated learning is an approach that trains machine learning models by the cooperation between users [26]. In this approach, an agent is deployed for each user. Each agent first trains the model independently by using the data obtained by itself. Then, agents share the trained models and construct the general model by combining the shared models. This approach can train the machine learning models without sharing private information. However, this approach also has difficulty in handling the lifestyles of each user. The general model constructed by the cooperation among all agents may not match the lifestyle of each user. As a result, this approach cannot achieve accurate detection of anomalous operations.

Table 4.1 shows the summary of the comparison. As shown in this table, any existing approaches do not satisfy the requirements mentioned in Section 4.1.2. Therefore, in this chapter, we propose a new framework that does not need the sharing of private information, where only agents with similar data can cooperate.

4.3 Platform to utilize similar data of users to detect anomalous operations without sharing private information

Fig. 4.1 shows an overview of our platform. In our platform, an agent is deployed for users in a home to learn the behaviors of the users and detect anomalous operations. When an agent cannot decide whether a current operation is legitimate or anomalous, it sends requests to the other agents via the platform to cooperate with them and decide whether the operation is legitimate or anomalous.

Each agent receiving the request first checks whether the request is sent from a similar user. Here, “similar” means that the user and the receiver of the request have the same behaviors. If the sender has a similar user, the agent checks whether the behavior of operations included in the request is legitimate or not based on its learned behavior. Then, the agents vote based on their

Table 4.1: Comparison of cooperative methods: The characteristics of various existing cooperative methods and the proposed method.

Characteristics	Centralized learning [27]	Centralized learning with differential privacy [56]	Federated learning [26]	Proposed platform
Centralized / distributed	centralized	centralized	distributed	distributed
Shared information	raw data	noised data	gradient of each trained model	yes/no answer
Generalized / personalized model	personalized	generalized	generalized	personalized
Requirements 1: Non-sharing private information		✓	✓	✓
Requirements 2: Utilizing only similar data	✓			✓

decision. By these steps, the platform collects the votes from similar agents. As a result, the agent that sends the request decides whether the current operation is legitimate or not by checking the results of the votes.

Our platform performs the above steps without identifying any agents. In our platform, the identifiers are set only to the requests. Thus, the other agents cannot identify the origin of the request.

The similarities between agents are calculated based on the IDs of the requests. The sender of a request attaches some IDs of past requests that the sender regards as legitimate to the request. Other agents use the attached IDs to identify if the sender of the request has learned similar behaviors.

4.3.1 Procedure of the agent sending request

Fig. 4.2 shows the procedure of the requesting agent. When a home gateway detects an operation of a device, an agent of the home checks whether the operation is legitimate or not. If the agent cannot determine whether the operation is legitimate or not due to a lack of learned data, it sends a request

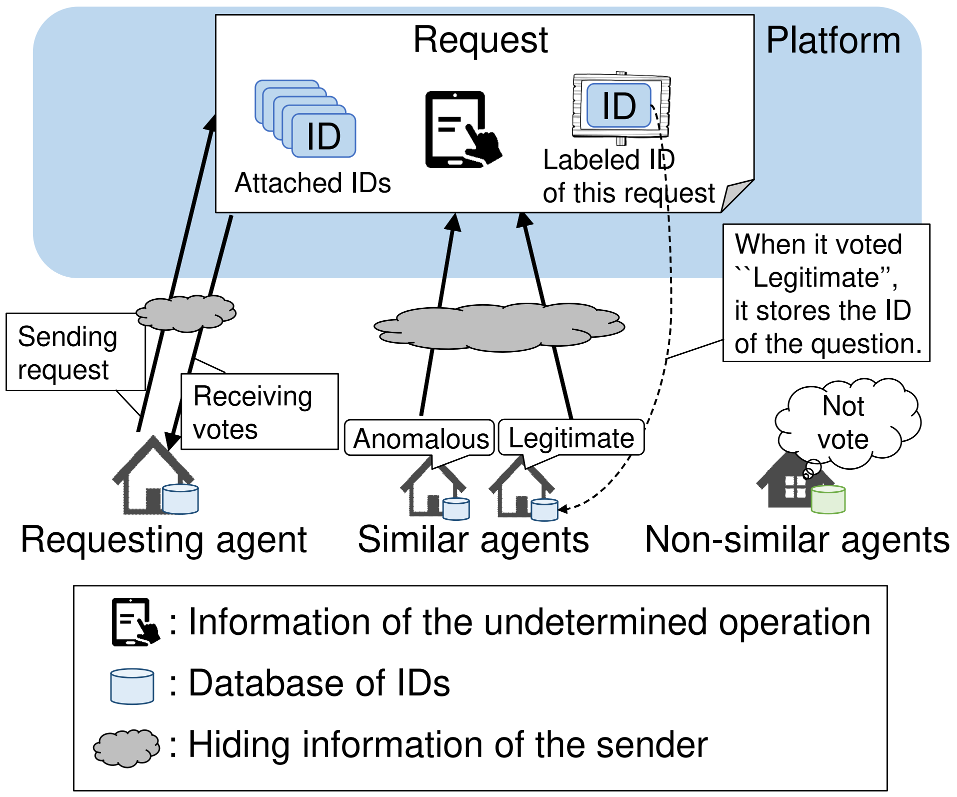

Request
Platform
ID
Attached IDs
ID
Labeled ID of this request
When it voted ``Legitimate", it stores the ID of the question.
Sending request
Receiving votes
Anomalous
Legitimate
Not vote
Requesting agent
Similar agents
Non-similar agents
: Information of the undetermined operation
: Database of IDs
: Hiding information of the sender

Figure 4.1: Overview of the proposed platform.

to the platform. The request includes the information of the undetermined operation that is used to identify whether the operation is legitimate or not. The sender randomly selects and attaches x number of IDs of the past requests that are identified to be legitimate by the sender agent; the x is a hyperparameter of attaching IDs. This information is used to check whether the sender has learned similar behavior to agents receiving the request. When the hyperparameter x is set to a certain value, the randomness of the selection is not significantly affected by the judgment of the similarity. When sending the request to the platform, the sender can also hide the information of who sent the request from the platform by using tools such as Tor [55]. After sending the request, the sender agent waits for the votes from the other agents.

When the agent receives the votes from the others via our platform, it calculates the number of votes for "Legitimate". If the number of votes to "Legitimate" is greater than the predefined threshold T, the agent regards the current operation as legitimate.

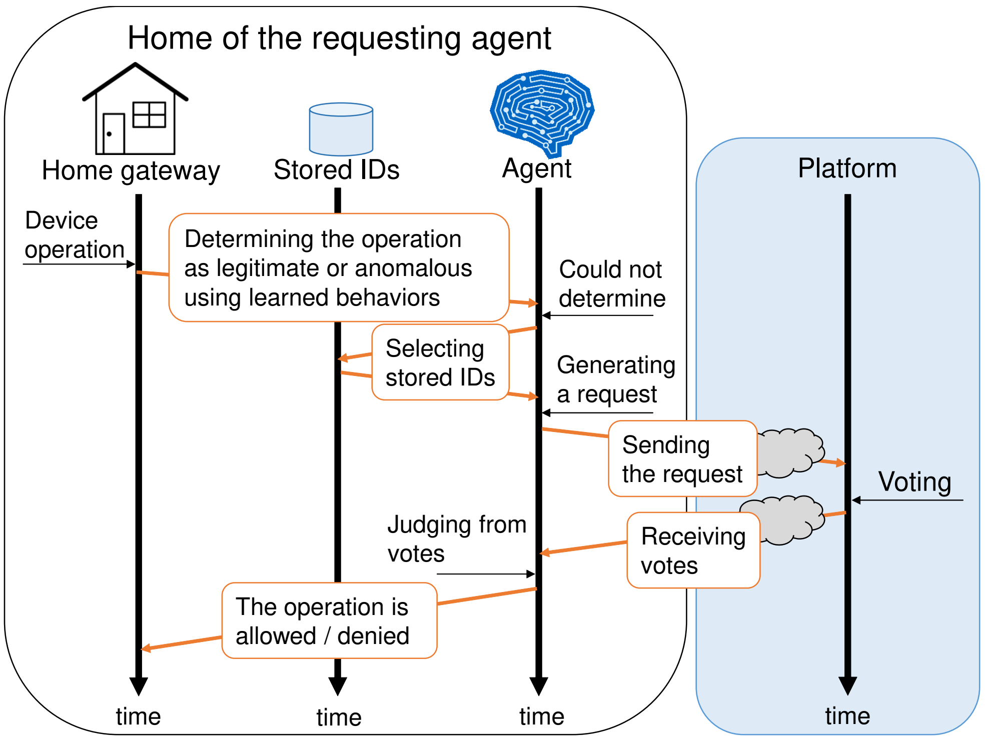


Figure 4.2: Flow chart of the requesting agent.

4.3.2 Procedure of agents receiving request

Fig. 4.3 shows a flow chart of the agents who receives a request. When an agent receives a request, the agent first checks the IDs attached to the received request. The agent then compares the attached IDs to the ID database that stores IDs of requests that the receiving agent identifies as legitimate. The similarity between the sender and receiving agent is judged by the number of matched IDs.

If the number of matched IDs is smaller than a threshold N, the agent does not vote for the request; this is because the sender has different behaviors. By doing so, we avoid the degradation of anomaly detection that could be caused by using data of non-similar users whose behaviors are different. Fig. 4.4 shows the procedure of the non-similar agents receiving a request.

If the number of matched IDs is larger than the threshold N, the agent judges the sender to be similar to itself. Subsequently, the agent checks if the behavior included in the request is legitimate or not by using its learned model. Then, it votes by returning its decision to the platform. The

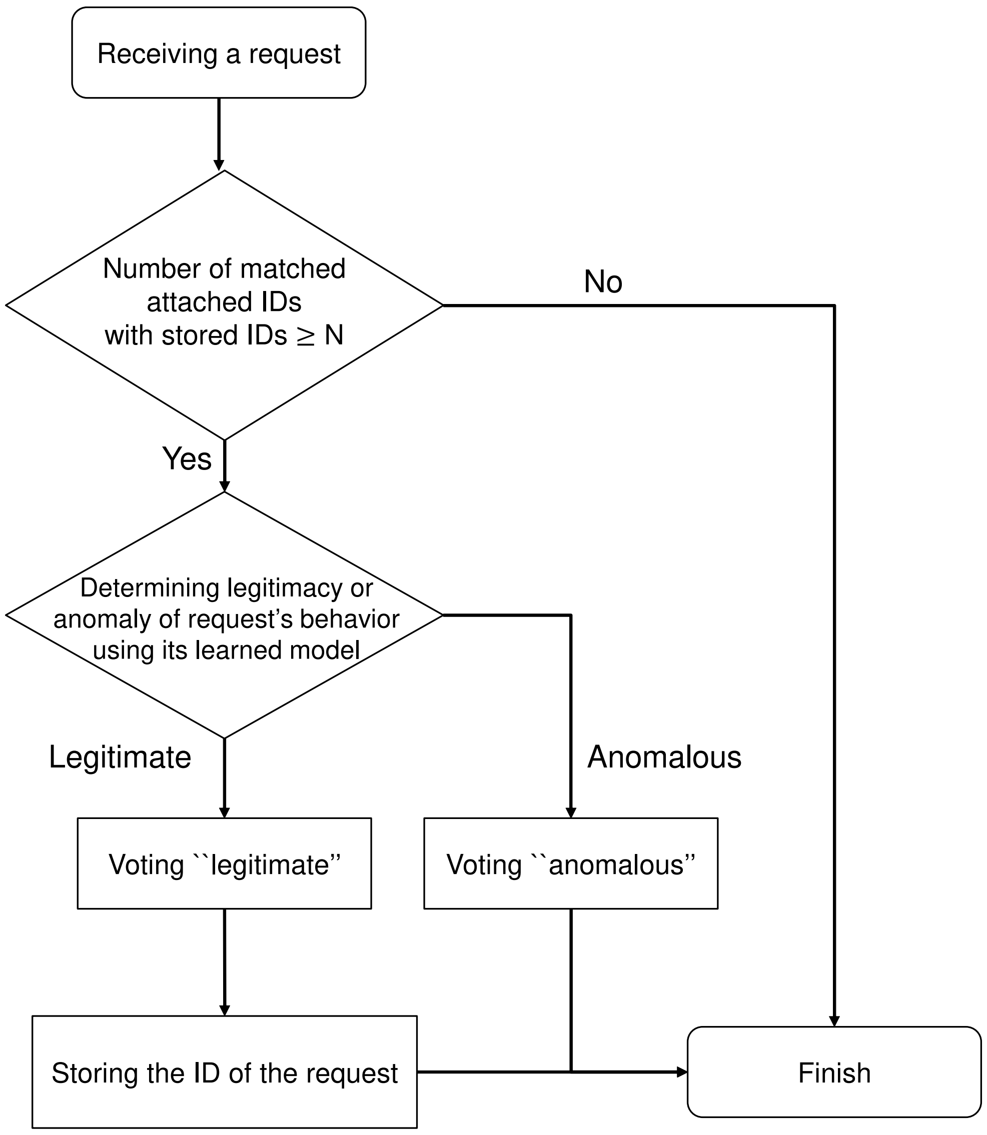

Receiving a request
Number of matched attached IDs with stored IDs ≥ N
No
Yes
Determining legitimacy or anomaly of request's behavior using its learned model
Legitimate
Anomalous
Voting ``legitimate"
Voting ``anomalous"
Storing the ID of the request
Finish

Figure 4.3: Flow chart of the agent receiving a request.

decision can either be "Legitimate", "Anomalous", or "Unknown". "Unknown" is a case where the agent does not have a sufficient amount of data to determine whether the behavior included in the request is legitimate or not.

In our platform, even if the number of matched IDs is larger than N, the agent can avoid voting if the user does not want to answer.

When the agent identifies the behavior included in the request as "Legitimate", the agent stores

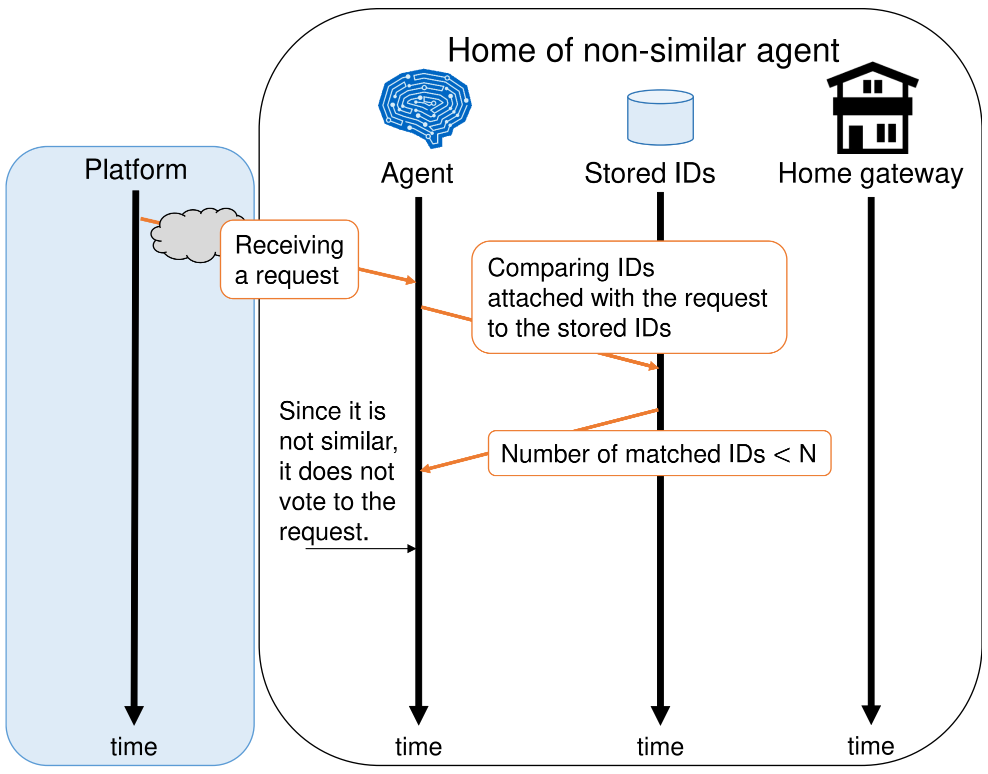

Home of non-similar agent
Platform
Agent
Stored IDs
Home gateway
Receiving a request
Comparing IDs attached with the request to the stored IDs
Number of matched IDs < N
Since it is not similar, it does not vote to the request.
time
time
time
time

Figure 4.4: Procedure for a non-similar agent receiving a request.

its ID into its ID database. The stored ID is used to identify the similarity of a future request of the agent. As the number of attached IDs to a request becomes larger, receiving agents can estimate the similarity more accurately. Fig. 4.5 shows the procedure of similar agents receiving a request.

4.4 Evaluation and Results

We have defined two requirements in Section 4.1.2; cooperation without sending personal information and cooperation with only similar agents. The former is achieved by the proposed platform because the platform does not require users to share their identifiers or their behaviors at home. Therefore, in this section, we demonstrate that our platform satisfies the second requirement. For this evaluation, we implemented and evaluated our platform and compared it to other methods by simulating on datasets captured in real homes. In addition, we checked which agents cooperated to show whether agents cooperated with similar agents.

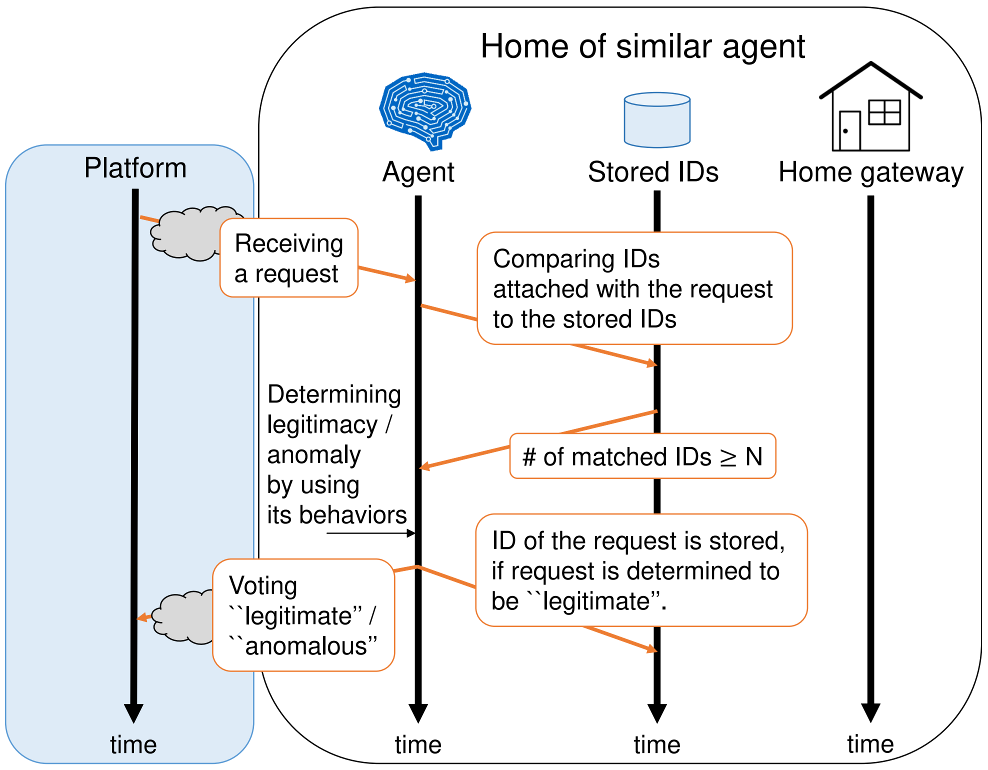

Home of similar agent
Platform
Agent
Stored IDs
Home gateway
Receiving a request
Comparing IDs attached with the request to the stored IDs
Determining legitimacy / anomaly by using its behaviors
# of matched IDs ≥ N
ID of the request is stored, if request is determined to be ``legitimate''.
Voting ``legitimate'' / ``anomalous''
time
time
time
time

Figure 4.5: Procedure for a similar agent receiving a request.

4.4.1 Evaluation Scenario

For this evaluation, we considered the case where a new agent was deployed at a home. The agent learned the behaviors of the users for five days; however, it did not have a sufficient amount of training data. Therefore, the agent cannot detect anomalous operations accurately without cooperating with other agents. Nevertheless, there still are agents that have been deployed at other homes before. These agents have enough data and can detect anomalous operations accurately. Therefore, the new agent would like to join our framework to cooperate with such agents who can detect anomalous operations accurately, regardless of the different lifestyles between the users and the user of the new agent.

To cooperate with the other agent, the newcomer agent needs to store the IDs of the questions similar to the behavior of the corresponding user. One approach to achieving this is to send past requests to the newcomer agents. This situation was evaluated with our method.

By this evaluation, we demonstrate that the cooperation within our framework improves the accuracy of the detection, even if the agent does not have sufficient training data yet.

4.4.2 Evaluation Environment

In this subsection, we describe the settings of our evaluation.

Dataset

To evaluate our platform, we collected datasets of activities in two real homes. We recorded the behaviors of the subjects living in the homes, including the time when home appliances were being operated, and the time the subjects entered and left the home. Some home appliances were not connected to the Internet, and we asked the subjects to record the time. To record the logs easily, we installed systems that included buttons, access points, and computers, as shown in Fig. 4.6. In this system, when a button was pushed, the button sent packets to a computer and the computer recorded the name of the button and the time of day. We put multiple buttons near each home appliance and named the buttons after the name of the home appliance. Their names and the action of the users are shown in Table. 4.2. We asked the subjects living in the homes to push the button when they used the corresponding consumer electronics or when they left or entered the homes. The collection of data on the in-home activities of users in real homes received approval from the Research Ethics Committee of the Graduate School of Information Science and Technology, Osaka University.

After collecting the logs, we divided the logs into multiple parts so that each part included the monthly data of a home. We collected data for 10 months, from September 2018 to August 2019, for home A. The data were divided and named $A_1, A_2, \ldots, A_{10}$. Data spanning 11 months was collected in home B, from January 2019 to November 2019, and was subsequently divided and named $B_1, B_2, \ldots, B_{11}$. For this evaluation, we simulated the case of multiple homes and some of the users who had have different lifestyles by considering each of A_1 to A_{10} and B_1 to B_{11} to the data of each home. That is, 21 homes participated in our framework in this simulation.

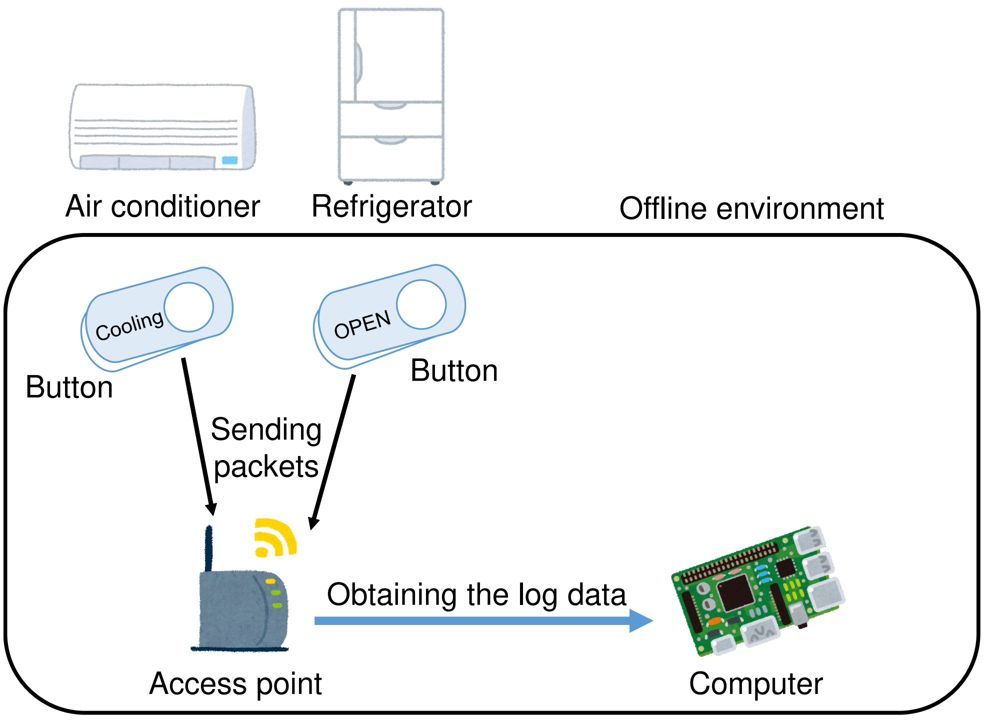


Figure 4.6: System for recording time logs of using home appliances in real homes.

Table 4.2: Collected operations and events by our experimental system deployed in real homes.

Device / event	Action
User position	Entry / exit
Lighting	ON / OFF
Air conditioner	Cooling / heating / dry / raise / lower / OFF
Electric Fan	ON / OFF
Heater	ON / OFF
Washing machine	ON
Refrigerator	Opening
TV	ON / OFF
Cooking stove	ON / OFF
Microwave	ON
Toaster oven	ON
Rice cooker	ON

Anomaly detection methods applied to each agent

In our platform, agents could cooperate with any anomaly detection methods. Each agent independently detected anomalous operations and asked other agents via our platform if the agent could not identify whether an operation was legitimate or anomalous.

For this evaluation, we applied our anomaly detection method [32] to each agent of our platform. This method learned the behaviors of the user as a combination of the condition of the home and the sequences of operations in the home. When users operated home IoT devices continuously, this method utilized the sequence information. According to the results of our previous work, the sequence of the operations plays a significant role in the identification of legitimate operations. However, we need a sufficient number of monitored sequences to train the model on the behavior of the users and achieve accurate detection. If each agent does not have a sufficient number of sequences, the agent alone cannot accurately identify legitimate operations. Thus, our framework, which enables cooperation between agents, is required. For this evaluation, we demonstrate that our platform works well for the agents that use the sequence of operations.

Though the sequence of operations is very powerful, we cannot use sequences that contain only a single operation; that is, the operations on the IoT devices that are used alone. In such a case, our detection method uses only the condition information, such as time of day, to identify the legitimate operations on such devices. The detection based on the condition information also requires a significant amount of the monitored operations that are used to train the model of the behavior of the users. However, the conditions where each IoT device is used depend on the lifestyles of the users. It should be noted that the information of the users whose lifestyles differ may degrade the accuracy of the detection. In this chapter, we also demonstrate that our platform works in such cases.

Therefore, we simulate two cases; (1) the case where the sequences of operations are monitored, and (2) the case where the device is used alone and the sequence cannot be used.

Metrics

For this evaluation, we used two metrics: the detection ratio and the number of misdetected legitimate operations.

We considered the operations by the users included in the dataset to be legitimate operations.

The number of misdetected operations was the number of legitimate operations detected as anomalous. For this evaluation, each home included a different number of legitimate operations. Therefore, the effects of one misdetection were different for each home. To ensure that the evaluation of the effectiveness of our method for each home was the same, we did not evaluate using the ratio of misdetections.

The detection ratio was defined by the number of anomalous operations detected as anomalous divided by the number of all inserted anomalous operations. For this evaluation, we inserted 100 anomalous operations per day into the test dataset at random times and calculated the detection ratio. For this evaluation, each home had a dataset with a different number of days. Since each home had a different number of inserted anomalous operations, we compared the detection accuracy using the ratio of detected anomalous operations.

The detection ratio of anomalous operations and the number of misdetected operations depended on the parameter of the detection methods and our framework. However, there was a tradeoff; the parameters set to detect more anomalous operations caused more misdetections. Therefore, we changed the parameters of the methods and obtained the ratios of the detected anomalous operations and the number of misdetected operations.

Compared methods

For this evaluation, we compared our cooperation platform with two methods; "Do not cooperate" and "Cooperate with all", as shown in Table. 4.3. "Do not cooperate" is where each agent does not use our platform but performs detection by only using the behavior data of the users monitored by itself. By comparing the cooperation platform to "Do not cooperate", we demonstrate the effectiveness of cooperation with other agents. "Cooperate with all" is where agents cooperate with all the other agents via our platform. By comparing the cooperation platform to "Cooperate with all", we demonstrate the effectiveness of cooperation with only similar agents.

Table 4.3: Details about proposed platform and comparative methods.

Method	Cooperation	Similarity
Cooperate with only similar (proposed)	✓	✓
Do not cooperate		—
Cooperate with all	✓	

Evaluation steps

We first selected a newcomer agent. Then we divided the one-month data corresponding to the selected agent into two half-month segments. We used the second part as the test data when the first part was used to train the model, and vice versa. By doing so, we can use the full data as the test data. After selecting the newcomer agent and the test data, we trained the model of the newcomer agent by using the data for the first five days that were not included in the test data. The other agents were the agents that cooperated with the newcomer agent and their models were trained using all of the data collected when the corresponding home was being monitored. After the models were trained, we simulated the preparation of the cooperation by sending past requests to new agents. For this simulation, we used the data of the first 14 days as the past requests sent to the agents. Finally, we simulated our framework again by using the test data of the newcomer agent. We evaluated all homes equally and only changed the new agent.

Parameters

Our platform has three parameters x, T, and N. We set x to 100, T to $1, 2, 3, \ldots, 20$, and N to $1, 2, 3, \ldots, 100$ for our framework. We set x to 100, T to $1, 2, 3, \ldots, 20$, and N to 0 for the "Cooperate with all" method.

The anomaly detection method used in this evaluation also had parameters T, α, n_1, $n_d (d \geq 2)$, and home condition. For this evaluation, we defined the home condition by using only the time-of-day information. We varied the parameters, as shown in Table 4.4 and 4.5.

Table 4.4: Parameter values of the anomaly detection method applied to agents of each home that learn by combining condition and sequence information.

Parameter	Set values
T	600
α	0, 900, 3600, 10800, 32400, 43200
n_1	0.00, 0.10, 0.25, 0.50, 1.00
$n_d (d \geq 2)$	0.00, 0.10, 0.25, 0.50, 1.00

Table 4.5: Parameter values of the anomaly detection method applied to agents of each home that learn only from the condition information.

Parameter	Set values
T	600
α	0, 150, 300, 450, . . . , 1200, 1500, 2100, 3600, 7200, 10800, 18000, 25200, 32400, 43200
n_1	0.05, 0.10, 0.15, . . . , 0.95, 1.00
$n_d (d \geq 2)$	1.00

4.4.3 Results

For this evaluation, we added anomalous operation for the cooking stoves and evaluated the accuracy of the detection.

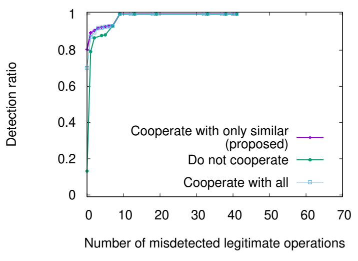
Fig. 4.7 and 4.8 show the detection results of the proposed platform and the compared methods when we use the sequences of the operations to detect anomalous operations. In this figure, the horizontal axis represents the number of misdetected legitimate operations and the vertical axis represents the detection ratio. The figure is a plot of the achievable detection ratio against the number of misdetected operations not exceeding the given value on the horizontal axis. These figures indicate that homes A_2, A_4, A_5, A_6, A_7, A_8, A_9, B_1, and B_5 has a reduction in the number of false negatives when using our platform. They achieve higher detection ratios than the non-cooperation method when the parameters are set to ensure that the number of misdetections is less than 20. Specifically, home A_5 records a significant reduction in the number of false negatives. The detection ratio of the proposed method for home A_5 is 50.5% higher than the non-cooperation method when the parameters are set to achieve less than four misdetections; this is less than one misdetection per week. The number of misdetections is greater than 23 for home A_7; this is because one of the half-month data for the home does not include the operation data of the first five days. When we ignore the 23 misdetections, the results show that our platform can reduce the number of false negatives. In other words, cooperation improves the accuracy of the detection. This is because the agents cannot learn the behaviors of the users sufficiently from the data of the first five days. Nevertheless, by using our platform, the agents avoid the case where false negatives of anomalous operations are caused by the lack of training data by cooperating with similar agents.

When we have enough training data in each home, which is the case for homes A_1, A_3, A_{10}, B_2, B_3, B_4, and B_6, our framework cannot reduce the number of misdetections. This is also true when the number of legitimate operations is too small to cooperate with other agents, such as for homes B_7, B_8, B_9, B_{10}, and B_{11}. In this case, each agent has only a small amount of operations and few requests are stored as requests that are similar to the behavior of the corresponding users. As a result, the agents cannot cooperate with other agents via our platform. However, the misdetections in such homes are not significant because there are only a small number of operations.

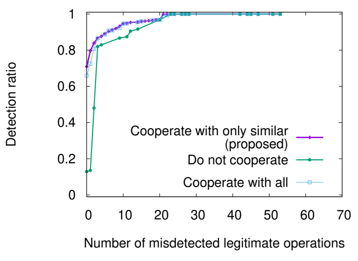
Nevertheless, the detection accuracy of our platform is similar to the method that used cooperation between all agents. This is because the sequences of the operations significantly aid in the detection of anomalous operations. The method using the sequences detects anomalous operations unless the current operations match the sequence of operations, including the operations of the other devices. For this evaluation, we added the anomalous operations on the cooking stoves, and the added operations were rarely included in the sequences that matched the legitimate sequence. As a result, should an agent cooperate with the agents of users who have different behaviors, such cooperation does not degrade the detection ratio.

Fig. 4.9 shows the heatmap of the similarity between the IDs of the requests stored by each agent. We calculated the similarity of the stored IDs by the percentages of IDs stored by the agents in the vertical axis that matched the IDs stored by the agents in the horizontal axis. For our platform, agents storing the same IDs in their ID database were defined as similar agents. From this figure, agents who corresponded to the same real home cooperated in our platform. That is, our platform achieved cooperation with agents with similar behaviors.

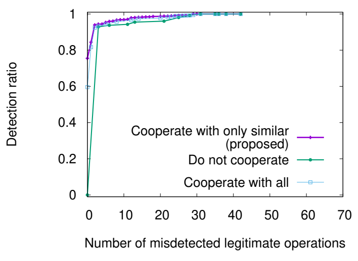
We also evaluated our framework when the anomaly detection method was only based on the condition of the home. Fig. 4.10 and 4.11 show the results for each agent, where the anomaly



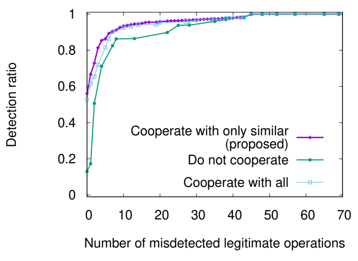
(a) A_1



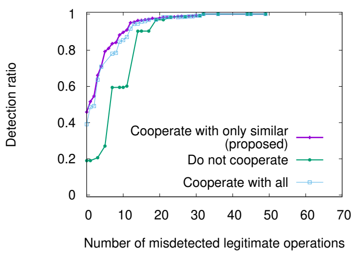
(b) A_2



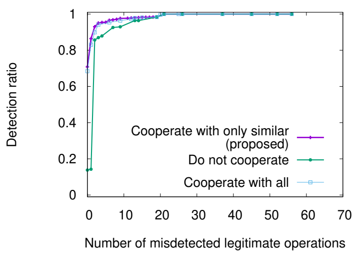
(c) A_3



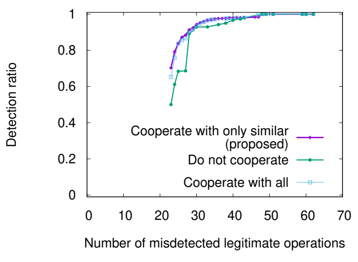
(d) A_4



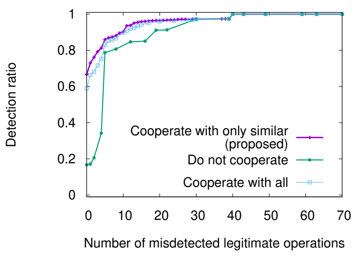
(e) A_5



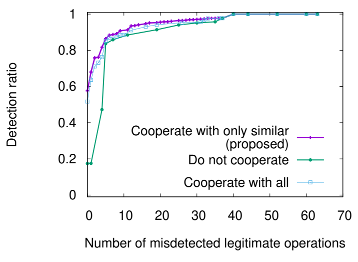
(f) A_6



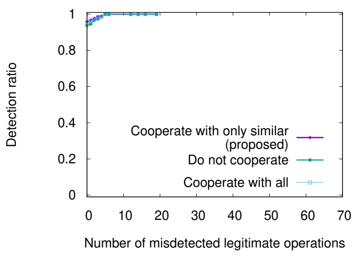
(g) A_7



(h) A_8

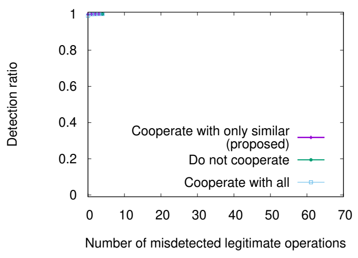


(i) A_9

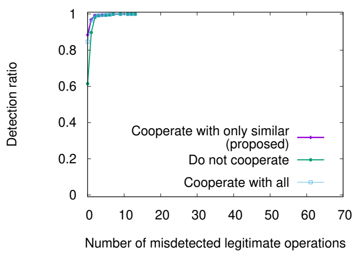


(j) A_{10}

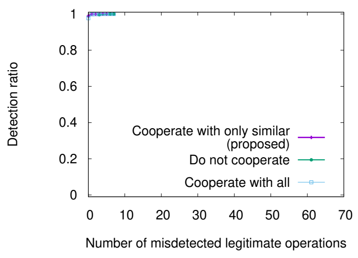
Figure 4.7: Detection results of each month in home A. Each agent performed the anomaly detection method using by considering both the condition of the home and the sequence of operations.



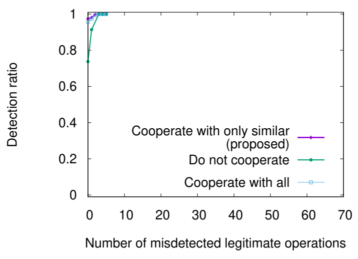
(a) B_1



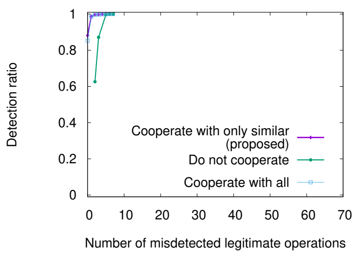
(b) B_2



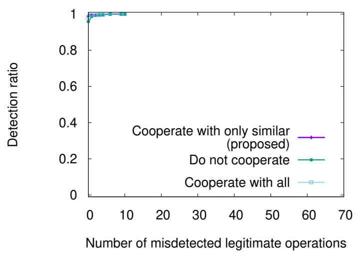
(c) B_3



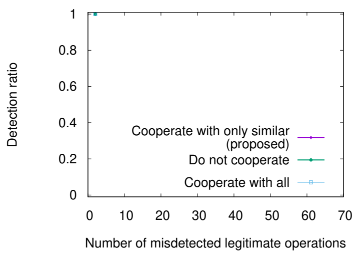
(d) B_4



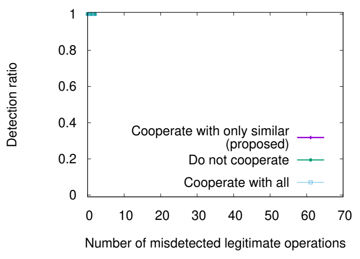
(e) B_5



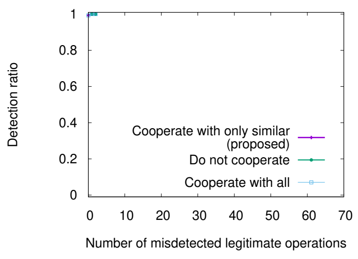
(f) B_6



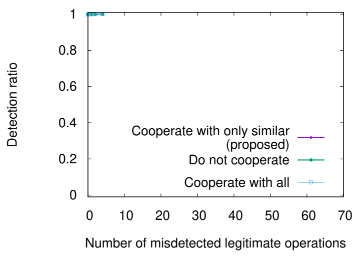
(g) B_7



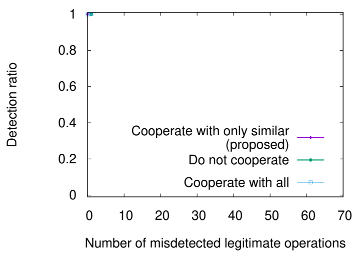
(h) B_8



(i) B_9



(j) B_{10}



(k) B_{11}

Figure 4.8: Detection results of each month in home B. Each agent performed the anomaly detection method using by considering both the condition of the home and the sequence of operations.

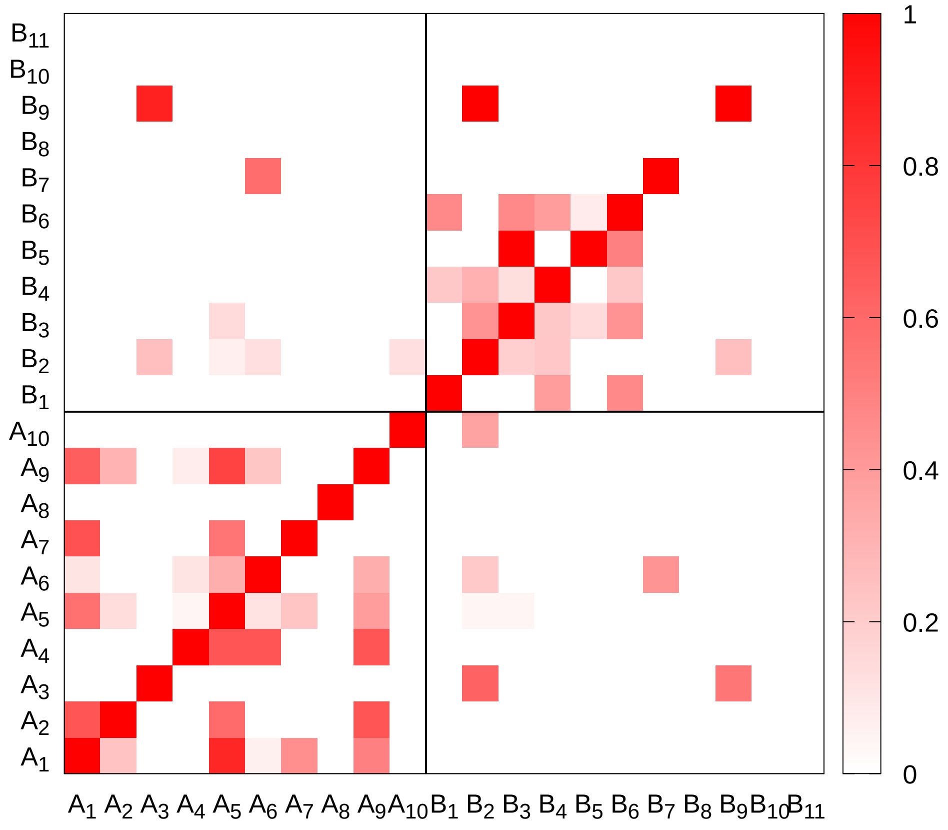


Figure 4.9: The heatmap of matched IDs of each agent; percentages of the ID database on the vertical axis that matched with the ID database of the user on the horizontal axis.

detection method is solely based on the condition of the home. We plotted the results in the same way as Fig. 4.7 and 4.8. These figures indicate that homes A_3, A_4, A_5, A_6, A_8, A_9, A_{10}, B_1, B_2, and B_5 achieve a smaller number of false negatives of anomalous operations than the method that has cooperation between all agents. When the parameters are set to achieve a misdetections number less than four, the detection ratio of the proposed method is 13.4% higher than that of the method that has cooperation between all agents in home A_4. This is because cooperation with agents of different behaviors degrades the accuracy of the detection. By cooperating with all agents, an agent uses information that does not match the behavior of the corresponding users. As a result, some attacks that are different from the behavior of the corresponding users but match the behavior of other users are not detected. However, in our framework, agents cooperate with only similar agents.

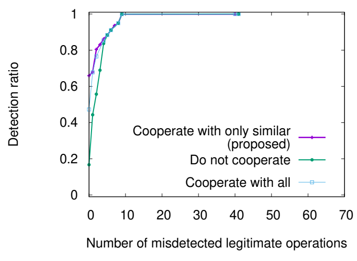
As a result, our framework avoids the degradation of the detection accuracy that is caused by using the information of users whose lifestyles are different.

Similar to Fig. 4.7 and 4.8, our method cannot reduce the misdetections for the case where the number of legitimate operations is too small to cooperate with other agents; such is the case for homes B_7, B_8, B_9, B_{10}, and B_{11}. However, if an agent has some legitimate operations and can calculate the similarities between agents, our framework achieves a smaller number of misdetections than the method that incorporates cooperation between all agents.

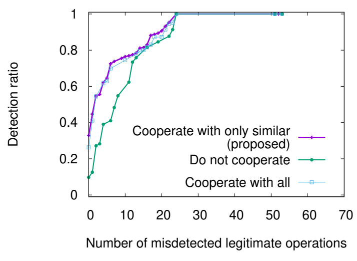
4.5 Conclusion and future work

In this chapter, we proposed a cooperation framework that utilized the dataset of similar users without sharing their private information to detect anomalous operations. In this platform, an agent was deployed at each home. The agent learned the behavior of the users and detected anomalous operations based on the learned behaviors. However, if the agent did not identify whether the current operation was legitimate or not, due to a lack of training data, it asked the other agents by sending a request. The agent informed the property of the users by attaching the IDs of the past requests that matched the behavior of the corresponding users. Then agents who also identified the past requests of the attached IDs as legitimate replied to it. By doing so, our framework enabled agents to cooperate with similar agents without sharing their private information.

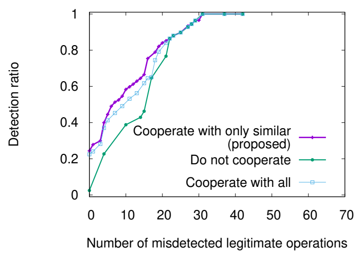
We evaluated our framework by using the dataset monitored at real homes. The results indicated that our framework reduced the number of false negatives of anomalous operations by cooperating between similar agents. We applied 21 homes to the simulation in this evaluation; however, when the number of homes increases, the detection result of our cooperation framework becomes accurate because the agents of users having similar behaviors increase. In particular, the importance of similarity judgment of agents becomes high because the number of agents of users having different lifestyles also increases. In addition, the more agents participate in the framework, the more types of behavior patterns are regarded as legitimate excessively. The parameter tuning can eliminate the unnecessary patterns. It is also considered that the framework scales up and the number of questions sent by the agents is too much for the others to answer. The hierarchization of the communication



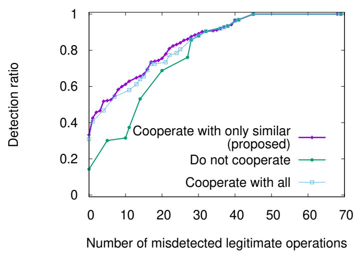
(a) A_1



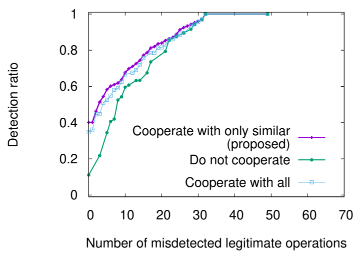
(b) A_2



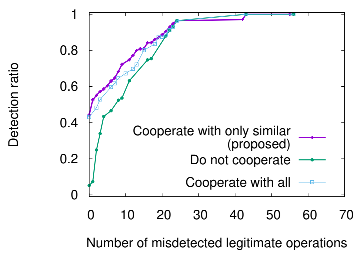
(c) A_3



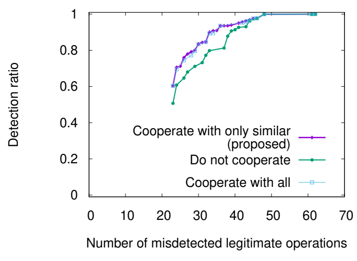
(d) A_4



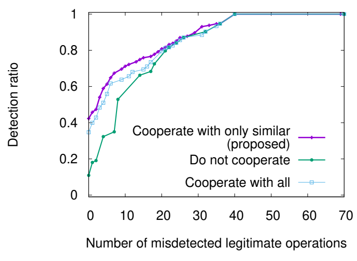
(e) A_5



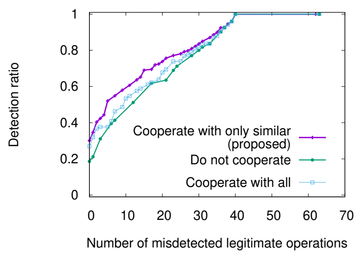
(f) A_6



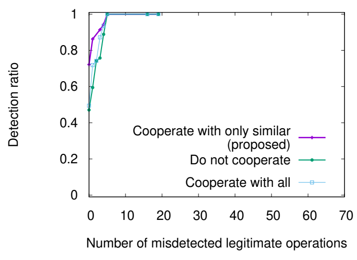
(g) A_7



(h) A_8

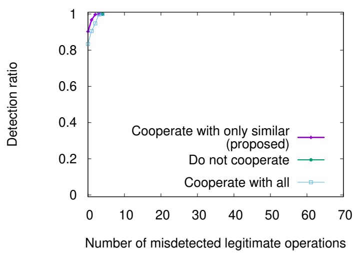


(i) A_9

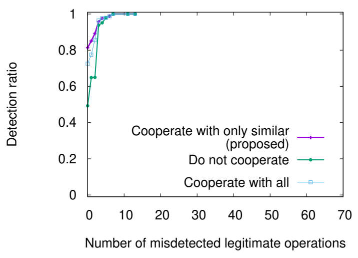


(j) A_{10}

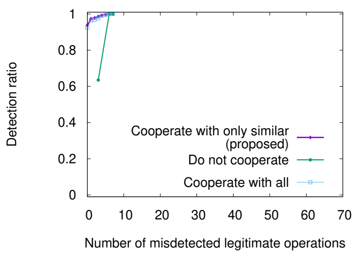
Figure 4.10: Detection results of each month in home A. Each agent performed the anomaly detection method using the condition of the home.



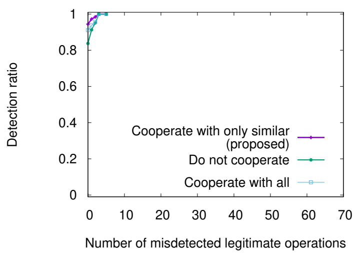
(a) B_1



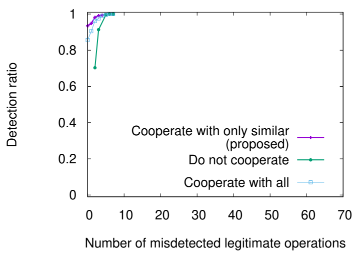
(b) B_2



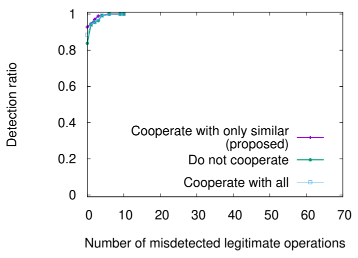
(c) B_3



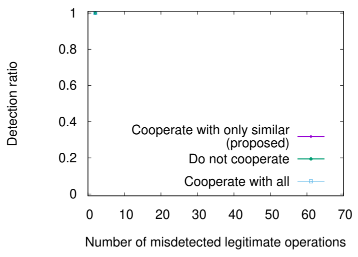
(d) B_4



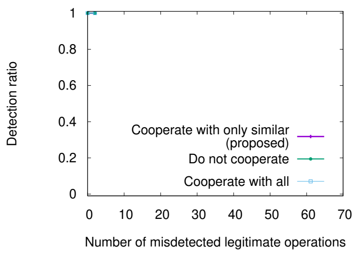
(e) B_5



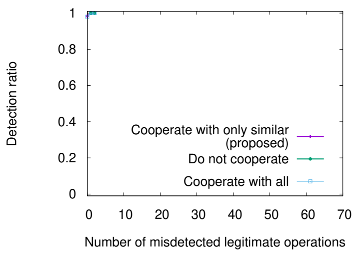
(f) B_6



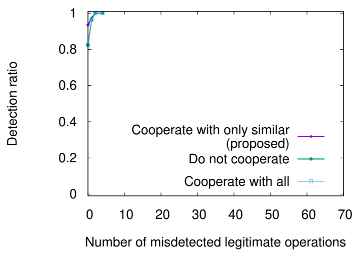
(g) B_7



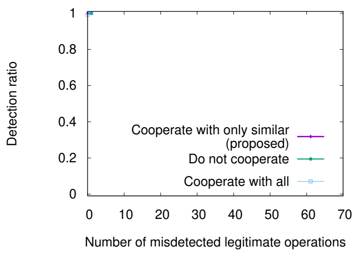
(h) B_8



(i) B_9



(j) B_{10}



(k) B_{11}

Figure 4.11: Detection results of each month in home B. Each agent performed the anomaly detection method using the condition of the home.

protocol, such as the IP address, is one of the ideas to solve the problem. The scalability of this framework is one of our future works.

We applied our framework to the detection of anomalous operations of home IoT devices. However, our framework can also be applied to other scenarios where cooperation between similar users is required; this can serve as future work. For example, by applying our framework to a system that recommends products to users, an agent recommends a product using the information on similar users without sharing their private information.

In this chapter, we assumed that all agents behaved correctly. However, we should consider the case where attackers join our framework to evaluate its robustness. The defense mechanism against such attacks targeting our framework is another future work.

Chapter 5

Improving Attack Tolerance of Privacy-Preserved Cooperation Framework

5.1 Introduction

Online service advertising, such as for Internet shopping, is tailored to user interests and preferences by analyzing usage data collected in a cloud environment [57, 58]. These services are considerably successful. For example, the Amazon recommender system accounts for more than 35% of all their purchases [59]. Smartwatches and other wearable devices make it possible to collect biometric user data; hence, services tailored to those variables are likely to become more diverse and widespread.

As data collection becomes more sophisticated and intuitive, personal privacy must be considered and protected. This concern is more crucial than ever with cloud-based big data becoming more prevalent [27]. Notably, information leakages caused by human errors are a significant risk [28]. Even if a small amount of data is compromised, more detailed information can be gleaned from it. For instance, attackers can accurately predict the gender of a user based on information about watched television programs [60].

Privacy-preserving data collection and analysis methods that collect the gradients of each trained model or noisy data and share a general model trained from these do exist [25, 26]. However, these methods cannot generate models that are tailored to the interests and preferences of users because they are highly generalized. For example, when we apply them to a model that learns the behaviors of users at home to detect anomalous operations of home Internet-of-things (IoT) devices [32], it is difficult to generate a model tailored to users preferences because the in-home behaviors of different people vary significantly.

Thus, we proposed a decentralized cooperation framework to utilize the data of similar users without sharing their private information [38] in Chapter 4. This privacy-preserved cooperation (PPC) framework installs an agent that learns user data. The agents train their own learning models and collaborate with other agents to utilize the data of similar users. When an agent does not have sufficient data, it anonymously sends a question to the other agents. The question can be answered using 1-bit information, such as “Yes” or “No,” based on the question. The questions are identified using question IDs. When the agent sends a question, the agent attaches multiple question IDs that, for example, are answered (voted) with “Yes.” Only agents who vote “Yes” to the same questions vote on the next one. The questioner summarizes the votes and uses the results. By doing so, user data with similar characteristics can be accumulated without revealing personally identifiable information. When an agent does not want to vote on a question, voting can easily be avoided. In doing so, they can maintain higher levels of control over user privacy.

As an example of using this PPC framework to verify whether the information of similar users can be obtained, we applied an anomaly detection method to detect the anomalous operations of home IoT devices [32]. We simulated this anomaly detection method using data obtained from actual homes and determined that the detection accuracy was improved by up to 50.5%. This is because the PPC framework can judge the similarity using the question IDs voted “Yes” and data of similar users who had same behaviors for detection. We found that it is possible to cooperate with user agents having similar characteristics while protecting the identities of the sender. However, the PPC framework assumes that all agents work properly according to the protocol and does not consider cases in which an attacker or a malicious third party participates. Notably, an attacker can spread false information and create a negative impact.

In this Chapter, we propose a countermeasure that ensures the functions of the PPC framework

work properly even when a malicious third party participates. We first analyze the risks of the scenario. Specifically, we identify the actions that can be taken from each position in the PPC framework (i.e., questioner, respondent, or viewer—who does not vote) to make it dysfunctional and analyze the impact of each action. In particular, we target attacks that cause agents to make incorrect decisions within the framework. We identify the problem of attackers sending fake questions and votes to mislead the similarity judgment between agents and to falsify the summarized results. When an attacker sends fake questions, most are not answered because agents judge the similarity from the ID attached to the question and answer only questions from similar agents. However, if an attacker sends many fake questions and some of them accidentally include the question IDs that a legitimate agent stored, the agent may store the IDs of the fake questions. The fake question ID stored by a legitimate agent may have a negative impact on the similarity assessment. As the number of fake question IDs stored increases, the negative impact grows. Furthermore, when an attacker sends a fake vote, it may affect the summarized results. In contrast to fake questions, most fake votes are accepted by legitimate agents. This is because the question similarity is judged; however, the vote is not judged and accepted until the number of votes collected exceeds a threshold. Hence, these attacks depend on the number of fake votes.

Currently, there are methods that prevent the mass posting of false information, such as by limiting the bandwidth of cellular networks [61] or preventing fake reviews on Amazon [62, 63]. These methods use sender certifications to restrict input and confirm the legitimacy of users based on past participation. However, these authentication methods are unsuitable for the PPC framework because the sender's information needs to be hidden. Another method that does not use personal identification is hashcash [64], a denial-of-service (DoS) countermeasure. Hashcash requires an arbitrary value that is added to the transmitting data. The hashed value of the data with added value is less than a certain number of digits. The mechanism requires computation time to search for this arbitrary value and makes it difficult to transmit false data many times. This method can be used to limit the number of fake questions and votes. However, many fake vote attack problems cannot be solved solely by limiting the number of votes using hashcash.

Hence, in this Chapter, we propose a countermeasure to reduce the negative effects of false questions and votes by an attacker while maintaining the advantages of the PPC framework, in which agents cooperate anonymously with similar agents. First, we leverage the hashcash method to reduce the number of questions and votes because the negative impact of fake questions is suppressed by limiting the number of questions. Adding to the limitation of the number of votes, agents summarize only the votes sent from a similar agent. When an agent votes to answer a question, it attaches to the vote the IDs of the questions to which it answered “Yes” in the past. The questioner agent receives the vote and checks the number of matched IDs between the attached IDs to the vote and the stored IDs of the questions to which the questioner answered “Yes” in the past. When the number of matched IDs exceeds a threshold, the vote is regarded as a vote from a similar agent and is summarized. The checking process makes it difficult for attackers to send fake votes that are summarized because the attacker does not know the IDs of the questions to which the questioner has answered “Yes.” This allows the questioner and respondent to judge the similarity with each other via the IDs attached to the question and vote. These IDs should have a time limit on their use; they should not be able to use arbitrary values that are bound to past question IDs and vote data that have been used in the past. However, they should be able to include random values in the questions and perform hash calculations. In this way, we can restrict the preparation of many fake questions and votes in advance. Furthermore, by attaching a public key to the question, the respondent encrypts and sends back a vote, preventing the attacker from referring to and sending the legitimate agent’s vote. This PPC with Countermeasure (PPCwC) framework makes it possible to reduce the negative effects of fake questions and votes without losing the PPC framework’s advantageous features using similar users’ anonymous data.

To evaluate the framework, we used numerical examples to confirm that even if an attacker sends many fake questions and votes, this does not affect the anonymous similarity judgment, and the summarized results are not falsified. Furthermore, we confirmed that these calculations can be executed.

The remainder of this article is organized as follows. We introduce the PPC framework [32] and analyze its risks in Section 5.2. In Section 5.3, we present the proposed countermeasure that prevents the effects of false questions and votes while cooperating with similar agents. Then, we discuss how our approach avoids attack risks in Section 5.4. Finally, we conclude and discuss future work in Section 5.5.

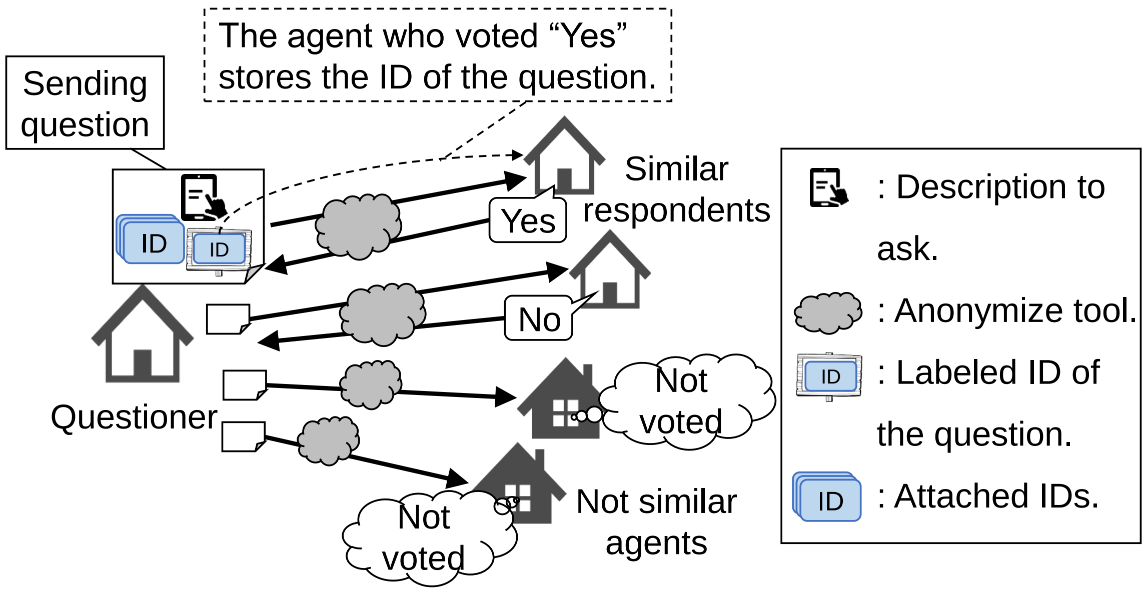


Figure 5.1: Overview of the privacy-preserved cooperation (PPC) framework in Chapter 4.

5.2 Overview and Risk Analysis of the PPC Framework

5.2.1 Overview of the PPC Framework

An overview of the PPC framework is given in Fig. 5.1. In this framework, agents are installed to train their learning model with user data in a home environment of routers and smartphones. If an agent's data are insufficient, the agent deals with the shortage by cooperating with other agents. The agent anonymously sends a question about the needed information; the question can be answered with a "Yes" or a "No." Then, the responding agents vote anonymously. Each question has an identifier. An agent who answered "Yes" to a question stores the ID of the question. This stored question ID is used as a key to determine similarity. The questioner selects question IDs from the stored database and sends them anonymously to other agents. Anonymization is achieved using Tor [55] (the onion router), which is an open source privacy network that permits users to transmit data while hiding the sender information, i.e., IP address. The agents that receive the question judge similarity based on whether the question was sent from a similar agent, which is based on whether the number of matched IDs between the attached ID and its own stored ID exceeds a threshold. Hence, agents who have answered "Yes" to the same questions in the past may collaborate with

Table 5.1: Actions and possible effects of an attacker in the PPC framework when the attacker is a questioner, respondent, or viewer according to each function.

Function	Viewer	Questioner	Respondents
Posting questions	Watching questions (Obtaining information)	Sending fake questions (Affecting similarity judgment)	—
Voting on questions	Watching votes (Obtaining information)	—	Voting false answers (Falsifying summarized results)
Summarizing votes	Watching votes on questions (Obtaining information)	Obtaining votes on fake questions (Obtaining information)	—

each other. If they find the question was sent from a similar agent, they select “Yes” or “No” to the question and vote on them anonymously. The respondents who vote “Yes” store the ID of the question and use it for similarity judgment in the future. The questioner then summarizes the votes and receives the results of the sent question.

In this framework, users control their privacy by choosing when not to vote on questions that they do not wish to answer by adding noise to the questions or by sending dummy questions.

5.2.2 Risk Analysis of the PPC Framework

We analyze the negative impact on the framework when a malicious third party participates in the PPC framework. In this framework, there are three roles: questioner, respondent, and viewer (who does not vote). It has three functions for agents: sending questions, voting on questions, and summarizing votes. Table 5.1 shows the possible actions that an attacker can take by using each function in each role of the PPC framework and their impact.

These effects can be divided into three categories: getting information, affecting similarity judgments, and falsifying voting results. Because agents interact anonymously and openly with each other, information can be obtained by observing their interactions, but attackers cannot know the sender. Here, when the questioner attaches a public encryption key to the question and encrypts the votes, the information in the votes is unavailable. Therefore, the information that an attacker can obtain is limited to statistical information about the question, including descriptions and question IDs. Furthermore, agents can add noise to these statistics by sending dummy questions.

An attacker can adversely affect similarity judgments and voting results by sending fake questions and fake votes as questioners and respondents. This may cause agents to cooperate with inappropriate agents, or it may pollute agents' judgments; the questioner agent may make bad decisions based on falsified votes. Thus, the PPC framework cannot work properly when the function of the framework is misused. We analyze the two impacts of fake questions and fake votes in the following sub-subsections.

Attack by Questioner

An attacker may act as a questioner and send fake questions, which may affect the similarity judgment among agents. If an agent receives a fake question and votes "Yes," it stores the ID of the fake question in its database. Because each agent judges similarity using the stored IDs of questions answered with "Yes" in the past, the similarity between agents may be corrupted. For example, by sending questions to which many agents would vote "Yes," agents that are not inherently similar could store the same IDs and receive votes from dissimilar agents. By contrast, sending questions to which only some agents vote "Yes" may cause agents that are inherently similar to each other to store different IDs, which may isolate them and make it harder for them to obtain answers from the framework.

A small number of fake questions is considered to have little impact on similarity judgment because the ID database of each contains many question IDs from many legitimate agents. However, if many false questions are sent, a similarity judgment may be impacted.

If the attacker wants to store the IDs of fake questions in the ID database of other agents, the attacker must attach a greater number of question IDs than a threshold. Therefore, the success rate of the fake question attack depends on how much information about the question ID is stored in the agent database.

Attack by Respondent

An attacker may act as a respondent and cast a fake vote, which could falsify the summarized results of all votes. If an attacker provides a false vote to a question, the questioner summarizes the fake vote accordingly. Hence, the questioner may receive incorrect results from the framework and may make incorrect decisions over time. For example, we applied an anomaly detection model to the framework, which learns a behavioral model in the home environment and detects anomalous operations of home IoT devices that deviate from the model. The framework was asked whether the operation was legitimate or anomalous. When an agent receives false votes from the framework, an anomalous operation may be successful.

A small number of false votes would have little effect on the summarized results because many votes by legitimate agents would be received. However, if many false votes are made, the summarized results may change.

Because votes are anonymous in the PPC framework, there is no way to distinguish between fake votes from attackers and votes from legitimate agents. Thus, all votes are summarized until the number of votes is collected. Therefore, the countermeasure of limiting the number of votes is insufficient.

5.3 PPCwC: Suppressing a Large Number of Fake Questions and Votes While Cooperating with Similar Agents Anonymously

We propose a cooperation framework that can suppress the effects of a large number of fake questions and votes while maintaining the advantages of the PPC framework through anonymous cooperation among similar agents.

5.3.1 Key Concept

We check the similarity between the questioner and the respondent. Specifically, when they send the vote on a question, the respondent attaches its stored IDs of the questions to which it voted “Yes” in the past. By doing so, the questioner can judge whether a vote is sent from a similar agent.

We also limit the number of answers and votes using the hashcash [64] algorithm.

To limit the number of fake questions and votes, the hashcash [64] method requires arbitrary values that are added to the sent data; the hashed value of the added data satisfies the conditions for questions and votes. This technique is used in blockchain [65] to prevent a large number of fake transactions. Similar to the terminology employed in blockchain, we call the arbitrary value a "nonce." The search for a nonce requires hashing computations. Because the hash value is not inferable from the original value, it is necessary to perform multiple hash calculations while setting different values for the nonce. This requires computation time and limits the number of questions and votes. If the nonce value is incorrect, the sent data are dropped. By setting the number of answers and votes that do not restrict the questions and votes of legitimate agents and limiting the fake questions and votes by attackers, the framework suppresses the effects of false questions. However, even if the number of votes is limited, false votes can still permeate the network, as in cases where the number of attackers is larger than the number of legitimate respondents.

Therefore, along with the limitation of the number of questions and votes, the respondent must attach their stored IDs to their vote, and the questioner must check whether the vote was sent from a similar agent. The questioner checks whether the IDs attached to the vote matches their own ID database according to a threshold number and accepts the vote if it satisfies the condition of coming from a similar agent. This makes it difficult for attackers to affect the summarized results by sending fake votes because the attacker must include the IDs stored by the questioner.

Next, we introduce a nonce generated when attaching IDs. The question IDs have an expiration date, and questions and votes sent in the past with the same nonce value and question ID combination are discarded. Thus, it is possible to prevent a large number of combinations of nonce values and question IDs from being generated in advance, and questions and votes are prevented from being sent many times to the same combination of nonce values and IDs. Furthermore, the questioner attaches a random value to the question based on a nonce search for voting. This prevents the preparation of a large number of false votes in advance.

Furthermore, the questioner generates a public key for encryption and a secret key for decryption and attaches the public key to the question. Then, the respondents encrypt the votes, and the questioner decrypts them. Thus, the viewer does not observe the vote information while limiting the viewer's information. If an attacker sends a fake question and receives votes on the question, the attacker can obtain the vote information, which includes the attached ID information. However, because the number of questions is limited, collectable information is also limited.

Moreover, IDs that can be attached to a vote must be within d days of the date of the question, and IDs that can be attached to a question must be older than d days and younger than $2d$ days. Even if the attacker obtains information about questions as a viewer and finds the IDs that are often attached to the same questions, the analyzed information between IDs cannot be used to falsify votes.

These ideas can suppress the negative impact of a large number of false questions and answers. Because the verification of the nonce value requires a single hash calculation, the computational load on the legitimate agent receiving the question and vote is minimal and is not a direct factor in creating a DoS attack. The number of votes, the content of the vote, and the summarized number of votes will vary depending on the combination of ID selections, and this framework is assumed to depend on this combination of IDs.

5.3.2 Procedures

The communication procedure of the proposed method is shown in Figure 5.2.

Posting Questions

First, the questioner prepares to ask the question by generating a public encryption key and a private decryption key while searching for the nonce value, $nonce_q$. The public key is attached to the question, and the private key is used to decrypt votes. To search for the nonce, the questioner selects n_q IDs, ID^q_1, ID^q_2, …, $ID^q_{n_q}$ from the ID database of the questioner, $\boldsymbol{ID^q}$. Then, it searches for a nonce value, $nonce_q$, that satisfies (5.1).

$$\text{Hash}(ID^q_1||ID^q_2||\ldots||ID^q_{n_q}||nonce_q) < T_q. \tag{5.1}$$

Here, T_q is the threshold of the hash value; when the value is small, the time required for the hash search increases. Hash denotes a hash function. By searching for the nonce value in advance, agents

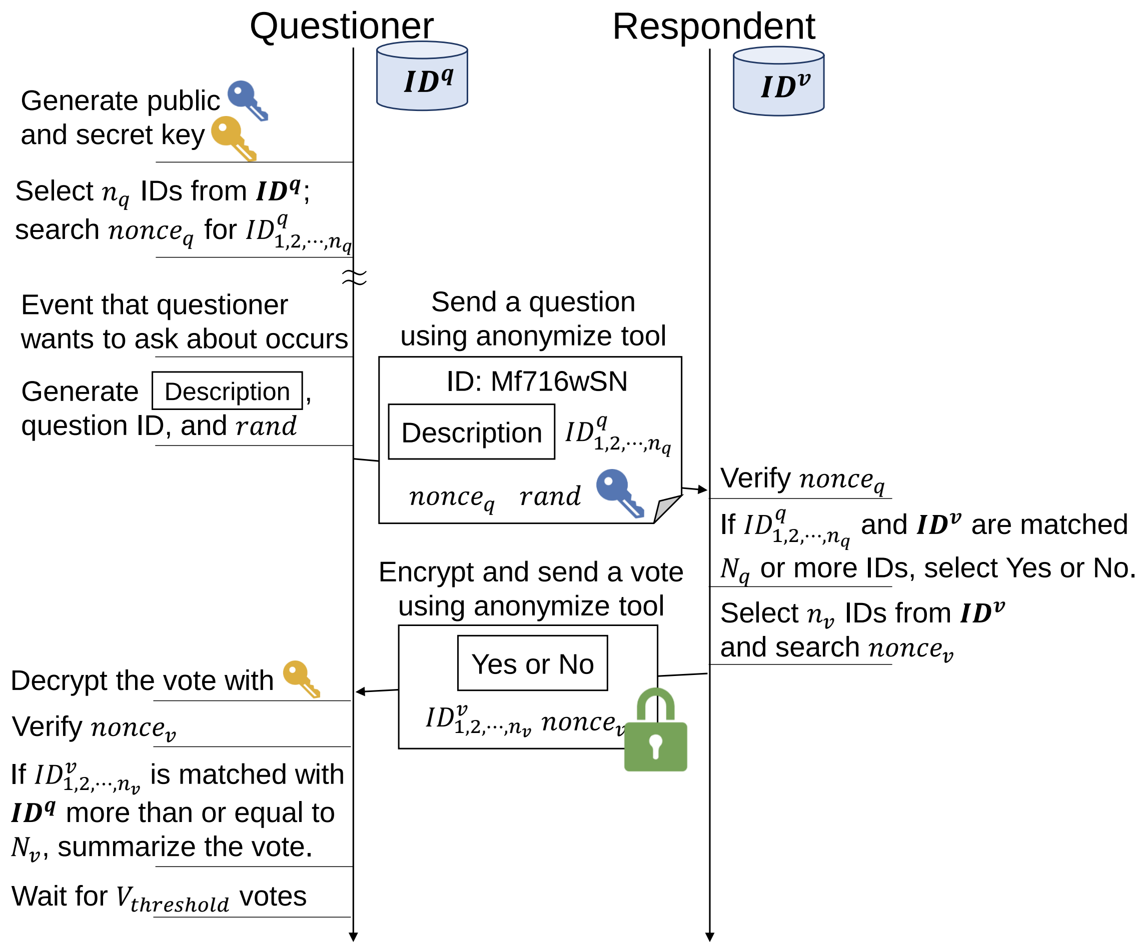

Questioner
Respondent
ID^q
ID^v
Generate public and secret key
Select n_q IDs from ID^q; search $nonce_q$ for $ID^q_{1,2,\cdots,n_q}$
Event that questioner wants to ask about occurs
Generate Description, question ID, and $rand$
Send a question using anonymize tool
ID: Mf716wSN
Description $ID^q_{1,2,\cdots,n_q}$
$nonce_q$ $rand$
Verify $nonce_q$
If $ID^q_{1,2,\cdots,n_q}$ and ID^v are matched N_q or more IDs, select Yes or No.
Select n_v IDs from ID^v and search $nonce_v$
Encrypt and send a vote using anonymize tool
Yes or No
$ID^v_{1,2,\cdots,n_v}$ $nonce_v$
Decrypt the vote with
Verify $nonce_v$
If $ID^v_{1,2,\cdots,n_v}$ is matched with ID^q more than or equal to N_v, summarize the vote.
Wait for $V_{threshold}$ votes

Figure 5.2: Overview of the privacy-preserved cooperation with countermeasure (PPCwC) framework.

can ask a question at any time. To prevent the same combination of nonce and question IDs from occurring more than twice, the questioner checks that the same combination is not used. It also checks to see if the IDs have expired. When the combination of nonce and IDs is used or expired, the agent disposes of the combination and searches for a new nonce.

Next, when an event occurs for which an agent has a question for other agents, the questioner generates the asking description. Then, the questioner selects the prepared question IDs, $nonce_q$, a public decryption key, a random string, *rand*, and a new question ID for the sending question. The asking description can be answered with either "Yes" or "No." The new question ID is set as a random character string that should not overlap with any ID within the usage period. The random string, *rand*, is used to generate the nonce for the vote, $nonce_v$, to prevent the nonce from searching for votes in advance. Then, the questions are sent anonymously.

Furthermore, agents can send dummy questions to prevent the collection of question information, such as IDs and question contents. This scatters the information about the combination of question IDs that are often attached to the same question. Dummy questions are sent using the same process as regular questions.

Generating Votes

The respondent who receives the question verifies that the nonce, $nonce_q$, attached to the question and $ID^q_{1,2,\ldots,n_q}$ satisfy (5.1). It also checks that the expiration date of the attached IDs has not expired and that the same nonce and ID have not been used in the past. If an inappropriate nonce or ID is used, the question is discarded.

After verifying the nonce, $nonce_q$, the respondent decides whether the question is sent from a similar agent by comparing the attached IDs and the stored ID database of the respondent. Specifically, the respondent checks whether the number of IDs that match in $ID^q_{1,2,\ldots,n_q}$ and $\boldsymbol{ID^v}$ is more than N_q, that is, whether (5.2) is satisfied.

$$\text{Num}(\{ID^q_{1,2,\ldots,n_q}\} \cap \boldsymbol{ID^v}) \geq N_q, \tag{5.2}$$

where $\text{Num}(e)$ is a function that returns the number of IDs that satisfies e.

When the respondent judges the question is sent from a similar agent, it selects “Yes” or “No” for the question. If the respondent does not want to vote on the question, the agent can choose not to answer it. When the respondent selects “Yes,” the respondent stores the ID of the question into its ID database, $\boldsymbol{ID^v}$, which is used for similarity judgments in the future.

To send a vote, the respondent generates a nonce, $nonce_v$, and attaches it to the vote. The respondent selects n_v IDs from the ID database and searches for the nonce with *rand*that is attached to the question. Specifically, the respondent searches for the $nonce_v$that satisfies (5.3).

$$\text{Hash}(ID_1^v||ID_2^v||\dots||ID_{n_v}^v||rand||nonce_v) < T_v. \tag{5.3}$$

This can prevent a large number of votes from being sent because multiple calculations are required to find the nonce, which requires considerable calculation time. By adding *rand*to the hash calculation, we prevent the attackers from searching for a combination of IDs and nonces before the question is asked. After searching for the nonce, the respondent attaches “Yes” or “No” and an ID with a nonce to the vote.

The respondent then encrypts the vote using the public encryption key attached to the question and votes anonymously. Encryption prevents the viewer from watching the vote, and attackers cannot collect the vote information. If a respondent votes on a fake question by an attacker, the attacker will get information about the vote and its attached IDs. However, it is not possible to obtain information about who sent the votes. Furthermore, the number of questions is limited; thus, the amount of information obtained is limited.

Summarizing Votes

The questioner who receives the votes decrypts them with the secret key and verifies the $nonce_v$attached to the vote using (5.3). The questioner then checks whether the attached ID is within the expiration date, and the same combination of nonce and IDs is not used. The votes, including inappropriate nonces and IDs, are discarded. This prevents the same combination of nonces and IDs from being used multiple times.

The questioner checks whether the vote is sent from a similar agent and checks to see if the IDs, $ID^v_{1,2,\dots,n_v}$, attached to the vote matches the questioner’s ID database, $\boldsymbol{ID^q}$, greater than or equal to N_v. That is, when the IDs satisfy (5.4), the questioner considers the vote to be sent from a similar agent.

$$\text{Num}(\{ID^v_{1,2,\dots,n_v}\} \cap \boldsymbol{ID^q}) \geq N_v, \tag{5.4}$$

where N_v is the threshold. By attaching IDs to the votes, summarized votes are selected, and the attacker cannot send the summarized vote easily. This is because the attacker must randomly select n_v IDs from the questions stored and match N_v or more with the $\boldsymbol{ID^q}$.

When the questioner receives more than or equal to $V_{threshold}$ votes, the questioner stops receiving the votes, summarizes them, and decides the result based on the number of "Yes" and "No" votes. After a certain number of votes are received, the questioner halts the operation; this prevents the receipt of votes sent by attackers many times. The decision method is set as appropriate for the application that is applied to the PPCwC framework.

5.4 Discussion

Numerical examples confirm that the proposed method prevents the negative effects of large numbers of questions and votes, and that these computational times are within the executable time. In this discussion, the agents used an anomaly detection method for home IoT devices that learns the behavior of users in each home for detection based on the usage of the IoT devices [32]. We assumed an environment in which agents who learn the behavior of users in the home cooperate with each other to compensate for the lack of data in each home in a city. Specifically, we assumed that the city has $R_l = 40,000$ homes. We also assumed that each home operates its home IoT devices 100 times per day, on average. On average, the agent wants to ask about 1% of these operations. Thus, $40,000$ questions are sent to the framework on average per day. We set the validity period of the ID to 7 days. We list the assumptions, representations, and assumed values in this discussion in Table 5.2.

5.4.1 Impact of Attacks by Questioner

We confirmed that the PPCwC framework obviates the negative impact of a large number of false questions based on similarity judgments and limiting the number of questions. If a large number of false questions are sent, and false question IDs are stored in the respondent's ID database, the similarity judgment may be adversely affected. In this evaluation, we compared how the probability that agent B votes on the question by agent A changes before and after the attack in the PPCwC framework and the PPC framework. We evaluated two cases: an imitating attack in which an

Table 5.2: Assumed city environment and representations for this discussion.

Explanation	Representation	Assumed value
Number of questions flowing through the framework per day	R_l	40,000
Expiration date of question IDs	d	7
Number of IDs attached to the question	n_q	100
Number of IDs attached to the vote	n_v	100
Threshold of matched IDs of the attached IDs of the question and the respondent stored	N_q	30
Threshold of matched IDs of the attached IDs of the vote and the questioner stored	N_v	30

attacker misleads by making two dissimilar agents similar, and an isolating attack that misleads by making two similar agents dissimilar.

We define the storing number of IDs of legitimate questions in a day as Q_l. When a question is sent from a questioner of the user having similar characteristics as a respondent, and the respondent votes "Yes," the ID of the question is stored. Here, we set the ratio that an agent judges the legitimate questions to be sent from similar users' agent and votes to the number of all legitimate questions in a day, R_l, as r_s. The probability of voting "Yes" to the similar questions that are judged to be sent from agents of users having similar characteristics is denoted by p_y. We can denote Q_l in (5.5).

$$Q_l(r_s) = p_y r_s R_l. \tag{5.5}$$

Similarly, we define the number of stored IDs of questions sent by the attacker for a day as Q_a. Here, the probability that a respondent voted on a fake question depends on how the attacker can limit the IDs that the respondent stores for all IDs. If the attacker can limit the IDs held by the respondent agent to $1/c$ of the all IDs, the probability that the respondent votes on the fake question by the attacker is denoted as cr_s. This is because the fake question is c times more likely to be accepted by the respondent. We assume that the number of questions an attacker can generate per day is R_a and the probability that a fake question is voted "Yes" is p_a; Q_a is represented by (5.6).

$$Q_a(M_q, r_s) = cp_a r_s R_a(M_q), \tag{5.6}$$

where R_a is the number of questions that an attacker can generate in a day. If an attacker can search one nonce for M_q s on average, R_a is expressed by (5.7).

$$R_a(M_q) = \frac{86400}{M_q}, \tag{5.7}$$

where 86400 is the seconds in a day.

We also define the number of IDs that the questioner, Agent A, and respondent, Agent B, store in common in a day. The number of legitimate questions' IDs that Agents A and B commonly store for a day, C_l, is calculated by the ratio of the number of question IDs that Agents A and B commonly store to the number of all legitimate questions, x_l. In addition to (5.5), we calculate using (5.8).

$$C_l(x_l) = p_y x_l R_l, \tag{5.8}$$

where x_l takes a larger value if Agents A and B are similar, and a smaller value if they are not. The number of false questions' IDs that Agents A and B store in common in a day, C_a, is denoted by (5.9), where x_a is the probability that Agents A and B store IDs of false questions in common to the total number of false questions.

$$C_a(M_q, x_a) = c p_a x_a R_a(M_q), \tag{5.9}$$

where x_a is assumed to be large when an attacker performs an imitating attack and small when an attacker performs an isolating attack.

The number of IDs, S, that an agent holds after more than d days is denoted by (5.10):

$$S(M_q, r_s) = d(Q_l(r_s) + Q_a(M_q, r_s)). \tag{5.10}$$

For the number of stored IDs, S_{AB}, that both Agents A and B hold, we use (5.11).

$$S_{AB}(M_q, x_l, x_a) = d(C_l(x_l) + C_a(M_q, x_a). \tag{5.11}$$

Here, we define two similarities of Agent B from the perspective of Agent A. We first set the similarity for legitimate questions between Agents A and B as X_l ($0 \leq X_l \leq 1$), where X_l represents the ratio of the number of legitimate question IDs commonly stored by Agent B to the number of legitimate question IDs stored by Agent A. X_l is close to 1 when Agents A and B are similar and close to 0 when they are not. We also set the similarity of the attacker's questions from Agents A and B as X_a ($0 \leq X_a \leq 1$), where X_a is the ratio of the number of the attacker's question IDs stored by Agent B to the number of the attacker's question IDs stored by Agent A. When an attacker tries the imitating attack, it generates fake questions for acceptance by both Agents A and B; when it tries the isolating attack, it makes the fake questions for acceptance by either Agent A or B. Hence, X_a is close to 1 when the attacker attempts the imitating attack, and it is close to 0 when the attacker attempts the isolating attack.

In this case, the probability, $P_{receive}$, that Agent B votes on Agent A's question and Agent B's vote is accepted by the Agent A is calculated by the probability, $P_{question}$, that Agent B votes on Agent A's question, and the probability, P_{vote}, that Agent A accepts Agent B's vote. The ratio of the number of legitimate questions that Agents A and B vote on the number of legitimate questions is $r_{\{s,A\}}$ and $r_{\{s,B\}}$, respectively, and $P_{receive}$ is represented by (5.12).

$$P_{receive} = \begin{cases} P_{question} P_{vote}, & \text{otherwise,} \\ Null, & X_l r_{\{s,A\}} > r_{\{s,B\}} \vee X_a r_{\{s,A\}} > r_{\{s,B\}}, \end{cases} \tag{5.12}$$

where $X_l r_{\{s,A\}}$ is the ratio that Agents A and B voted on legitimate questions to all legitimate questions; $X_a r_{\{s,A\}}$ is the ratio that Agents A and B voted on fake questions to all fake questions. There is no case in which the number of questions answered by Agents A and B in common exceeds the number of questions answered by Agent B; thus, we can set this case as *Null*. $P_{question}$ and P_{vote} are the probabilities of selecting n_q and n_v IDs and more or equal question IDs than a threshold N_q and N_v attached to the question and voting from the IDs held by the questioner and respondent, respectively. These are expressed by (5.13) and (5.14).

$$
\begin{aligned}
P_{question} = \sum_{i=N_q}^{n_q} & \binom{S_{AB}(M_q, X_l r_{\{s,A\}}, X_a r_{\{s,A\}})}{i} \\
& \binom{S(M_q, r_{\{s,A\}}) - S_{AB}(M_q, X_l r_{\{s,A\}}, X_a r_{\{s,A\}})}{n_q - i} \\
& / \binom{S(M_q, r_{\{s,A\}})}{n_q}
\end{aligned} \tag{5.13}
$$

$$
\begin{aligned}
P_{vote} = \sum_{j=N_v}^{n_v} & \binom{S_{AB}(M_q, X_l r_{\{s,A\}}, X_a r_{\{s,A\}})}{j} \\
& \binom{S(M_q, r_{\{s,B\}}) - S_{AB}(M_q, X_l r_{\{s,A\}}, X_a r_{\{s,A\}})}{n_v - j} \\
& / \binom{S(M_q, r_{\{s,B\}})}{n_v}
\end{aligned} \tag{5.14}
$$

We list the representations of these equations of Subsection 5.4.1 in Table 5.3.

Impact of the Ratio of Agent Votes, $r_{\{s,A\}}$ and $r_{\{s,B\}}$, and the Average Time of Nonce Searching, M_q

In the case of imitating and isolating attacks, we discuss the probability that Agent B's vote is accepted for Agent A's question, $P_{receive}$, changed by the ratio of Agents A and B votes on the question, $r_{\{s,A\}}$ and $r_{\{s,B\}}$, and the average time of nonce searching, M_q s. We compare the change in $P_{receive}$ using the proposed method with the case in which there are no attacks. The no-attack case is set as $Q_a = 0$ and $C_a = 0$. We also compare $P_{receive}$ of the PPC framework. In the PPC framework, we assume that an attacker sends one fake question per second, $M_q = 1$; thus, it sends 86,400 fake questions per day, as shown in (5.7).

In the case of an imitating attack, we show the difference in the no-attack case, as shown in Fig. 5.3). In this case, the similarity of Agent B from the perspective of Agent A is set to $X_l = 0.2$, and the attacker's target of the imitating attack is $X_a = 0.9$. The case in which $P_{receive}$ is $Null$ is described in white. Using PPCWC and setting M_q to an appropriate value, as shown in Fig. 5.3d, the value of $P_{receive}$ is almost the same as that in the case without an attack. By contrast, in the PPC

Table 5.3: Representations in Section 5.4.1.

Explanation	Representation and value
Number of stored IDs of legitimate questions in a day	Q_l
Number of stored IDs of fake questions in a day	Q_a
Number of questions an attacker can generate in a day	R_a
Number of stored IDs of legitimate questions that both Agents A and B store in common in a day	C_l
Number of stored IDs of fake questions that both Agents A and B store in common in a day	C_a
Number of IDs that an agent holds after more than d days	S
Number of IDs that both Agents A and B store in common after more than d days	S_{AB}
Ratio of the number of legitimate questions that an agent votes to all legitimate questions	r_s
Probability of voting "Yes" on the legitimate questions that includes same IDs more than or equal to the threshold as IDs the respondent stored	$p_y = 0.9$
Percentage of IDs that an attacker assumes that Agents A and B stored to all IDs	$1/c$
Probability that a respondent votes "Yes" on a fake question that includes same IDs more than or equal to the threshold as IDs the respondent stored	$p_a = 1.0$
Averaged seconds that the attacker can search the nonce for a question	M_q
Ratio of the number of legitimate question IDs that Agents A and B commonly store to the number of all legitimate questions	x_l
Probability of the number of fake questions that Agents A and B store in common to the total number of fake questions	x_a
Ratio of the number of legitimate question IDs that Agent B commonly stores to the number of legitimate question IDs that Agent A stores	X_l
Ratio of the number of the attacker's questions' IDs stored by Agent B to the number of the attacker's questions' IDs stored by Agent A	X_a
Ratio of the number of legitimate questions that Agent A votes to the number of legitimate questions	$r_{\{s,A\}}$
Ratio of the number of legitimate questions that Agent B votes to the number of legitimate questions	$r_{\{s,B\}}$
Probability that Agent B votes on Agent A's question and that the vote is accepted	$P_{receive}$
Probability that Agent B votes on Agent A's question	$P_{question}$
Probability that Agent A summarizes Agent B's vote	P_{vote}

framework shown in Fig. 5.3a, the probability that Agents A and B are mistakenly judged as similar

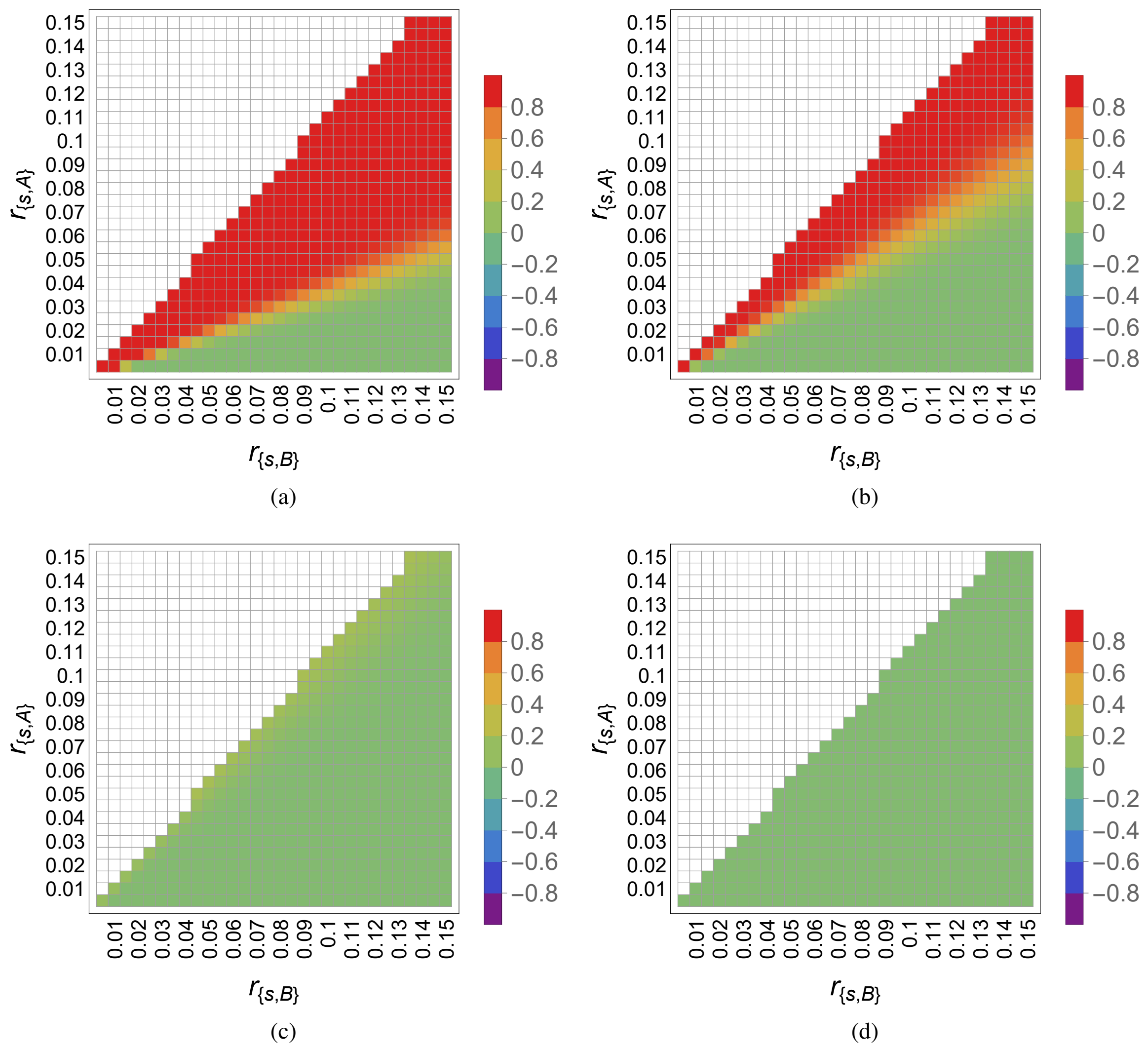


Figure 5.3: Difference of $P_{receive}$ between imitating attack and no attack, where $c = 3$. (a) is the case using the PPCwC framework, where $M_q = 1$, and is also the case for the PPC framework. (b) is the case using the PPCwC framework, where $M_q = 10$. (c) is the case using the PPCwC framework, where $M_q = 60$. (d) is the case using PPCwC, where $M_q = 300$.

by the imitating attack and $P_{receive}$ was high. This is because the PPCwC framework limits the number of questions by nonce searching and suppresses the negative impact of false questions on the similarity judgment. When M_q is small, the PPCwC framework does not suppress the negative impact. Because M_q is the average seconds taken to search for the nonce of a question, we can reduce the negative impact of false questions by setting the nonce search to an appropriate difficulty. However, if the value of M_q is extremely large, the nonce search seconds required for one question becomes too long, which may result in an unimplementable execution time. The conditions on M_q were evaluated in Subsection 5.4.3. In this evaluation, we found that $M_q = 300$ is sufficient to reduce the impact of the fake question attack. Additionally, if the number of question IDs stored by Agent A is less than 1/3 of the number of IDs stored by Agent B, $P_{question}$ and P_{vote} are both small, and there is no change in the case with no attack. By contrast, the impact of the imitating attack is wide when the number of IDs held by the questioner, Agent A, is large.

Similarly, for the case of the isolating attack, we show the changes in $P_{receive}$ compared with the case where there are no attacks by changing $r_{\{s,A\}}$ and $r_{\{s,B\}}$ in Fig. 5.4). For this isolating attack, we set the similarity of Agent B from Agent A as $X_l = 0.8$ and the similarity of the attacker's goal as $X_a = 0.1$. Similar to Fig. 5.3, using the proposed method and setting M_q to an appropriate value, as shown in Fig. 5.4d, the value of $P_{receive}$ is almost the same as in the case of no attacks. This is because, in the PPCwC framework, the search for the nonce requires more time, thus reducing the negative impact of false responses on the similarity judgment. By contrast, in the case of the PPC framework shown in Fig. 5.4a, the isolation attack causes Agents A and B to incorrectly perceive that they are dissimilar, and the probability of voting becomes low. This is caused by the receiving of fake questions geared towards isolating Agents A and B. That is, attempting to suppress the negative impact of the fake questions solely via similarity checks based on the attached IDs is inadequate. The effect of isolating attacks is larger when the number of IDs stored by questioner Agent A is smaller than that of Agent B.

Regarding the fake questions, the proposed countermeasure limits the number of questions by nonce searching in addition to the PPC framework's similarity check based on the IDs attached to questions. This reduces the impact of fake questions to almost the same level as in the case without an attack.

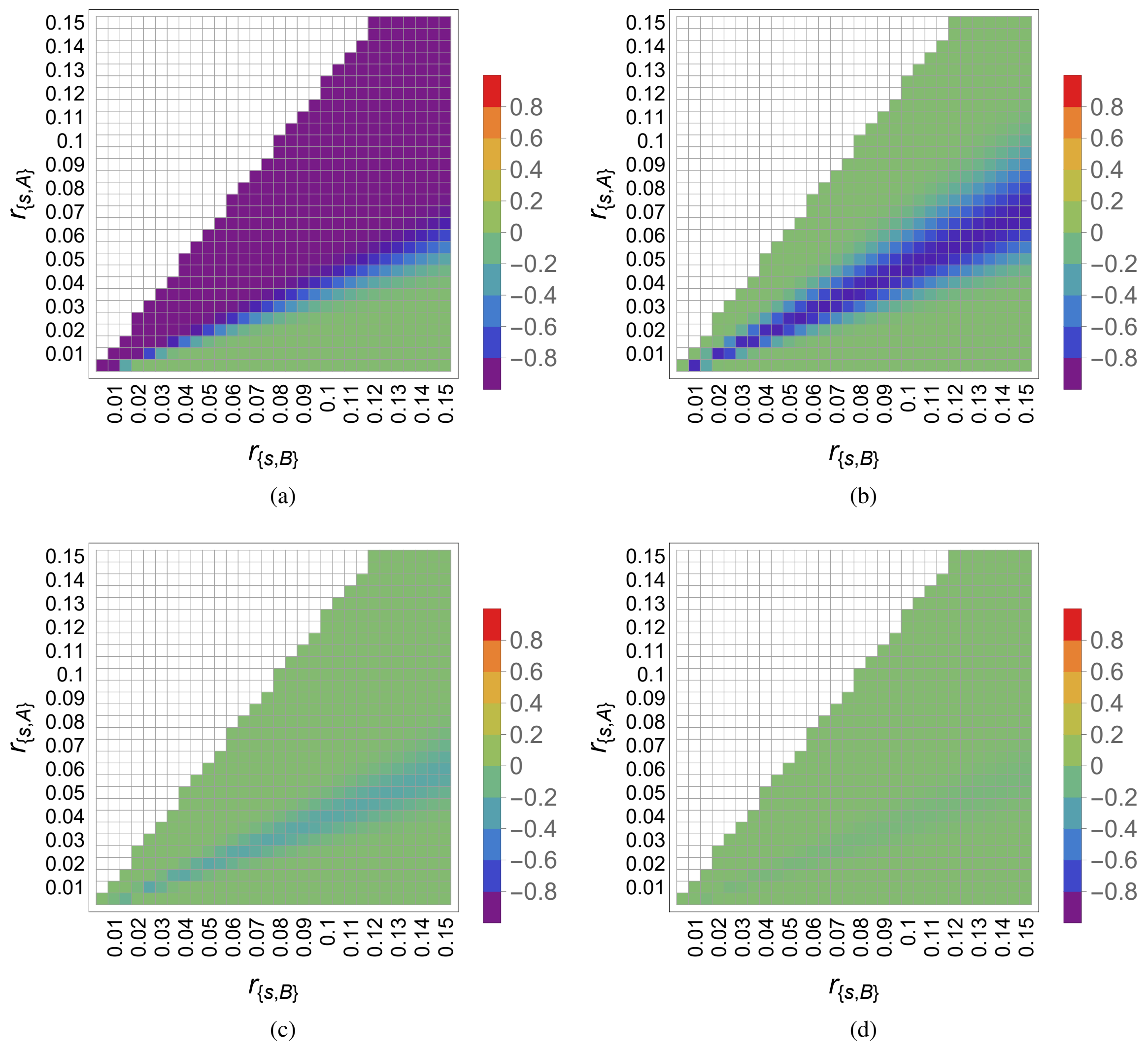
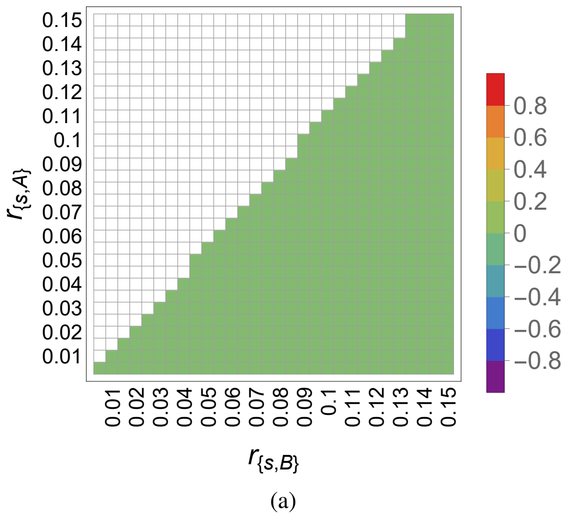
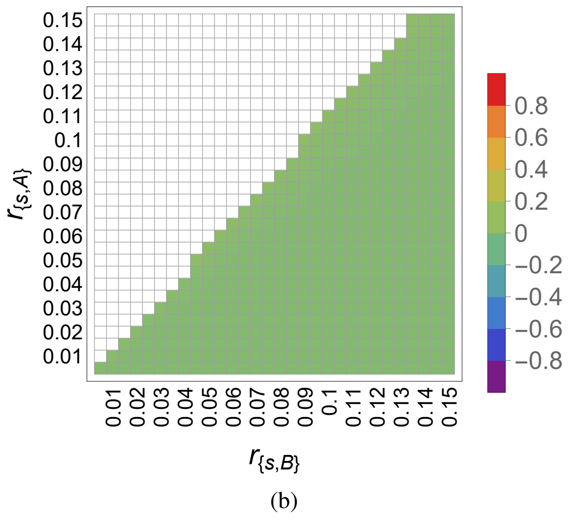


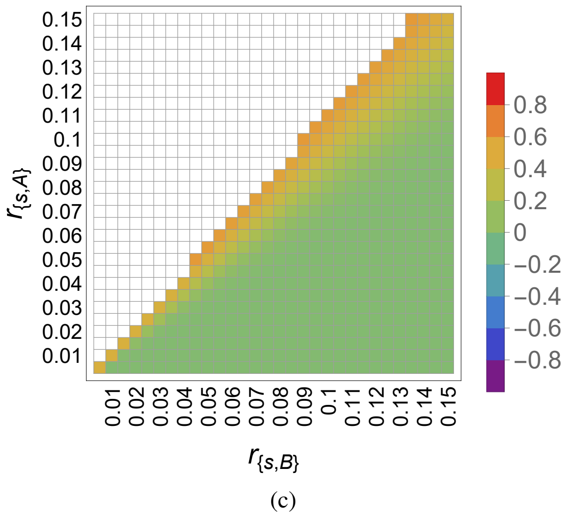
Figure 5.4: Difference of $P_{receive}$ between the isolating attack and no attack, where $c = 3$. (a) is the case using the PPCwC framework, where $M_q = 1$, which is the same as the case of the PPC framework. (b) is the case using the PPCwC framework, where $M_q = 10$. (c) is the case using PPCwC, where $M_q = 60$. (d) is the case using the PPCwC framework, where $M_q = 300$.

Impact of the Amount of Question ID Information Known by the Attacker, c

In the case where the attacker can limit the number of IDs that the target agent stores to $1/c$ of the total number of IDs, the fake question is c times more likely to be accepted by the respondent. For the case of the imitating attack, we show the changes in $P_{receive}$ compared with the case where there







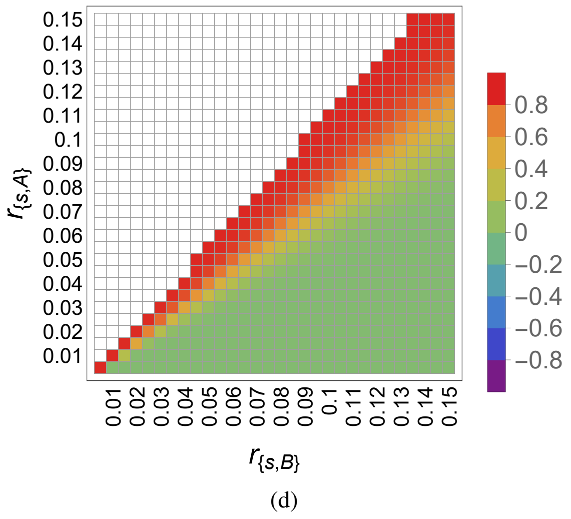


Figure 5.5: Difference of $P_{receive}$ between imitating attack and no attack, where $M_q = 300$. (a) is the case using the PPCwC framework, where $c = 3$. (b) is the case using the PPCwC framework, where $c = 10$. (c) is the case using the PPCwC framework, where $c = 25$. (d) is the case using the PPCwC framework, where $c = 50$.

are no attacks by changing c with $r_{\{s,A\}}$ and $r_{\{s,B\}}$ in Fig. 5.5). As shown in Figs. 5.5a and 5.5b, even when the attacker could limit the number of IDs that Agents A and B stored to $1/3$ and $1/10$ of all IDs, the attack has no impact. This is because the PPCwC framework limits the number of questions, and even if an attacker can limit the information about the question's IDs to $1/10$, the impact of fake questions on the similarity judgment is suppressed. However, as shown in Fig. 5.5c and 5.5d, if the attacker can limit the information on the IDs held by Agents A and B to 4 and 2% of the total IDs, the fake question may affect to the similarity judgement. However, because the number of questions is limited throughout the framework, and there is little opportunity to collect information about the IDs of questions that the attack target is likely to keep, it is difficult for an attacker to limit the question IDs that the attack target is likely to keep to a few percentages. If the attacker trains its own model using the attacker's data as well as the legitimate agent for limiting the question IDs, an attack against the agent of users having similar characteristics as the attacker's own learning model may succeed. However, in this case, the attack target is limited to the agent of the user who has very similar characteristics that can limit the number of stored IDs to only 2%.

5.4.2 Impact of Attacks by Respondent

We ensure that the PPCwC framework suppresses the negative impact of a large number of false votes. When an attacker tries to send a large number of valid votes, it is necessary to attach the IDs stored by the questioner to the vote and search for the nonce value. We ensure that the summarized results are not affected by false votes by calculating the percentage of summarized votes given by the attacker in all summarized votes that the questioner received.

We define the number of summarized votes by legitimate respondents and attackers as a function of time. The total number of summarized votes collected by the questioner, V, is represented by the sum of $V_{legitimate}$—the number of summarized votes by legitimate agents—and V_{attack} that is the number of summarized votes by the attacker. We denote t as the number of computations needed to verify one nonce value and to send votes, represented by (5.15).

$$V(t, c_l, y) = V_{legitimate}(t, c_l) + V_{attack}(t, y). \tag{5.15}$$

When V exceeds a threshold, $V_{threshold}$, the collecting of votes is terminated. Moreover, the minimum value of t that satisfies the condition is t^*, which is obtained using (5.16).

$$t^* = \text{Min}(t | V(t, c_l, y) \geq V_{threshold}). \tag{5.16}$$

The legitimate agent and attacker generate votes on the question and send them to the questioner. When a questioner receives a vote, the questioner verifies its nonce, and whether the same IDs are attached to the ID database of the questioner or not. Therefore, the number of votes to be adopted, $V_{\mathit{legitimate}}$, is obtained by multiplying the number of votes sent by the legitimate agent with the correct nonce, L, by the probability, p_l, that the questioner stores the same IDs as the IDs attached to the votes based on the threshold. This is given by (5.17).

$$V_{\mathit{legitimate}}(t, c_l) = p_l L(t, c_l), \tag{5.17}$$

where p_l is assumed to be 0.9. Because legitimate agents will only send votes on the questions that sent from agents of users having similar characteristics, the probability that the questioner holds the IDs attached to the vote is considered high. When a legitimate agent votes, it must search for a nonce and send the vote as soon as the nonce is found. We set the probability of finding a proper nonce value in one hash calculation to p_n. The probability of finding a nonce by t-time calculations is expressed as $p_n \sum_{k=1} t(1-p_n)^{k-1}$. If the number of agents attempting to vote is c_l, the number of received votes from legitimate agents, L, obtained for the number of computations, t, is expressed by (5.18).

$$L(t, c_l) = c_l p_n \sum_{k=1}^{t} (1-p_n)^{k-1}\,. \tag{5.18}$$

p_n is expressed by (5.19), where the hash calculation outputs a decimal number, and the nonce value satisfies the condition that the top x digits are 0.

$$p_n = \frac{1}{10^x}. \tag{5.19}$$

Similarly, for the attacker's votes, the attacker selects the question IDs for attaching to the vote and searches for a nonce. Similar to (5.17), we use the number of false votes by the attacker, A, and the probability, p_{attack}, that the IDs attached to the attacker's vote match the questioner's ID database with a threshold. We can define V_{attack} by (5.20).

$$V_{attack}(t, y) = p_{attack}(y)A(t). \tag{5.20}$$

Unlike the legitimate respondents, the attacker sends votes multiple times. When the number of nonce searches is executed t times, the probability of finding k nonces is expressed as ${}_tC_k p_n^k (1 - p_n)^{t-k}$, and the expected value of the number of nonces that can be found is $\sum_{k=1}^{t} k_t C_k p_n^k (1 - p_n)^{(t-k)}$. Thus, we assume that the attacker has c_a times the calculation power to a legitimate agent, and the total number of answers, A, sent by the attacker is represented by (5.21).

$$A(t) = c_a \sum_{k=1}^{t} k \binom{t}{k} p_n^k (1 - p_n)^{(t-k)}. \tag{5.21}$$

The probability that the ID attached to the vote by the attacker matches the IDs held by the questioner, p_{attack}, depends on how the attacker limits the IDs held by the questioner. We assume that the attacker keeps B IDs within the expiration period, and yB IDs are common to the questioner's ID database, where y is the ratio of the attacker having the same IDs as the questioner to all the IDs that the attacker keeps. The probability, p_{attack}, that the attacker's fake votes will be summarized by the questioner is the probability that the attacker will select N_V or more IDs of the yB IDs when selecting n_v question IDs from B IDs. It is expressed by (5.22).

$$p_{attack}(y) = \sum_{i=N_v}^{n_v} \frac{\binom{yB}{i}\binom{(1-y)B}{n_v-i}}{\binom{B}{n_v}}. \tag{5.22}$$

In this discussion, attackers have $B = 20,000$ IDs.

We define R_v as the ratio of summarized votes sent by legitimate agents to the total number of summarized votes on a question, as in (5.23).

$$R_v(c_l, y) = \frac{V_{legitimate}(t^*, c_l)}{V(t^*, c_l, y)}. \tag{5.23}$$

When we calculate t^*, we set the threshold, $V_{threshold} = 100$.

Table 5.4: Representations in Section 5.4.2.

Explanation	Representation and value
Number of summarized votes	V
Number of summarized votes from legitimate agent	$V_{legitimate}$
Number of summarized votes from attackers	V_{attack}
Threshold of V	$V_{threshold} = 100$
Ratio of summarized votes by legitimate agents to all summarized votes	R_v
Number of summarized votes from legitimate agent of PPC framework	$\hat{V}_{legitimate}$
Number of summarized votes from attackers of PPC framework	$\hat{V}_{attack}$
Computation times for verifying one nonce value and send a vote	t
Minimum time that the questioner receives $V_{threshold}$ or more votes	t^*
Number of received votes by the legitimate agent with the correct nonce	L
Probability that the agent has stored the same IDs with the IDs attached to the received votes and accepts the vote	$p_l = 0.9$
Probability of finding a nonce value in one hash calculation	p_n
Number of agents attempting to vote	c_l
As a conditional expression for nonce, the number of digits that should be 0 from the top in each digit of the hash value	x
Number of false votes by the attacker	A
Probability that the questioner have stored the same IDs attached to the attacker's vote and accepts the vote	p_{attack}
Calculation power times of attacker to a legitimate agent	c_a
Number of IDs that the attacker holds	$B = 20,000$
Ratio of the attacker having the same IDs as the questioner to all the IDs that the attacker keeps	y

We list the representations of the equations of Subsection 5.4.2 in Table 5.4.

Impact of Number of Agents Who Vote on Questions, c_l, and Amount of Question ID Information Known by the Attacker, y

We calculate R_v for each c_l and y, as shown in Fig. 5.6). In Fig. 5.6, even if there are 50 attackers, the negative impact is suppressed when y is 20% or lower. This is because in the PPCwC framework, the questioner summarizes the votes with avoiding the votes sent from attackers judging by the IDs attached to the votes. Here, we discuss y. The case in which y becomes high, or the case in which the available amount of ID information is large, is rare. One reason for this is that the votes

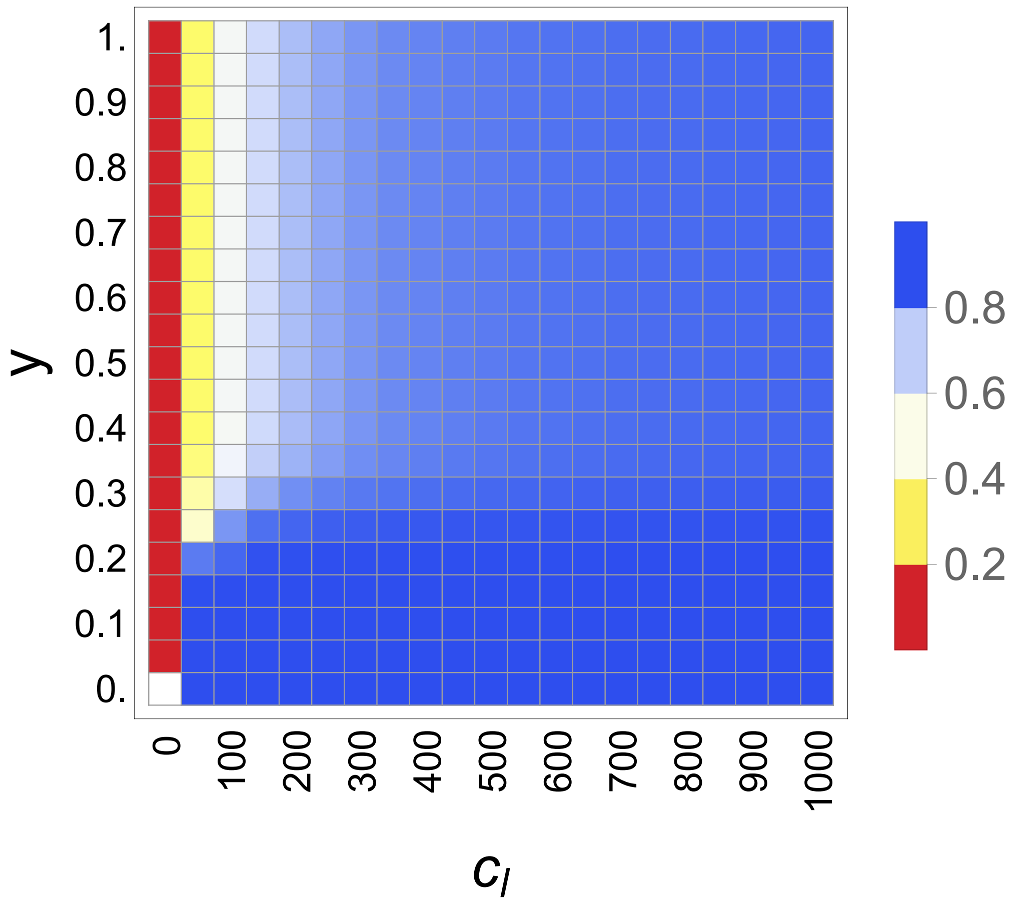


Figure 5.6: Ratio of accepted votes by the legitimate agents out of all accepted votes, R_v, for the number of agents that voted on the question, c_l, and the ratio of the attacker having the same IDs as the questioner to all the IDs that the attacker keeps, y. The attacker sets $c_a = 50$ times the calculation power of a legitimate agent with the PPCwC framework.

are encrypted and the viewer cannot collect the information of IDs attached to the votes. Another reason is that the expiration terms for attaching IDs between questions and votes differ, and the attacker cannot utilize the ID information collected by watching the question as a viewer. Even if the attacker sends fake questions and collects the ID information from the votes on the questions as a questioner, the collectable ID information is limited because the number of questions is limited. Thus, when the questioner answers fewer than 20% of the questions, it is difficult for y to exceed 20%. The border value, 20%, of y is changed by the setting of a parameter N_v; this is discussed in Section 5.4.2. Only in a case where an attacker uses its own data to train a model does it become possible to attack agents of users with similar characteristics using the IDs collected by the attacker's agent. Even in the rare case where y is large, that is, the attacker generates its own model, it is possible to suppress the negative impact with a certain number of votes.

Impact of the Calculation Power of Attackers, c_a

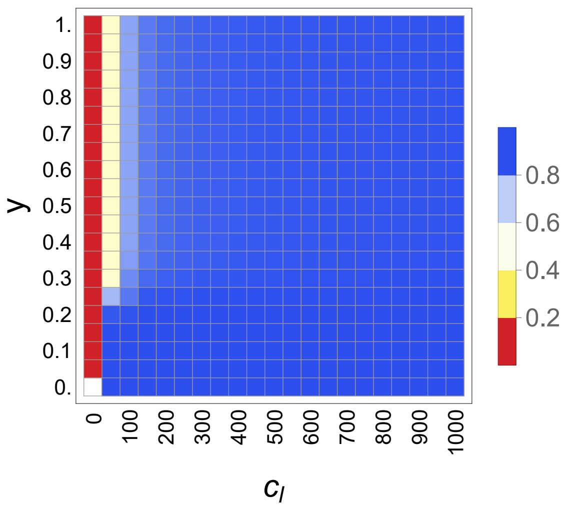
We discuss the calculation power of the attackers, c_a, which is equivalent to the number of attackers. We calculate R_v for each c_l and y, as shown in the heat map in Fig. 5.7). In Fig. 5.7d, even if the number of attackers exceeds the number of legitimate respondents who vote, the proposed countermeasure can suppress the negative impact when y is 0.20 or lower. This is because the questioner judges the similarity to the votes in the PPCwC framework and the framework limits the votes by the nonce searching. Thus, the proposed countermeasure suppresses the negative impact of a large number of false votes.

Impact of the Threshold, N_v, of Matched IDs Between the Attached IDs to Votes and Questioner's ID Database

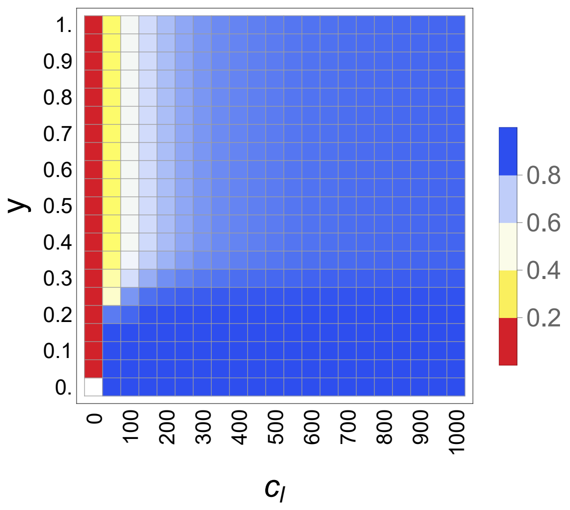
We evaluated R_v by changing the N_v with c_l and y, as illustrated in Fig 5.8. By changing the threshold, N_v, even if there are more attackers than the legitimate votes, and the attacker can limit the IDs held by the questioner, y, from 0.2 to 0.4, the proposed framework can suppress the negative impacts. Thus, by setting appropriate a threshold value for applying services to the PPCwC framework, PPCwC framework can fit multiple types of services.

Comparison of PPCwC and PPC Frameworks

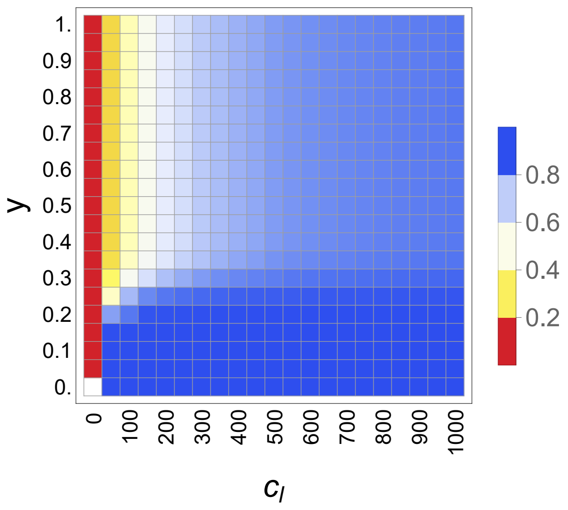
We compare the PPCwC with the PPC framework to discuss the effectiveness of the proposed countermeasure. In the PPC framework, both the attacker and the legitimate agent can generate a vote at $t = 1$ because a questioner does not require nonce searches. Therefore, in the PPC framework, the number of summarized legitimate votes, $\hat{V}_{legitimate}$, and the number of summarized attacker's votes, $\hat{V}_{attack}$, are defined by equations (5.24) and (5.25). Because legitimate agents can



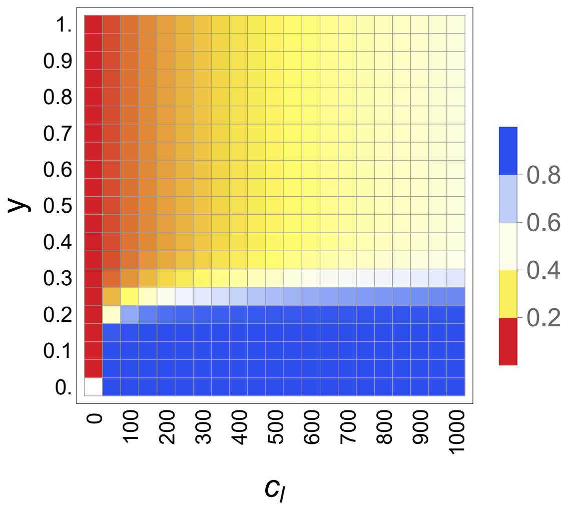
(a)



(b)



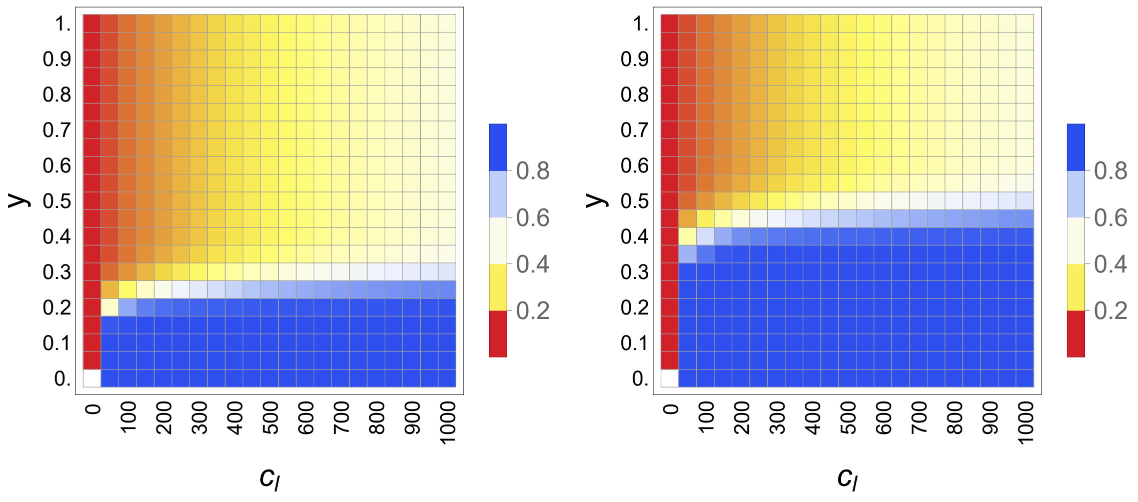
(c)



(d)

Figure 5.7: Ratio of summarised votes by the legitimate agents out of all summarised votes, R_v, for the number of voting agents, c_l and the ratio of the attacker having the same IDs as the questioner to all the IDs that the attacker keeps, y, using the PPCwC framework. (a) Attacker sets $c_a = 10$ times the calculation power of the legitimate agent. (b) $c_a = 50$. (c) $c_a = 100$. (d) $c_a = 1,000$.

generate votes at $t = 1$ and the PPC framework summarizes all votes until the number of received votes is greater than the threshold, $V_{threshold}$, we denote the number of people who voted as c_l



(a) (b)

Figure 5.8: Ratio of summarized votes by the legitimate agents out of all summarized votes, R_v, for the number of voting agents, c_l, and the ratio of the attacker having the same IDs as the questioner to all the IDs that the attacker keeps, y, with the PPCwC framework. There are $c_a = 1,000$ attackers. (a) Threshold of N_v was set to 30. (b) $N_v = 50$.

in (5.24).

$$\hat{V}_{\mathit{legitimate}}(t, c_l) = \begin{cases} 0, & t = 0 \\ c_l, & t > 0. \end{cases} \tag{5.24}$$

Then, because the attacker votes many times, we denote $\hat{V}_{\mathit{attack}}$ using c_a in (5.25).

$$\hat{V}_{\mathit{attack}}(t) = c_a t. \tag{5.25}$$

We set the proposed method and describe the percentage of accepted votes by legitimate agents to all accepted votes, R_v, for the number of legitimate agent who votes on the question, c_l, as shown in Fig. 5.9). We set $y = 0.2$.

The proposed method maintains a high percentage of legitimate votes. For example, in Fig. 5.9d, the PPCwC framework accepted more than 90% of legitimate votes when the 1,000 legitimate agents and the 1,000 attackers voted. That is, the proposed countermeasure suppressed the impact of the fake votes to less than 10%. However, the ratio of accepted legitimate votes of the PPC

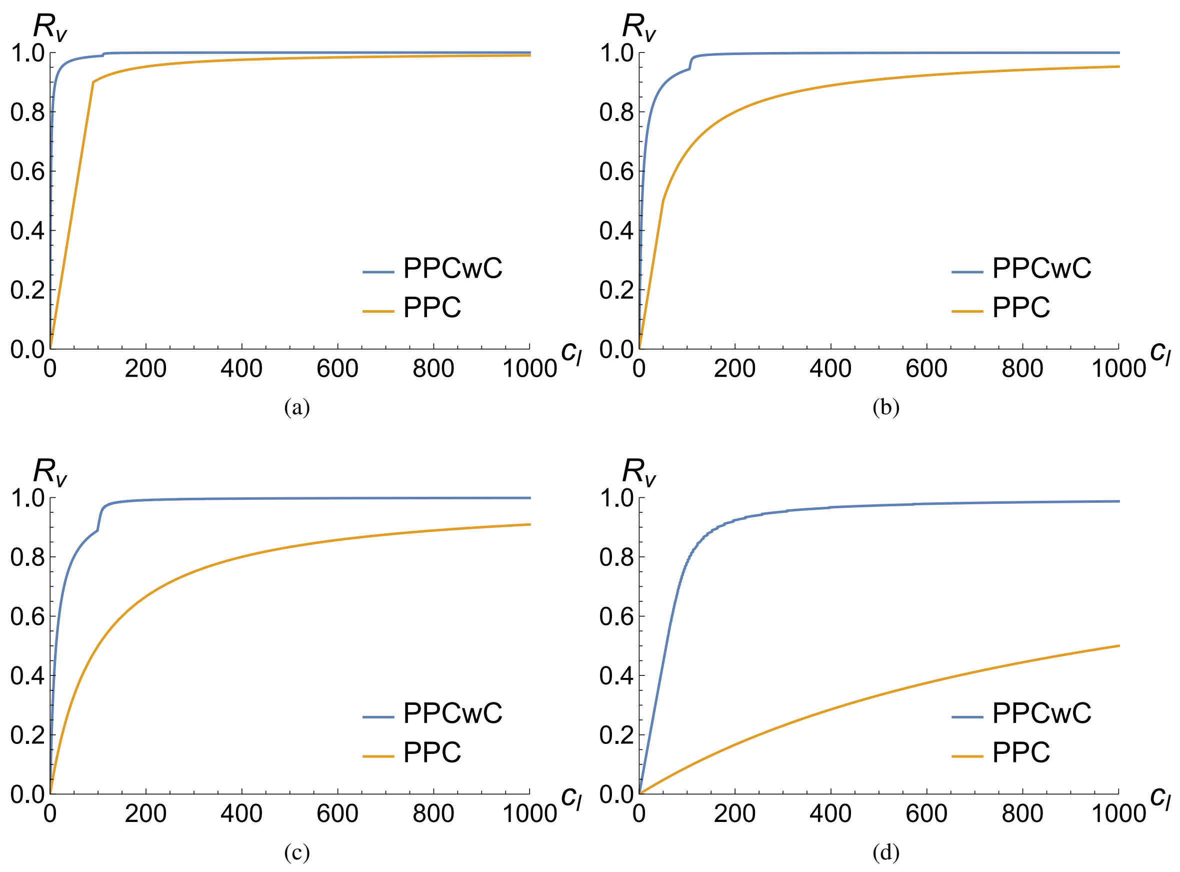


Figure 5.9: Ratio of summarized votes by legitimate agents to all summarized votes, R_v, with the PPCwC and PPC framework to the number of respondents c_l, where $y = 0.2$. (a) The number of attackers is $c_a = 10$. (b) $c_a = 50$. (c) $c_a = 100$. (d) $c_a = 1000$.

framework remains lower than 50%. This is because the number of false votes is limited, and the questioner checks whether the vote is sent from a similar agent and avoids the votes sent by the attackers in the PPCwC framework. We also found that the PPCwC framework is robust to changes in the number of attackers because the ratio of the summarized legitimate votes exceeds 90% in any case in Fig. 5.9.

We confirmed that even when many false votes are given, the impact on the summarized results can be suppressed by conducting a nonce search and determining the similarity by attaching IDs to the votes. Only in a case where an attacker uses its own data to train a model will it be possible to attack agents of users with similar characteristics to the attacker using the IDs collected by the attacker's agent.

5.4.3 Evaluation of Implementability by Computation Time

In this section, we evaluate the relationship between the time required for nonce searching and the number of questions and votes required and check whether the proposed countermeasure can be implemented. If the time required for the nonce search is too long, it will be difficult to ask and answer the required number of questions and votes. Conversely, as discussed in the Sections 5.4.1 and 5.4.2, if a nonce search can be performed easily, the impact of false questions and votes may not be suppressed. Therefore, we confirm that our method is feasible by checking the calculation seconds per day.

The PPCwC framework requires nonces to be searched for the questions and votes. The time required for a question is calculated by multiplying the averaged calculation time for finding a nonce for a question, M_q, and the number of questions, u. The time required for voting is also calculated by multiplying the calculation time for finding a nonce for a vote, M_v, and the number of votes, $R_l r_s$, where R_l is the number of questions asked by all agents in a day, and r_s is the ratio of questions voted by an agent to all questions. The computation seconds, F, per day is represented by the sum of these values. When F does not exceed 1 day, 86,400 seconds, it can be calculated. Thus, the condition is denoted by (5.26).

$$F(u, r_s) = uM_q + R_l r_s M_v \leq 86400. \tag{5.26}$$

At this time, M_v is expressed in (5.27) using the calculation seconds, T_{nonce} s, per hash calculation. We assume that the search for the nonce of a vote is performed at least t times hash calculations, such that the probability of finding the nonce for the vote is greater than the threshold, T_v.

$$M_v = T_{nonce}\text{Min}(t|p_n \sum_{k=1}^{t} (1 - p_n)^{k-1} > T_v), \tag{5.27}$$

where the value of T_{nonce} varies depending on the computing environment, and $p_n \sum_{k=1}^{t} (1 - p_n)^{k-1}$

Table 5.5: Representations in Section 5.4.3.

Explanation	Representation and value
Total computation seconds per day	F
Number of questions per day	u
Seconds required to search one nonce for a question	M_q
Number of questions asked by all agents in a day	R_l
Ratio of questions answered by an agent to all questions	r_s
seconds required to search one nonce for voting	M_v
Calculation seconds per hash calculation	T_{nonce}
Threshold of percentage that an agent found a nonce for a vote	T_v
Hash calculation times	t

is the probability that a nonce satisfying the condition is found for the first time in the t-th hash calculation.

We list the representations in Table 5.5.

We show the computation seconds, F, for r_s and the sum of seconds for the question, uM_q, while varying the seconds, T_{nonce} s, per nonce search in Fig. 5.10 as a heatmap, where $T_v = 0.999$. If the computation seconds does not satisfy the condition of (5.26), we show that in white.

We found that, by setting up the computing environment appropriately and setting the value of T_{nonce} correctly, the constraint on the computation time can be satisfied. For example, regarding the discussed cases in this section, in Fig. 5.10b, we set $u = 1$ and $M_q = 300$ such that the value can avoid the negative impact of the fake questions based on Subsection 5.4.1, and $r_s = 0.025$, which means voting for $1,000$ questions and it can suppress the negative impact, as discussed in Subsection 5.4.2, the calculation seconds, F, is $20,100$ s, which satisfies the condition. Thus, we can prepare a calculation environment in which the value of T_{nonce} is 0.3. Moreover, by changing the calculation resources, the proposed framework is applied to multiple services that require different numbers of questions and votes. In summary, we confirmed that the proposed method can be implemented within a computationally feasible range by preparing a computational environment according to the number of questions and answers.

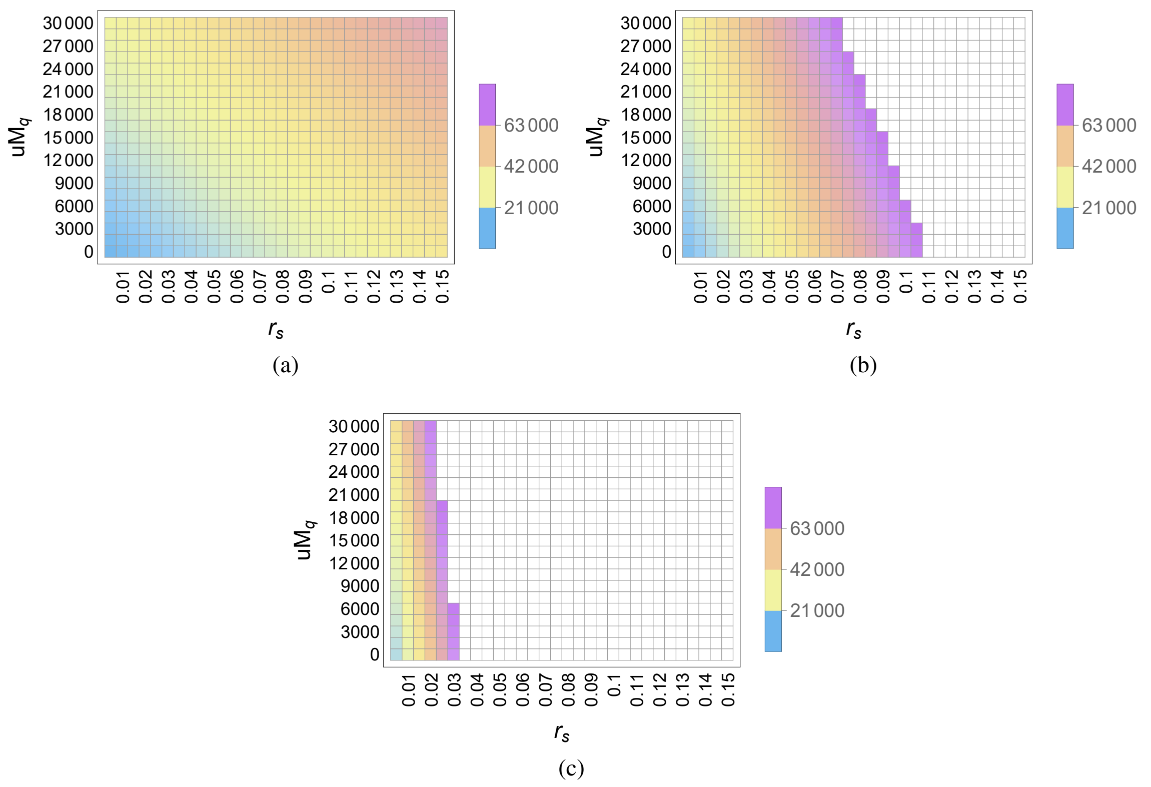


Figure 5.10: Computation seconds, F, for the ratio, r_s, that an agent votes on the questions and the calculation seconds for questions in a day, uM_q, where $T_v = 0.999$. (a) sets T_{nonce} as 0.1. (b) $T_{nonce} = 0.3$. (c) $T_{nonce} = 1.0$.

5.4.4 Limitations of the PPC Framework and the Countermeasure

We analyzed the security risks of the PPC framework, not only those of false questions and vote problems, but also those of service suspension attacks by DoS and acquisition of statistical information and information provider information based on the Information-technology Promotion Agency, Japan's security risk analysis guide [66]. We show the results and countermeasures in Table 5.6. Threats, vulnerabilities, and business damage are rated on a scale of 1 to 3, with higher numbers indicating greater risk. Risk values are rated on a scale of A to E, with A being the highest risk and E being the lowest.

The high risk of attacks on similarity judgments by fake questions and votes and falsifying voting results can be resolved by the proposed countermeasure. The leakage of informants can

Table 5.6: Risks of the PPC framework, their respective risk levels, and countermeasures.

Risk	Threats	Vulnerabilities	Business Damage	Risk Value	Countermeasures
Attack on similarity judgments	3	3	3	A	The proposed countermeasure reduces the impact of a large number of questions.
Falsifying voting results	3	3	3	A	The proposed countermeasure reduces the impact of a large number of votes.
Leakage of informants	1	1	3	C	This can be addressed by anonymization.
Acquisition of statistical information	3	3	1	C	Encryption of votes, and dummy questions can be used to some extent, but owing to the open specification of this framework, the acquisition of statistical information itself cannot be prevented.
Out of service (DoS)	1	3	1	E	It does not affect the agent's learning itself, but the framework is not available, and data from other homes are not available. One of the countermeasures is to monitor the network pathway [67].

be prevented by the anonymization of the PPC framework. However, it is not possible to prevent the leakage of statistical information. This is because this framework is open to everyone and communicates in the open. It is possible to prevent viewers from obtaining information about the votes by encryption. It is also possible to reduce the amount of information by adding noise to the information obtained from questions by using dummy questions. Moreover, because this framework is a decentralized and open framework with which anyone can participate, it is difficult to deal with service outages caused by DoS attacks. When such a service-shutdown attack occurs, it does not give false data to the agents, but the framework becomes unavailable. This means that other users' data are not available, and the data shortage cannot be resolved. One countermeasure method that drops packets of the same signature on the path of the network using software-defined network technology is considered [67].

5.5 Conclusions and Future Work

The privacy-preserved cooperation (PPC) framework enables users to cooperate with others having similar characteristics without sharing information about who the sender is. However, if an attacker sends a large number of false questions and votes, legitimate agents cannot cooperate with the agents of similar users and receive false information. In this Chapter, we proposed a countermeasure for this weak point of the PPC framework that obviates the negative impact of a large number of false questions and votes. In the proposed method, the agents utilize the information that only agents of users having similar characteristics know to avoid accepting false questions and votes. The agents attach the IDs of the questions to which they voted "Yes" to their questions and votes. When the number of matched IDs between the attached IDs and the IDs for which the received agent voted "Yes" exceeds a threshold, the question and the vote are accepted by the received agent. Because fake questions and votes could be accepted if the attached IDs accidentally match the IDs that the receiver stored, the countermeasure also limits the number of questions and votes that agents can send by requiring that a nonce be searched for to enable sending a question and voting. The nonce is an arbitrary value that is added to the attached IDs and the hash value of the added value is less than a specified threshold. We showed that the countermeasure is effective via a numerical example that showed the probability that an agent votes on a question sent from another agent remains unchanged from no-attacks; the ratio of the accepted votes sent from attackers to all accepted votes is suppressed to less than 10%. Moreover, we ensured that the calculation time is reasonable via a numerical example by setting up the computing environment appropriately.

One future task is to confirm that the system can be applied to various social issues by implementing actual applications. We adapted the anomaly detection method of home IoT in Chapter 4. To determine the pros and cons of this framework, we will apply other services and evaluate them. We will also study how to ensure scalability when the number of users increases.

Chapter 6

Conclusion

Anomalous operation attack that an attacker operates home IoT devices by sending packets via an intruded smartphone or an IoT device is an important problem. In this thesis, we proposed a detection method focusing on the behaviors of users. Learning the behaviors sufficiently may require a large amount of data, but the amount of data collected in each home is limited. However, anomalous operations still need to be detected regardless of whether the data amount is sufficient or not. In this thesis, we also proposed a framework to utilize data of similar users without sharing private information.

First, we proposed a method to detect the anomalous operations based on user behavior. 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 could detect over 90% of anomalous operations with less than 10%

of misdetections if the events related to legitimate operations could be monitored. Therefore, we found that the 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. To distinguish whether the single operation is usual usage or not, we proposed the estimation-based method.

Second, to detect anomalous operation attacks on IoT devices in a home, we proposed a detection method that estimates the home state based on the observed values of IoT sensors and device operations and learns the event sequences of users in the home in each estimated state. After training, when a device operation is observed to determine whether it is legitimate or anomalous, the proposed method calculates the occurrence probability of the sequence related to the target operation. If the occurrence probability is lower than the threshold, the operation is detected as anomalous. For this evaluation, we simulated anomaly detection using behavioral logs and sensor data obtained from real homes for one month. We evaluated the improvements of the proposed method and the effectiveness of each part by comparing the proposed method to other methods, one of which did not use sequence information and the other did not estimate the in-home situation. We found that the proposed method achieved a 15.4% higher detection ratio with fewer than 10% misdetections by using the sequence information, and it achieved a 46.0% higher detection ratio with fewer than 10% misdetections by using the estimation of the in-home situation. Thus, the proposed approach can analyze the legitimate behavior of users and legitimate usages of the IoT devices comprehensively by using long- and short-term information, that is, by estimating the home state transition and using the sequence of behaviors. However, a certain amount of data was required to learn the behaviors of users in the home.

Third, we proposed a cooperation framework that utilize the dataset of similar users without sharing their private information to detect anomalous operations. In this platform, an agent was deployed at each home. The agent learned the behavior of the users and detected anomalous operations based on the learned behaviors. However, if the agent did not identify whether the current operation was legitimate or not, due to a lack of training data, it asked the other agents by sending a request. The agent informed the property of the users by attaching the IDs of the past requests that matched the behavior of the corresponding users. Then agents who also identified the past requests of the attached IDs as legitimate replied to it. By doing so, our framework enabled agents to co-operate with similar agents without sharing their private information. We evaluated our framework by using the dataset monitored at real homes. The results indicated that our framework reduced the number of false negatives of anomalous operations by cooperating between similar agents.

Finally, we proposed a countermeasure for the framework that suppress the affection by a large number of false questions and false votes by checking the similarity of the questioner and respondent each other and required computation time through the search for nonce. In this countermeasure, we check the message is sent from the similar agent by checking the attached IDs to the message. If the agent who received the message answered “Yes” to the same questions of the attached IDs, the agent accepts the message. By doing so, the agent avoids to accept the messages that sent from attackers because it is difficult for attackers to attach the IDs that the agent answered “Yes” in the past. By the numerical example, we showed that it is possible to suppress the negative impact of a large number of questions and answers, and that the countermeasure is workable.

By combining all methods proposed in this thesis, we can construct an anomaly detection system that can detect anomalous operations based on user behavior even if each home does not have enough amount of data. In this system, an agent in a home learns user behaviors based on the combination of event sequences and home states. When a command arrives at a home IoT device, the agent verifies whether it matches with the legitimate behaviors. If the command does not agree with the learned behavior, it is classified as an anomaly and dropped. When the agent cannot decide whether the command is legitimate or not, it can ask agents that learn similar behaviors via the cooperation framework.

In this thesis, we did not set a concrete goal of detection percentages, but, our methods could detect anomalous operations that the users cannot observe physically, such as the cases that all users are out of home, sleeping, or doing other tasks. In other words, users can find the operated devices anomalously and can stop the devices by themselves. Hence, the damage from the anomalous operations could be limited. In addition, users can accept some omissions of anomalous operations and misdetections, considering the running of the anomaly detection system. The acceptable range differs from the types of devices and environment of users. For example, if a heater is operated in a home of a student living alone in summer, the damage may be only the electricity cost, but in a home with a baby, the damage would be fatal such as heat stroke. Therefore, in this thesis, we calculated and described the achievable detection ratio with each misdetection for each IoT device; we discussed the accuracy of anomalous operation detection for IoT devices in general. Nonetheless, the final goal of anomalous operation detection for smart home IoT devices is to detect all anomalous operations without misdetections despite the types of IoT devices. However, this requires more different and detailed sensor data, such as location information, usage of smartphones or AI speakers, and biodata obtained from smartwatches, for learning the behaviors. Thus, one of our future work is to achieve the perfect detection.

Another challenge is to apply the method to learn user behaviors to other services such as automations of operating home appliances. By learning user behaviors, we can estimate the users' preferable operations and control home appliance based on the estimated preferences. We also plan to apply our cooperation framework to other applications. For example, by applying our framework to a system that recommends products to users, an agent recommends a product using the information on similar users without sharing their private information.

We believe that the whole discussion in this thesis and the remained research topics above will contribute for activating efforts in development of more various ICT services.

Bibliography

[1] S. Sinha, “State of IoT 2021: Number of connected iot devices growing 9% to 12.3 billion globally, cellular iot now surpassing 2 billion,” https://iot-analytics.com/number-connected-iot-devices/, IoT Analytics, Sep. 2021, accessed: 2021-12-9.

[2] I. Stellios, P. Kotzanikolaou, M. Psarakis, C. Alcaraz, and J. Lopez, “A survey of IoT-enabled cyberattacks: Assessing attack paths to critical infrastructures and services,” *IEEE Communications Surveys and Tutorials*, vol. 20, no. 4, pp. 3453–3495, Jul. 2018.

[3] M. Antonakakis, T. April, M. Bailey, M. Bernhard, E. Bursztein, J. Cochran, Z. Durumeric, J. A. Halderman, L. Invernizzi, M. Kallitsis, D. Kumar, C. Lever, Z. Ma, J. Mason, D. Menscher, C. Seaman, N. Sullivan, K. Thomas, and Y. Zhou, “Understanding the Mirai botnet,” in *Proceedings of the 26th USENIX Security Symposium*, Aug. 2017, pp. 1093–1110.

[4] D. Palmer, “120,000 IoT cameras vulnerable to new Persirai botnet say researchers,” https://www.zdnet.com/article/120000-iot-cameras-vulnerable-to-new-persirai-botnet-say-researchers/, May 2017, accessed: 2021-10-9.

[5] Y. M. P. Pa, S. Suzuki, K. Yoshioka, T. Matsumoto, T. Kasama, and C. Rossow, “IoTPOT: A novel honeypot for revealing current IoT threats,” *Journal of Information Processing*, vol. 24, no. 3, pp. 522–533, May 2016.

[6] M. Lyu, D. Sherratt, A. Sivanathan, H. H. Gharakheili, A. Radford, and V. Sivaraman, “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, Jul. 2017, pp. 46–51.

[7] R. McPherson and J. Irvine, “Using smartphones to enable low-cost secure consumer IoT devices,” *IEEE Access*, vol. 8, pp. 28 607–28 613, Jan. 2020.

[8] C. West and L. Harriss, “Cyber security of consumer devices,” https://post.parliament.uk/research-briefings/post-pn-0593/, Feb. 2019, accessed: 2021-9-15.

[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*, Dec. 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, 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 *Proceedings of International Conference on Wireless Algorithms, Systems, and Applications*, Jun. 2014, pp. 313–323.

[13] K. Xu, F. Wang, L. Gu, J. Gao, and Y. Jin, “Characterizing home network traffic: An inside view,” *Personal and Ubiquitous Computing*, vol. 18, no. 4, pp. 967–975, Apr. 2014.

[14] N. Komninos, E. Philippou, and A. Pitsillides, “Survey in smart grid and smart home security: Issues, challenges and countermeasures,” *IEEE Communication Surveys and Tutorials*, vol. 16, no. 4, pp. 1933–1954, Apr. 2014.

[15] “Vault 7: CIA hacking tools revealed - CIA malware targets iPhone, Android, smart TVs.” https://wikileaks.org/ciav7p1/, Wikileaks, Mar. 2017, accessed: 2021-9-15.

[16] S. Soltan, P. Mittal, and H. V. Poor, “BlackIoT: IoT botnet of high wattage devices can disrupt the power grid,” in *Proceedings of 27th USENIX Security Symposium (USENIX Security 18)*, Aug. 2018, pp. 15–32.

[17] K. Yamanishi, *Anomaly Detection with Data Mining*. Kyoritsu Shuppan Co., Ltd., May 2009, (in Japanese).

[18] M.-O. Pahl and F.-X. Aubet, "All eyes on you: Distributed multi-dimensional IoT microservice anomaly detection," in *2018 14th International Conference on Network and Service Management (CNSM)*, Rome, Italy, Nov. 2018, pp. 72–80.

[19] M. Hasan, M. M. Islan, M. I. I. Zarif, and M. Hashem, "Attack and anomaly detection in IoT sensors in iot sites using machine learning approaches," *Internet of Things*, vol. 7, pp. 100 059:1–100 059:14, Sep. 2019.

[20] R.-H. Hwang, M.-C. Peng, C.-W. Huang, P.-C. Lin, and V.-L. Nguyen, "An unsupervised deep learning model for early network traffic anomaly detection," *IEEE Access*, vol. 8, pp. 30 387–30 399, Feb. 2020.

[21] T. Munir Dipon, M. S. Hossain, and H. S. Narman, "Detecting network intrusion through anomalous packet identification," in *Proceedings of 2020 30th International Telecommunication Networks and Applications Conference (ITNAC)*, Melbourne, VIC, Australia, Nov. 2020, pp. 1–6.

[22] D. Myridakis, G. Spathoulas, and A. Kakarountas, "Supply current monitoring for anomaly detection on IoT devices," in *Proceedings of the 21st Pan-Hellenic Conference on Informatics*, no. 9, Larissa, Greece, Sep. 2017, pp. 1–2.

[23] M. Novák, F. Jakab, and L. Lain, "Anomaly detection in user daily patterns in smart-home environment," *Journal of Selected Areas in Health Informatics (JSHI)*, vol. 3, no. 6, pp. 1–11, Jun. 2013.

[24] S. Ramapatruni, S. N. Narayanan, S. Mittal, A. Joshi, and K. Joshi, "Anomaly detection models for smart home security," in *Proceedings of 2019 IEEE 5th International Conference on Big Data Security on Cloud (BigDataSecurity), High Performance and Smart Computing (HPSC), and Intelligent Data and Security (IDS)*, May 2019, pp. 19–24.

[25] M. Al-Rubaie and J. M. Chang, “Privacy-preserving machine learning: Threats and solutions,” *IEEE Security and Privacy*, vol. 17, no. 2, pp. 49–58, Apr. 2019.

[26] Q. Yang, Y. Liu, T. Chen, and Y. Tong, “Federated machine learning: Concept and applications,” *ACM Transactions on Intelligent Systems and Technology*, vol. 10, no. 2, pp. 1–19, Jan. 2019.

[27] J. Qiu, Q. Wu, G. Ding, Y. Xu, and S. Feng, “A survey of machine learning for big data processing,” *EURASIP Journal on Advances in Signal Processing*, vol. 67, pp. 1–16, May 2016.

[28] “Seattle tech worker arrested for data theft involving large financial services company |usao-wdwa |department of justice,” https://www.justice.gov/usao-wdwa/pr/seattle-tech-worker-arrested-data-theft-involving-large-financial-services-company, Western District of Washington, Jul. 2019, accessed: 2020-2-13.

[29] M. Yamauchi, Y. Ohsita, M. Murata, K. Ueda, and Y. Kato, “Anomaly detection for smart home IoT based on users’ behavior,” *Technical Reports of IEICE (IN2017-58)*, vol. 117, no. 353, pp. 73–78, Dec. 2017, (in Japanese).

[30] ——, “[invited talk] anomaly detection for smart home based on user behavior -ICCE2019 report-,” *Technical Reports of ITE (ME2019-46)*, vol. 43, no. 5, pp. 249–254, Feb. 2019, (in Japanese).

[31] ——, “Anomaly detection for smart home based on user behavior,” in *Proceedings of 37th IEEE International Conference on Consumer Electronics 2019 (ICCE 2019)*, Jan. 2019, pp. 1–10.

[32] ——, “Anomaly detection in smart home operation from user behaviors and home conditions,” *IEEE Transactions on Consumer Electronics*, vol. 66, no. 2, pp. 183–192, May 2020.

[33] M. Tanaka, M. Yamauchi, Y. Ohsita, M. Murata, K. Ueda, and Y. Kato, “Anomaly detection in smart home networks using situation estimation in-home activities,” *Technical Reports of IEICE (IN2019-115)*, vol. 119, no. 461, pp. 219–224, Mar. 2020, (in Japanese).

[34] M. Yamauchi, M. Tanaka, Y. Ohsita, M. Murata, K. Ueda, and Y. Kato, “Modeling home IoT traffic using users’ in-home activities for detection of anomalous operations,” in *Proceedings of 32nd International Teletraffic Congress (ITC32) - PhD workshop*, Sep. 2020, pp. 1–2.

[35] ——, “Smart-home anomaly detection using combination of in-home situation and user behavior,” *CoRR*, vol. abs/2109.14348, pp. 1–13, Sep. 2021.

[36] M. Yamauchi, Y. Ohsita, and M. Murata, “Framework to utilize others’ behavior without sharing privacy information,” *Technical Reports of IEICE (IN2019-114)*, vol. 119, no. 461, pp. 213–218, Mar. 2020, (in Japanese).

[37] ——, “Platform utilizing others’ behavior data to detect anomalous operation hiding private information,” in *Proceedings of 7th IEEE International Conference on Consumer Electronics - Taiwan*, Sep. 2020, pp. 1–2.

[38] ——, “Platform utilizing similar users’ data to detect anomalous operation of home IoT without sharing private information,” *IEEE Access*, vol. 9, pp. 130 615–130 626, Sep. 2021.

[39] K. L. Lueth, “State of the IoT 2018: Number of IoT devices now at 7B - market accelerating,” https://iot-analytics.com/state-of-the-iot-update-q1-q2-2018-number-of-iot-devices-now-7b/, IoT Analytics, Aug. 2018, accessed: 2019-6-5.

[40] 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, Jul. 2015.

[41] M. U. Farooq, M. Waseem, A. Khairi, and P. S. Mazhar, “A critical analysis on the security concerns of Internet of Things (IoT),” *International Journal of Computer Applications*, vol. 111, no. 7, pp. 1–6, Feb. 2015.

[42] 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, no. 3, pp. 1454–1464, Jan. 2017.

[43] G. Wang, M. Atiquzzaman, Z. Yan, and K.-K. R. Choo, "Security and attack vector analysis of IoT devices," in *Proceedings of International Conference on Security, Privacy and Anonymity in Computation, Communication and Storege, SpaCCS 2017 International Workshop*, Dec. 2017, pp. 593–606.

[44] B. B. Zarpelao, R. S. Miani, C. T. Kawakani, and S. C. de Alvarenga, "A survey of intrusion detection in Internet of Things," *Journal of Network and Computer Applications*, vol. 84, pp. 25–37, Apr. 2017.

[45] E. Hodo, X. Bellekens, A. Hamilton, P.-L. Dubouilh, E. Iorkyase, C. Tachtatzis, and R. Atkinson, "Threat analysis of IoT networks using artificial neural network intrusion detection system," in *Proceedings of 2016 International Symposium on Networks, Computers and Communications*, May 2016, pp. 1–6.

[46] T. Rashid, I. Agrafiotis, and J. R. Nurse, "A new take on detecting insider threats: Exploring the use of hidden Markov models," in *Proceedings of the 8th ACM CCS International Workshop on Managing Insider Security Threats*, Oct. 2016, pp. 47–56.

[47] 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 Transactions on Big Data*, vol. 5, no. 3, pp. 305–316, Sep. 2019.

[48] O. Aran, D. Sanchez-Cortes, M.-T. Do, and D. Gatica-Perez, "Anomaly detection in elderly daily behavior in ambient sensing environments," in *Proceedings of International Workshop on Human Behavior Understanding*, Sep. 2016, pp. 51–67.

[49] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer, Feb. 2006.

[50] K. L. Lueth, "State of the IoT 2020: 12 billion IoT connections, surpassing non-IoT for the first time," https://iot-analytics.com/state-of-the-iot-2020-12-billion-iot-connections-surpassing-non-iot-for-the-first-time/, IoT Analytics, Nov. 2020, accessed: 2021-9-15.

[51] A. Sivanathan, H. H. Gharakheili, and V. Sivaraman, “Managing IoT cyber-security using programmable telemetry and machine learning,” *IEEE Transactions on Network and Service Management*, vol. 17, no. 1, pp. 60–74, Mar. 2020.

[52] Z. Whittaker, “Smartwatch hack could trick patients to ‘take pills’ with spoofed alerts,” https://techcrunch.com/2020/07/09/smartwatch-hack-spoofed-alerts/, TechCrunch, Jul. 2020, accessed: 2021-9-15.

[53] D. K. Madhugundu, F. Ahmed, and B. Roy, “A survey on security issues and challenges in IoT based smart home,” in *Proceedings of 3rd International Conference on Internet of Things and Connected Technologies (ICIoTCT) 2018*, May 2018, pp. 423–427.

[54] V. Moustaka, Z. Theodosiou, A. Vakali, and A. Kounoudes, “Smart cities at risk!: Privacy and security borderlines from social networking in cities,” in *Proceedings of The Web Conference 2018 (WWW ’18)*, Apr. 2018, pp. 905–910.

[55] R. Dingledine, N. Mathewson, and P. Syverson, “Tor: The second-generation onion router,” in *Proceedings of 13th USENIX Security Symposium (USENIX Security 04)*, Aug. 2004, pp. 303–320.

[56] M. U. Hassan, M. H. Rehmani, and J. Chen, “Differential privacy techniques for cyber physical systems: A survey,” *IEEE Communications Surveys and Tutorials*, vol. 22, no. 1, pp. 746–789, Mar. 2020.

[57] G. Linden, B. Smith, and J. York, “Amazon.com recommendations: item-to-item collaborative filtering,” *IEEE Internet Computing*, vol. 7, no. 1, pp. 76–80, Jan. 2003.

[58] J. P. Johnson, “Targeted advertising and advertising avoidance,” *The RAND Journal of Economics*, vol. 44, no. 1, pp. 128–144, Apr. 2013.

[59] I. MacKenzie, C. Meyer, and S. Noble, “How retailers can keep up with consumers,” https://www.mckinsey.com/industries/retail/our-insights/how-retailers-can-keep-up-with-consumers, Oct. 2013, accessed: 2021-12-18.

[60] T. Higuchi, N. Yanai, K. Ueda, Y. Ishihara, and T. Fujiwara, "A privacy risk arising from communication with TV: Consideration from attribute inference for users," in *Proceedings of the Network and Distributed System Security Symposium (NDSS) 2019*, San Diego, California, Feb. 2019, pp. 1–2.

[61] W. Song and W. Zhuang, "Multi-service load sharing for resource management in the cellular/wlan integrated network," *IEEE Transactions on Wireless Communications*, vol. 8, no. 2, pp. 725–735, Feb. 2009.

[62] S. Shojaee, A. Azman, M. Murad, N. Sharef, and N. Sulaiman, "A framework for fake review annotation," in *Proceedings of 2015 17th IEEE UKSIM-AMSS International Conference on Modelling and Simulation*, Mar. 2015, pp. 153–158.

[63] A. Mukherjee, V. Venkataraman, B. Liu, and N. Glance, "Fake review detection: Classification and analysis of real and pseudo reviews," *Technical Report, Department of Computer Science (UIC-CS-2013-03). University of Illinois at Chicago*, Mar. 2013.

[64] A. Back, "Hashcash - a denial of service counter-measure," http://www.hashcash.org/papers/hashcash.pdf, Aug. 2002, accessed: 2021-12-3.

[65] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," https://bitcoin.org/bitcoin.pdf, Oct. 2008, accessed: 2021-12-3.

[66] "Security risk assessment guide for industrial control systems," https://www.ipa.go.jp/files/000078098.pdf, Oct. 2019, accessed: 2021-2-19.

[67] M. E. Ahmed and H. Kim, "DDoS attack mitigation in internet of things using software defined networking," in *Proceedings of the 2017 IEEE 3rd International Conference on Big Data Computing Service and Applications (BigDataService)*, Apr. 2017, pp. 271–276.