Received 19 May 2023, accepted 13 June 2023, date of publication 21 June 2023, date of current version 27 June 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3288337

RESEARCH ARTICLE

Anonymous Broadcast Authentication With One-to-Many Transmission to Control IoT Devices

KAZUHIKO MINEMATSU[1,2], JUNJI SHIKATA[2,3], (Member, IEEE), YOHEI WATANABE[4,5], (Member, IEEE), AND NAOTO YANAI[5,6], (Member, IEEE)

[1]Secure System Research Laboratories, NEC, Kawasaki 211-8666, Japan
[2]Institute of Advanced Science, Yokohama National University, Yokohama 240-0067, Japan
[3]Graduate School of Environment and Information Sciences, Yokohama National University, Yokohama 240-0067, Japan
[4]Graduate School of Informatics and Engineering, The University of Electro-Communications, Chofu 182-8585, Japan
[5]Japan Datacom Company Ltd., Suita 565-0871, Japan
[6]Graduate School of Information Science and Technology, Osaka University, Suita 565-0890, Japan

Corresponding author: Yohei Watanabe (watanabe@uec.ac.jp)

This work was supported by the Ministry of Internal Affairs and Communications, Japan, conducted under a Contract of ''Research and Development on Internet of Things Malware Removal/Make it Non-Functional Technologies for Effective Use of the Radio Spectrum'' among ''Research and Development for Expansion of Radio Wave Resources'' under Grant JPJ000254.

ABSTRACT We consider a basic system to securely and remotely control many IoT devices. Specifically, we require that: 1) a system manager broadcasts information to IoT devices, e.g., wireless environment, only the designated devices can identify operations sent from the manager; 2) each IoT device can detect (malicious) manipulation of the broadcast information and hence prevents maliciously generated operations from being executed. In this paper, we introduce *anonymous broadcast authentication* (ABA) as a core cryptographic primitive of the basic remote-control system. Specifically, we formally define the syntax and security notions for ABA so that it achieves the above requirements. We then show provably-secure ABA constructions and their implementations to provide their practical performance. Our promising results show that the ABA constructions can remotely control devices over a typical wireless network within a second.

INDEX TERMS Anonymous broadcast authentication, applied cryptography, message authentication codes, provable security, remote control system.

I. INTRODUCTION

Internet-of-Things (IoT) are now an indispensable part of our daily lives, and we are surrounded by many kinds of IoT devices such as smart speakers and sensors. According to Cisco's report [1], tens of billions of IoT devices are expected to be deployed over the next few years. Besides, recently edge AI devices, whereby the devices managed by a user contain machine learning models, appeared, and hence more various advanced IoT devices would be introduced shortly [2]. Along with the rapid development of IoT technologies, we have to focus our efforts on cybersecurity. However, unfortunately, most IoT devices do not seem to do enough to protect the data stored inside for some reason, such as the development cost and constrained resources. This poor security state indeed has a profound effect on the real world; let us give an example. Mirai, a notorious IoT malware, infected many IoT devices, turning them into botnets. The botnets infected nearly 65,000 IoT devices in its first 20 hours [3]. The widespread outbreak of Mirai made a considerable impact on the world.

One may think IoT devices can implement and deploy security functions for each specific security threat. Indeed, it does not lead to a comprehensive solution since most IoT devices' resources are constrained [4]. An adversary may also find out new types of attacks to circumvent existing solutions [5]. There, hence, seem to be no versatile solutions [6], and then we need to discuss a solution dealing with various threats and realizing IoT devices' security.

The associate editor coordinating the review of this manuscript and approving it for publication was Sabu M. Thampi.

A. OUR GOALS AND CONTRIBUTIONS

In this paper, we aim to present a promising solution for IoT devices' security from the cryptographic lens. We focus on arbitrarily controlling devices infected with malware and develop a framework to control IoT devices remotely towards deployment in existing IoT systems. We emphasize that we prevent malicious behaviors of infected IoT devices, not prevent infections themselves. One of our solution's gaols is to bring malicious IoT devices halt so it reduces damages by various attacks such as DDoS attacks.

Specifically, we consider a basic system where a central entity (e.g., a system manager) wants to remotely and simultaneously control many target entities (i.e., IoT devices). Note that due to numerous IoT devices, we here consider one-to-many communications, not individual communication between the central entity and each IoT device. In the system, a system manager broadcasts command,[1] called an abort instruction for the sake of convenience, to all IoT devices *so that only designated devices can execute the command*. Suppose that many devices (though not all) are infected. The system manager can then designate infected devices to bring them to a halt, while non-designated devices (which receive the same instruction) do not stop. Consequently, even when vulnerable devices are attacked, their resultant damage can be reduced by controlling them. We believe that this simplified model of the remote control system captures characteristics of the IoT era and massive machine-type communications (mMTC) in the context of (Beyond) 5G. In Section II, We describe the simple model for the above remote control system and the above essential requirements more concretely.

We then introduce *anonymous broadcast authentication* (ABA) as a core cryptographic primitive that fulfills the requirements for our remote control system. ABA allows the system manager to choose an arbitrary subset of IoT devices and create authenticated commands so that only the designated targets' verification algorithms accept them. We formalize the syntax and security notions, *unforgeability* and *anonymity*, for ABA in Section III. In Section IV, we show two concrete ABA constructions and prove that the proposed constructions are secure in the sense of unforgeability and anonymity. Our ABA schemes can be constructed from only message authentication codes (MACs) and pseudorandom functions (PRFs), thus providing efficient running time. In Section V, we discuss further improvements of our ABA schemes. Finally, we evaluate the performance of the ABA schemes through implementation and experiments in Section VI. Our experimental results show that all the processes can be executed in one second under a typical wireless network setting. We have released source code of the ABA schemes for reproducibility of the experimental results.

B. RELATED WORK

To the best of our knowledge, there are no cryptographic solutions to how many IoT devices are controlled efficiently and securely. Though several "broadcast authentication" protocols [9], [10], [11], [12], [13] have been proposed, they have different features from ABA: they do not have any functionality to designate a subset of receivers. The cryptographic protocols closely related to ABA are broadcast encryption [7], [8], which enables a sender to encrypt a plaintext and designate any subset of receivers so that only designated receivers can decrypt the encrypted plaintext. Anonymous broadcast encryption [14], [15], [16], [17] further guarantees that no useful information on the designated receivers is leaked. Thus, ABA can be regarded as anonymous broadcast encryption in the authentication setting, although we specifically focus on ABA that suits the IoT setting.

To realize the IoT security, researchers have devoted effort from the firmware level [18], [19] to the application [20], [21] instead of the use of cryptography. Whereas the conventional approach is based on the data flow [22], [23], [24], the use of cryptography is discussed in the past years [25], [26], [27]. ABA is a framework to realize IoT security by leveraging cryptography.

A typical approach for IoT security in recent years is the use of machine learning [28], [29], [30], [31] or trusted execution environments [32], [33]. The approaches described above assume a central server that contributes to the IoT security instead of resource-constrained IoT devices. In contrast, ABA enables IoT devices themselves to contributes to IoT security despite a pretty computational cost. There is blockchain as a cryptographic tool where IoT devices contribute to IoT security, and many blockchain-based applications [34], [35], [36], [37] have been proposed in recent years. They provide decentralized systems but require computational costs such as Proof of Work, whereas compared to them, ABA can provide higher throughput for centralized systems.

C. REFINEMENTS

A conference version of this paper appeared at [38], and this paper is its full version. This paper includes several improvements and new results that did not appear in the conference version. We list them as follows.

- We simplify the first construction shown in [38] without degrading its security level. Specifically, We remove a random permutation and one of two PRFs from the first construction in [38], and only require a MAC and PRF for the construction.
- We formally provide security proofs of our proposed constructions (and details of their building blocks), while the conference version only provides sketches.
- We discuss how the constructions can be improved in terms of efficiency. Specifically, we briefly describe how to improve each scheme beyond the syntax of ABA.
- Although unforgeability and anonymity, which we mainly focus on, seem to be sufficient for many applications, we discuss how our constructions can achieve stronger security levels. Specifically, we consider *stronger anonymity* and the *confidentiality of commands* and show how to achieve them; those notions

[1] We use the term "broadcast" rather than "multicast" since we consider *cryptographic broadcast* as in the context of broadcast encryption [7], [8] in this paper.

might be important in some situations, though they are theoretical improvements rather than practical ones.

- We discuss how we efficiently avoid replay attacks against our constructions. In contrast to the above strong notions, the efficient countermeasure against replay attacks is quite important in practice. Indeed, our implementation, which will be explained below, includes this countermeasure.
- We provide implementations and experimental evaluations of our constructions to show their performance. We have also released our source code via GitHub, although we did not implement and evaluate it in the conference version [38]. We confirm that our ABA constructions are practical over typical network settings through the implementations and evaluations.

D. NOTATION

For all natural number $n \in \mathbb{N}$, $\{1, \ldots, n\}$ is denoted by $[n]$. For a finite set $\mathcal{X}$, we denote by $|\mathcal{X}|$ the cardinality of $\mathcal{X}$. For finite sets $\mathcal{X}, \mathcal{Y}$, let $\mathcal{X} \triangle \mathcal{Y}$ be the symmetric difference $\mathcal{X} \triangle \mathcal{Y} := (\mathcal{X} \setminus \mathcal{Y}) \cup (\mathcal{Y} \setminus \mathcal{X})$. For any finite set $\mathcal{X}$ and any natural number $\ell \in \mathbb{N}$, let $2^{\mathcal{X}}_{\leq \ell} := \{\mathcal{Y} \subset \mathcal{X} \mid |\mathcal{Y}| \leq \ell\}$ be a family of subsets of $\mathcal{X}$ such that its cardinality is at most ℓ (i.e., a part of a power set of $\mathcal{X}$). Concatenation is denoted by $\|$. For any algorithm A, $\mathsf{out} \leftarrow \mathsf{A}(\mathsf{in})$ means that A takes in as input and outputs out. If we write $\mathsf{A}(\mathsf{in}; r)$, r indicates an internal randomness that is chosen uniformly at random. Throughout the paper, we denote by κ a security parameter and consider probabilistic polynomial-time algorithms (PPTAs). We say a function $\mathsf{negl}(\cdot)$ is negligible if for any polynomial $\mathsf{poly}(\cdot)$, there exists some constant $\kappa_0 \in \mathbb{N}$ such that $\mathsf{negl}(\kappa) < 1/\mathsf{poly}(\kappa)$ for all $\kappa \geq \kappa_0$.

II. SYSTEM MODEL

This section describes a basic model for the remote control system we consider throughout this paper. The system aim to have only designated devices securely execute commands requested by the system manager. Specifically, we consider the following basic system.

(1) The system manager decides a command and the corresponding devices that execute it.
(2) The system manager sends the command to *all* devices via a broadcast channel.
(3) Each device executes the command if it is designated, or does not otherwise.

As described above, we assume broadcast channels and that each device receives commands without errors. We would like to emphasize that each command is broadcasted and is common for all devices. Moreover, we do not consider upstream communication (i.e., from devices to the manager). The system manager can designate devices with some identifiers associated with them. We assume that those identifiers are public information. Note that we do not consider internal attacks by the system manager since we focus on bringing *malicious IoT devices*, which are resource-constrained devices that are more vulnerable than computers, halt.

We give essential requirements, which should be resolved by mechanism design, for the system below.

Requirement 1: to accurately control which devices execute commands. We require the following correctness of the system.

- The devices designated by the manager surely execute a command.
- The non-designated devices never execute a command.

Requirement 2: to have resilience against falsification of commands. Roughly speaking, each device should confirm the validity of transmitted commands, i.e., whether the commands are (maliciously) modified. We consider the strongest attack in this context; the adversary tries to maliciously modifies commands transmitted over broadcast channels so that at least one non-designated device executes the modified commands. Note that the adversary may obtain information about a certain number of IoT devices, which might have their secret information, by infecting them with malware, (physically) stealing them, etc. Therefore, we require that even if the adversary (maliciously) modifies commands transmitted over broadcast channels, one cannot have non-designated devices execute the (modified) commands; the devices can detect the modifications and halt executing the commands. Note that we do not consider attacks whose aim is to make commands undelivered such as jamming, and the falsification of verification mechanisms in IoT devices through their firmware updates.

Requirement 3: to hide which devices are designated. The information on which devices are designated should not be leaked from broadcast information and operations. In general, cybersecurity-related information, such as vulnerable devices, should be treated as sensitive information [39]. More specifically, if an adversary observes that an abort instruction is sent to some device, they can know that the device is vulnerable through the instruction. Namely, naively sending a command that includes the information on designated devices may attract further attacks. Therefore, we require that it should be hidden.

Requirement 4: to have efficient procedures and compact commands. As described above, commands are broadcasted to all devices. Therefore, taking into account practical situations such as the bandwidth of broadcast channels the manager would use, the size of commands should be small enough. Furthermore, we assume IoT devices, and hence, their resources might be poor. Thus, the process executed by the devices should be efficient enough. Ideally, the system should be efficient so that even microcomputers such as ARM Cortex-M3 can execute commands.

III. ANONYMOUS BROADCAST AUTHENTICATION

In this section, we put forward anonymous broadcast authentication (ABA), which is a core cryptographic primitive for the remote control system described in the previous section. We will describe how to realize the system from ABA after the syntax definition of ABA. The requirement 1 follows from verification correctness below, and the requirements 2 and 3 follow from security notions of ABA. We also aim to satisfy the requirement 4 with our constructions in Section IV.

A. SYNTAX

We introduce ABA $\Pi = (\mathsf{Setup}, \mathsf{Join}, \mathsf{Auth}, \mathsf{Vrfy})$ as follows. Suppose that there are a system manager and a lot of IoT devices. At the beginning of the protocol, a manager runs Setup to get an authentication key ak. Each device is *activated* by executing Join, which generates a verification key $\mathsf{vk}_{\mathsf{id}}$ for the device's identifier id and embeds it inside. Let $\mathcal{D}$ be an identifier set of activated devices. We assume the maximum number of devices that join the protocol is predetermined, and we describe it as N. The manager can run Auth with ak to generate an authenticated command $\mathsf{cmd}_{\mathcal{S}}$ for a command m with a privileged set $\mathcal{S} \subset \mathcal{D}$ so that only $\mathsf{id} \in \mathcal{S}$ can check the validity of $\mathsf{cmd}_{\mathcal{S}}$. Note that $\mathcal{S}$ should satisfy $|\mathcal{S}| \leq \ell$, where ℓ is also predetermined value at the beginning of the protocol. Each device designated by the manager (i.e., each device in $\mathcal{S}$) runs Vrfy with $\mathsf{vk}_{\mathsf{id}}$ to check the validity of $\mathsf{cmd}_{\mathcal{S}}$. If the device received $\mathsf{cmd}_{\mathcal{S}}$ as it is, Vrfy outputs m, which should be the same as what the manager sent. Otherwise, it outputs $\perp$, which indicates "rejection." Non-designated devices (i.e., devices whose identifiers are included in $\mathcal{D} \setminus \mathcal{S}$) definitely output $\perp$ regardless of whether or not $\mathsf{cmd}_{\mathcal{S}}$ is (maliciously) modified. Formally, the algorithms of ABA are defined below.

1) $\mathsf{Setup}(1^\kappa, N, \ell) \to \mathsf{ak}$: a probabilistic algorithm for setup. It takes a security parameter 1^κ, the maximum number of devices $N \in \mathbb{N}$, the maximum number of devices ℓ designated at once as input, and outputs an authentication key ak.
2) $\mathsf{Join}(\mathsf{ak}, \mathsf{id}) \to \mathsf{vk}_{\mathsf{id}}$: a verification-key generation algorithm. It takes ak and an identifier $\mathsf{id} \in \mathcal{I}$ as input, and outputs a verification key $\mathsf{vk}_{\mathsf{id}}$ for id, where $\mathcal{I}$ is a set of all possible identifiers. We assume that $\mathcal{D}$ is simultaneously updated (i.e., $\mathcal{D} := \mathcal{D} \cup \{\mathsf{id}\}$).
3) $\mathsf{Auth}(\mathsf{ak}, \mathsf{m}, \mathcal{S}) \to \mathsf{cmd}_{\mathcal{S}}$: an algorithm for generating authenticated commands. It takes ak, a command $\mathsf{m} \in \mathcal{M}$, and a privileged set $\mathcal{S} \subset \mathcal{D}$ as input, and outputs $\mathsf{cmd}_{\mathcal{S}} \in \mathcal{T}$, where $\mathcal{M}$ and $\mathcal{T}$ are sets of commands and authenticated commands, respectively.
4) $\mathsf{Vrfy}(\mathsf{vk}_{\mathsf{id}}, \mathsf{cmd}_{\mathcal{S}}) \to \mathsf{m} / \perp$: a deterministic algorithm for verification. It takes $\mathsf{vk}_{\mathsf{id}}$ and $\mathsf{cmd}_{\mathcal{S}}$ as input, and outputs m if it accepts $\mathsf{cmd}_{\mathcal{S}}$. Otherwise (i.e., it outputs $\perp$), it rejects it.

Verification Correctness. For all $\kappa, N \in \mathbb{N}$, all ℓ such that $1 \leq \ell \leq N$, all $\mathsf{ak} \leftarrow \mathsf{Setup}(1^\kappa, N, \ell)$, all $\mathcal{D} \subset \mathcal{I}$, all $\mathsf{m} \in \mathcal{M}$, all $\mathcal{S} \subset \mathcal{D}$ such that $|\mathcal{S}| \leq \ell$, the following holds with overwhelming probability:

$$\begin{cases} \mathsf{Vrfy}(\mathsf{Join}(\mathsf{ak}, \mathsf{id}), \mathsf{Auth}(\mathsf{ak}, \mathsf{m}, \mathcal{S})) \to \mathsf{m} & \text{if } \mathsf{id} \in \mathcal{S}, \\ \mathsf{Vrfy}(\mathsf{Join}(\mathsf{ak}, \mathsf{id}), \mathsf{Auth}(\mathsf{ak}, \mathsf{m}, \mathcal{S})) \to \perp & \text{if } \mathsf{id} \notin \mathcal{S}. \end{cases}$$

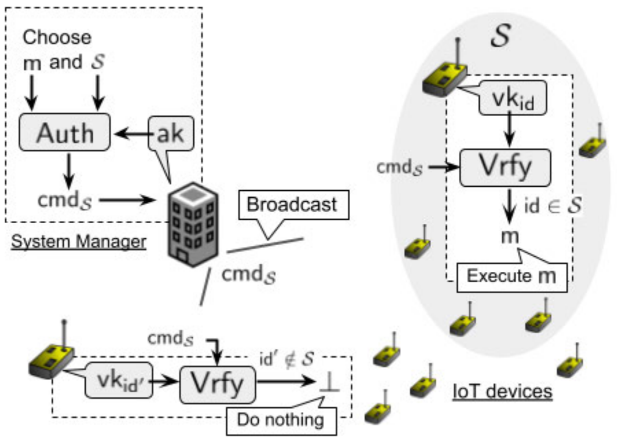
Choose
m and $\mathcal{S}$
Auth
ak
$\mathsf{cmd}_{\mathcal{S}}$
System Manager
Broadcast
$\mathsf{cmd}_{\mathcal{S}}$
$\mathcal{S}$
$\mathsf{vk}_{\mathsf{id}}$
$\mathsf{cmd}_{\mathcal{S}}$
Vrfy
$\mathsf{id} \in \mathcal{S}$
m
Execute m
$\mathsf{cmd}_{\mathcal{S}}$
$\mathsf{id}' \notin \mathcal{S}$
$\mathsf{vk}_{\mathsf{id}'}$
Vrfy
$\perp$
Do nothing
IoT devices

FIGURE 1. The overview of the remote control system from ABA.

Remark 1 (On the Syntax): One may think that the verification algorithm should be $\mathsf{Vrfy}(\mathsf{vk}_i, \mathsf{m}, \mathsf{cmd}_{\mathcal{S}}) \to \top / \perp$, *instead of* $\mathsf{Vrfy}(\mathsf{vk}_i, \mathsf{cmd}_{\mathcal{S}}) \to \mathsf{m} / \perp$, *as in traditional MAC/signatures. The reason why we adopt the above syntax is that we would like to emphasize that only designated devices obtain and execute the command* m. *Moreover, this syntax, which is similar to authenticated encryption [40], allows us to consider "confidentiality" of* $\mathsf{cmd}_{\mathcal{S}}$ *since* Vrfy *does not explicitly take* m *as input (see Section V-B for details).*

B. REMOTE CONTROL SYSTEM FROM ABA

Fig. 1 illustrates how we realize our remote control system with ABA. Each IoT device with its identity id has its own verification key $\mathsf{vk}_{\mathsf{id}}$ embedded in, e.g., a manufacturing process. According to which devices the system manager wants to allow to execute a command m, the manager designates a subset $\mathcal{S}$ of all devices, and runs Auth with m and $\mathcal{S}$. The manager broadcasts the resultant authenticated command $\mathsf{cmd}_{\mathcal{S}}$ to all devices. Each device check the validity of $\mathsf{cmd}_{\mathcal{S}}$, and executes the command m only if Vrfy outputs m (it means the device is in $\mathcal{S}$). On the other hand, Vrfy outputs $\perp$ if the device is not in $\mathcal{S}$, which means the device is not designated. The non-designated device then does nothing. Therefore, the system manager successfully designates devices so that only the designated ones execute commands due to the correctness of ABA; it means the requirement 1 described in Section II is satisfied. Moreover, the requirements 2 and 3 are satisfied by unforgeability and anonymity, which are security notions of ABA formally introduced in the next sections. Specifically, due to unforgeabiility, it is difficult for any adversary to forge the authenticated command $\mathsf{cmd}_{\mathcal{S}}$ so that non-designated devices execute the command m. Anonymity guarantees that $\mathsf{cmd}_{\mathcal{S}}$ leaks no useful information on $\mathcal{S}$. Note that the requirement 4 is about efficiency, and so we will discuss it in Section IV, which shows our ABA constructions.

C. UNFORGEABILITY

Unforgeability is a fundamental security requirement for ABA. Unforgeability against chosen message attacks (UF-CMA), which is a fundamental security requirement for ABA, guarantees that even if an adversary A with polynomially many corrupted devices attempts to modify authenticated commands, it is difficult for A to derive another authenticated command such that at least one honest (i.e., uncorrupted) and non-designated device accepts m. Specifically, we consider the following experiment $\mathsf{Exp}^{\mathsf{CMA}}_{\Pi,\mathsf{A}}(\kappa, N, \ell)$ between a challenger C and any PPTA A.

$\boxed{\mathsf{Exp}^{\mathsf{CMA}}_{\Pi,\mathsf{A}}(\kappa, N, \ell)}$ C runs $\mathsf{Setup}(1^\kappa, N, \ell)$ to get ak. Let $\mathcal{W}$, $\mathcal{M}_\mathsf{a}$, and $\mathcal{M}_\mathsf{v}$ be empty sets, and flag be a flag initialized as zero. $\mathcal{W}$ indicates sets of identifiers of corrupted devices during the experiment, and $\mathcal{M}_\mathsf{a}$ and $\mathcal{M}_\mathsf{v}$ are sets of commands used for authentication queries and verification queries, respectively. A may adaptively issue the following queries to C.

Key-Generation Query. Upon a query $\mathsf{id} \in \mathcal{I}$ from A, C adds id to $\mathcal{D}$ and generates $\mathsf{vk_{id}} \leftarrow \mathsf{Join}(\mathsf{ak}, \mathsf{id})$. Note that A obtains nothing, and that A is allowed to make this query at most N times.

Corruption Query. Upon a query $\mathsf{id} \in \mathcal{D}$ from A, C adds id to $\mathcal{W}$, and returns $\mathsf{vk_{id}}$ to A.

Authentication Query. Upon a query $(\mathsf{m}, \mathcal{S}) \in \mathcal{M} \times 2^{\mathcal{D}}_{\leq \ell}$ from A, if m has not used for any verification queries (i.e., $\mathsf{m} \notin \mathcal{M}_\mathsf{v}$), C returns $\mathsf{cmd}_\mathcal{S} \leftarrow \mathsf{Auth}(\mathsf{ak}, \mathsf{m}, \mathcal{S})$ to A, and adds m to $\mathcal{M}_\mathsf{a}$. Otherwise, it returns $\perp$.

Verification Query. Upon a query $(\mathsf{m}, \mathcal{S}, \mathsf{cmd}_\mathcal{S}) \in \mathcal{M} \times 2^{\mathcal{D}}_{\leq \ell} \times \mathcal{T}$ from A, C runs $\mathsf{Vrfy}(\mathsf{vk_{id}}, \mathsf{cmd}_\mathcal{S})$, adds m to $\mathcal{M}_\mathsf{v}$, and returns its output to A. If there exists at least one device's index $\mathsf{id} \in \mathcal{S}$ such that the following conditions hold, C sets $\mathsf{flag} := 1$.

1) The output of Vrfy is the same as m. Namely, $\mathsf{Vrfy}(\mathsf{vk_{id}}, \mathsf{cmd}_\mathcal{S}) = \mathsf{m}$ holds.
2) The device is not corrupted. Namely, $\mathsf{id} \notin \mathcal{W}$ holds.
3) A has not created any authentication queries for m, i.e., $\mathsf{m} \notin \mathcal{M}_\mathsf{a}$.

A is restricted to issue this query only once.

At some point (right after the verification query without loss of generality), A terminates the experiment, and C sets flag as the output of $\mathsf{Exp}^{\mathsf{CMA}}_{\Pi,\mathsf{A}}(\kappa, N, \ell)$. A's advantage is defined by $\mathsf{Adv}^{\mathsf{CMA}}_{\Pi,\mathsf{A}}(\kappa, N, \ell) := \Pr[\mathsf{Exp}^{\mathsf{CMA}}_{\Pi,\mathsf{A}}(\kappa, N, \ell) = 1]$.

Definition 1 (Unforgeability): Let Π be an ABA scheme. We say Π is UF-CMA *secure if for any PPTA* A*, it holds that* $\mathsf{Adv}^{\mathsf{CMA}}_{\Pi,\mathsf{A}}(\kappa, N, \ell) < \mathsf{negl}(\kappa)$ *for all sufficiently-large $\kappa \in \mathbb{N}$, all $N \in \mathbb{N}$, and all ℓ $(\leq N)$.*

*Remark 2 (On the Restriction of the Number of Verification Queries): We can also consider unforgeability against chosen message and verification attacks (*UF-CMVA*), which allows the unlimited numbers of verification queries. It can be viewed as an analogy of a similar notion in traditional MAC [41], though we do not deal with it in this paper.*

D. ANONYMITY

Anonymity is also a fundamental security notion in ABA. Specifically, we consider two kinds of anonymity notions, called ANO-CMA and ANO-eq-CMA security. Roughly speaking, ABA is anonymous if an adversary who corrupts some IoT devices and observes an authenticated command obtains no information on which devices are designated. UF-CMA-secure ABA is said to be ANO-CMA secure if an authenticated command $\mathsf{cmd}_\mathcal{S}$ does not tell any information on the corresponding privileged set $\mathcal{S}$ except for corrupted devices. ANO-eq-CMA security is a relaxed version of ANO-CMA security; if the number of devices in $\mathcal{S}$ is leaked (but their identifiers themselves are still hidden), ABA is said to is ANO-eq-CMA secure.

Formally, we define them based on anonymous broadcast encryption [14], [16] and privacy-preserving MAC [42]. Specifically, for ANO-CMA-security, we consider an experiment $\mathsf{Exp}^{\mathsf{ANO}}_{\Pi,\mathsf{A}}(\kappa, N, \ell)$ between a challenger C and any PPTA A.

$\boxed{\mathsf{Exp}^{\mathsf{ANO}}_{\Pi,\mathsf{A}}(\kappa, N, \ell)}$ C runs $\mathsf{Setup}(1^\kappa, N, \ell)$ to get ak. Let $\mathcal{W}$, $\mathcal{M}_\mathsf{a}$ be empty sets, ctr be a counter initialized as zero. $\mathcal{W}$ and $\mathcal{M}_\mathsf{a}$ are the same as in unforgeability. A may adaptively issue the following queries to C.

Key-Generation Query and Corruption Query. Same as the unforgeability experiment $\mathsf{Exp}^{\mathsf{CMA}}_{\Pi,\mathsf{A}}(\kappa, N, \ell)$.

Authentication Query. Upon a query $(\mathsf{m}, \mathcal{S}) \in \mathcal{M} \times 2^{\mathcal{D}}_{\leq \ell}$ from A, C returns $\mathsf{cmd}_\mathcal{S} \leftarrow \mathsf{Auth}(\mathsf{ak}, \mathsf{m}, \mathcal{S})$ to A, and adds m to $\mathcal{M}_\mathsf{a}$.

Challenge Query. Upon a query $(\mathsf{m}, \mathcal{S}_0, \mathcal{S}_1) \in \mathcal{M} \times (2^{\mathcal{D}}_{\leq \ell})^2$ from A, C randomly chooses $b \in \{0, 1\}$ and returns $\mathsf{cmd}_{\mathcal{S}_b} \leftarrow \mathsf{Auth}(\mathsf{ak}, \mathsf{m}, \mathcal{S}_b)$ to A. A is allowed to make this query only once under the following restriction.

(i) $(\mathcal{S}_0 \triangle \mathcal{S}_1) \cap \mathcal{W} = \emptyset$.
(ii) A has not created any authentication queries for m, i.e., $\mathsf{m} \notin \mathcal{M}_\mathsf{a}$.

The former restriction is necessary to avoid trivial attacks: if there exists at least one $\mathsf{id}^\star \in ((\mathcal{S}_0 \triangle \mathcal{S}_1) \cap \mathcal{W})$, A can distinguish $\mathcal{S}_0$ and $\mathcal{S}_1$ with probability one depending on the output of $\mathsf{Vrfy}(\mathsf{vk}_{\mathsf{id}^\star}, \mathsf{cmd}_{\mathcal{S}_b})$. The latter restriction is the same as the one in the unforegeability experiment. Therefore, after the challenge query, A is not allowed to make any authentication query for m.

At some point, A outputs b'. If $b' = b$, C then sets 1 as the output of $\mathsf{Exp}^{\mathsf{ANO}}_{\Pi,\mathsf{A}}(\kappa, N, \ell)$. Otherwise, C then sets 0. C terminates the experiment.

We can also define ANO-eq-security with an experiment $\mathsf{Exp}^{\mathsf{ANO\text{-}eq}}_{\Pi,\mathsf{A}}(\kappa, N, \ell)$, which is the same as $\mathsf{Exp}^{\mathsf{ANO}}_{\Pi,\mathsf{A}}(\kappa, N, \ell)$ except for the following additional condition of the restriction during the challenge query: (iii) $|\mathcal{S}_0| = |\mathcal{S}_1|$.

Definition 2 (Anonymity): Let Π be a UF-CMA*-secure ABA. We say Π is* X-CMA *secure (*$\mathsf{X} \in \{\mathsf{ANO}, \mathsf{ANO\text{-}eq}\}$*)*

if for any PPTA A*, it holds that* $|\mathsf{Adv}^{\mathsf{X}}_{\Pi,\mathsf{A}}(\kappa, N, \ell) - 1/2| < \mathsf{negl}(\kappa)$ *for all sufficiently-large* $\kappa \in \mathbb{N}$*, all* $N \in \mathbb{N}$*, and all* ℓ $(\leq N)$*, where* $\mathsf{Adv}^{\mathsf{X}}_{\Pi,\mathsf{A}}(\kappa, N, \ell) := |\Pr[\mathsf{Exp}^{\mathsf{X}}_{\Pi,\mathsf{A}}(\kappa, N, \ell) = 1] - 1/2|$.

From theoretical perspective rather than practical one, we can consider stronger notions called strong anonymity, while there is only one kind of anonymity definitions in the context of anonymous broadcast encryption. This difference depends on algorithms we consider; both deterministic and probabilistic authentication algorithms are employed in this work, whereas in general, only probabilistic encryption algorithms are considered in the encryption setting.

We remove the condition (ii) of the restriction for authentication queries in the above experiment $\mathsf{Exp}^{\mathsf{ANO}}_{\Pi,\mathsf{A}}(\kappa, N, \ell)$ to define sANO security. Analogously, we can also define $\mathsf{sANO\text{-}eq}$ security. It is clear that sANO security is stronger than ANO security. The same holds for $\mathsf{sANO\text{-}eq}$ security.

Definition 3 (Strong Anonymity): Let Π *be a* $\mathsf{UF\text{-}CMA}$*-secure ABA. We say* Π *is* X-CMA *secure* ($\mathsf{X} \in \{\mathsf{sANO}, \mathsf{sANO\text{-}eq}\}$) *if for any PPTA* A*, it holds* $|\mathsf{Adv}^{\mathsf{X}}_{\Pi,\mathsf{A}}(\kappa, N, \ell) - 1/2| < \mathsf{negl}(\kappa)$ *for all sufficiently-large* $\kappa \in \mathbb{N}$*, all* $N \in \mathbb{N}$*, and all* ℓ $(\leq N)$*, where* $\mathsf{Adv}^{\mathsf{X}}_{\Pi,\mathsf{A}}(\kappa, N, \ell) := |\Pr[\mathsf{Exp}^{\mathsf{X}}_{\Pi,\mathsf{A}}(\kappa, N, \ell) = 1] - 1/2|$.

Remark 3 (Achievability of Strong Anonymity): sANO *security tells us that it can only be achieved by probabilistic ABA schemes, where probabilistic ABA means that* Auth *is probabilistic (i.e., if we rule out deterministic* Auth *algorithm). Indeed, we can show an attack against any deterministic ABA scheme with* sANO *or* $\mathsf{sANO\text{-}eq}$ *security as follows:* A *makes an authentication query* $(\mathsf{m}, \mathcal{S}_0)$ *to get* $\mathsf{cmd}_{\mathcal{S}_0}$ *and then makes a challenge query* $(\mathsf{m}, \mathcal{S}_0, \mathcal{S}_1)$ *to gets* $\mathsf{cmd}_{\mathcal{S}_b}$*. If* $\mathsf{cmd}_{\mathcal{S}_0} = \mathsf{cmd}_{\mathcal{S}_b}$*, then* A *outputs* $b' := 0$*; otherwise* A *outputs* $b' := 1$*. We can see that the stronger definition can be viewed as a combination of the anonymity definition in anonymous broadcast encryption [14], [16] and confidentiality called* $\mathsf{IND\text{-}CMA}$ *security in probabilistic MACs [43] (also see Section V-B for the latter).*

IV. CONSTRUCTIONS

We propose two concrete constructions of ABA schemes. Taking into account the requirement 4, we aim to show constructions that meets the following properties.

- **Lightweight**: Due to the framework of our remote control system, we keep the information size and computational costs for each IoT device as small as possible. Specifically, we only use MACs and PRFs, which are well-known symmetric-key primitives, for small verification keys and efficient Vrfy algorithms.
- **Applicability**: Our constructions are fully black-box and simple combinations of well-known cryptographic primitives (i.e., MACs and PRFs) with rigorous security proofs. Therefore, one can easily implement our ABA schemes and employ not only HMAC and AES but also future lightweight MACs and PRFs for the underlying ones.

Before going into the constructions, we would like to discuss achievable lower bounds on efficiency in ABA schemes. Kiayias and Samari [16] showed that lower bounds on ciphertext sizes in ANO-secure and $\mathsf{ANO\text{-}eq}$-secure anonymous broadcast encryption schemes are $\Omega(n \cdot \kappa)$-bits and $\Omega(|\mathcal{S}| \cdot \kappa)$-bits, respectively, where n is the number of all users. Therefore, we expect that ANO-secure and $\mathsf{ANO\text{-}eq}$-secure ABA schemes have similar lower bounds on the size of authenticated commands, and leave deriving lower bounds as our future works. Indeed, our constructions below achieve $\Theta(n \cdot \kappa)$-bits and $\Theta(|\mathcal{S}| \cdot \kappa)$-bits for ANO-secure and $\mathsf{ANO\text{-}eq}$-secure ABA schemes, respectively.

A. BUILDING BLOCKS

1) MESSAGE AUTHENTICATION CODE (MAC)

MAC $\Pi_{\text{mac}} = (\mathsf{MAC.Gen}, \mathsf{MAC.Auth}, \mathsf{MAC.Vrfy})$ is defined as follows.

1) $\mathsf{MAC.Gen}(1^\kappa) \rightarrow \mathsf{K}$: a probabilistic algorithm for setup. It takes a security parameter 1^κ as input, and outputs a secret key K.
2) $\mathsf{MAC.Auth}(\mathsf{K}, \mathsf{m}) \rightarrow \tau$: an algorithm for generating authentication tags. It takes K and a message $\mathsf{m} \in \mathcal{M}$ as input, and outputs an authentication tag τ.
3) $\mathsf{MAC.Vrfy}(\mathsf{K}, \mathsf{m}, \tau) \rightarrow \top / \bot$: a deterministic algorithm for verification. It takes K, m, and τ as input, and outputs $\top$, which indicates "acceptance", or $\bot$, which indicate "rejection."

Π_{mac} meets the following correctness: for all $\kappa \in \mathbb{N}$, all $\mathsf{K} \leftarrow \mathsf{MAC.Gen}(1^\kappa)$ and all $\mathsf{m} \in \mathcal{M}$, it holds $\mathsf{MAC.Vrfy}(\mathsf{K}, \mathsf{MAC.Auth}(\mathsf{K}, \mathsf{m})) \rightarrow \mathsf{m}$ with overwhelming probability.

MAC provides the following unforgeability. Specifically, we consider the following experiment $\mathsf{Exp}^{\mathsf{UF}}_{\Pi_{\text{mac}},\mathsf{A}}(\kappa)$ between a challenger C and any PPTA A.

$\boxed{\mathsf{Exp}^{\mathsf{UF}}_{\Pi_{\text{mac}},\mathsf{A}}(\kappa)}$ C runs $\mathsf{MAC.Gen}(1^\kappa)$ to gets K. Let $\widetilde{\mathcal{M}}$ be an empty set, which is a set of message used for authentication queries defined below. A is allowed to make an authentication query $\mathsf{m} \in \mathcal{M}$ and send C it. Receiving the query m, C returns $\tau \leftarrow \mathsf{MAC.Auth}(\mathsf{K}, \mathsf{m})$ to A, and adds m to $\widetilde{\mathcal{M}}$. At the end of the experiment, A makes a verification query (m', τ') and sends C it. C sets 1 as the output of $\mathsf{Exp}^{\mathsf{UF}}_{\Pi_{\text{mac}},\mathsf{A}}(\kappa)$ if the output is $\mathsf{MAC.Vrfy}(\mathsf{K}, \mathsf{m}', \tau') \rightarrow \top$ and $\mathsf{m}' \notin \widetilde{\mathcal{M}}$ holds; sets 0 otherwise. Let $\mathsf{Adv}^{\mathsf{UF}}_{\Pi_{\text{mac}},\mathsf{A}}(\kappa) := \Pr[\mathsf{Exp}^{\mathsf{UF}}_{\Pi_{\text{mac}},\mathsf{A}}(\kappa) = 1]$ be A's advantage.

Definition 4 (Unforgeability for MAC): Let Π_{mac} *be a MAC. We say* Π_{mac} *is* $\mathsf{UF\text{-}CMA}$*-secure if for any PPTA* A*, it holds that* $\mathsf{Adv}^{\mathsf{UF}}_{\Pi_{\text{mac}},\mathsf{A}}(\kappa) < \mathsf{negl}(\kappa)$ *for all sufficiently-large* $\kappa \in \mathbb{N}$.

We next describe confidentiality of MAC as an additional security requirement. Specifically, we consider the following experiment $\mathsf{Exp}^{\mathsf{PP}}_{\Pi_{\text{mac}},\mathsf{A}}(\kappa)$ between a challenger C and any PPTA A.

$\boxed{\mathsf{Exp}^{\mathsf{PP}}_{\Pi_{\mathrm{mac}},\mathsf{A}}(\kappa)}$ C randomly chooses $b \in \{0, 1\}$, and runs $\mathsf{MAC.Gen}(1^\kappa)$ to get K. Let $\mathcal{M}_0$ and $\mathcal{M}_1$ be empty sets. A may make an authentication query $(\mathsf{m}_0, \mathsf{m}_1) \in \mathcal{M}^2$ and sends C it. C returns $\tau \leftarrow \mathsf{MAC.Auth}(\mathsf{K}, \mathsf{m}_b)$ to A, and adds m_0 and m_1 to $\mathcal{M}_0$ and $\mathcal{M}_1$, respectively. At some point, A outputs b'. If the following conditions hold, C sets 1 as the output of $\mathsf{Exp}^{\mathsf{PP}}_{\Pi_{\mathrm{mac}},\mathsf{A}}(\kappa)$:

- All elements of $\mathcal{M}_0$ are distinct.
- All elements of $\mathcal{M}_1$ are distinct.
- $b' = b$ holds.

Otherwise, C sets 0 as the output of the experiment. The first two conditions are necessary to avoid trivial attacks. For example, if A is allowed to issue two queries $(\mathsf{m}, \mathsf{m}')$ and $(\mathsf{m}, \mathsf{m}'')$, A can easily distinguish b.

Definition 5 (Confidentiality for MAC [42]): Let Π_{mac} be UF-CMA*-secure MAC. We say Π_{mac} satisfies confidentiality, or is Privacy-Preserving MAC (PP-MAC) if for any PPTA* A*, it holds that $\mathsf{Adv}^{\mathsf{PP}}_{\Pi_{\mathrm{mac}},\mathsf{A}}(\kappa) < \mathsf{negl}(\kappa)$ for all sufficiently-large $\kappa \in \mathbb{N}$, where $\mathsf{Adv}^{\mathsf{PP}}_{\Pi_{\mathrm{mac}},\mathsf{A}}(\kappa) := |\Pr[\mathsf{Exp}^{\mathsf{PP}}_{\Pi_{\mathrm{mac}},\mathsf{A}}(\kappa) = 1] - 1/2|$.*

2) PSEUDORANDOM FUNCTION (PRF)

PRF $\Pi_{\mathrm{prf}} = (\mathsf{PRF.Gen}, \mathsf{PRF.Eval})$ is defined below.

1) $\mathsf{PRF.Gen}(1^\kappa) \rightarrow \mathsf{k}$: a probabilistic algorithm for setup. It takes a security parameter 1^κ, and outputs a secret key k.
2) $\mathsf{PRF.Eval}(\mathsf{k}, x) \rightarrow y$: a deterministic algorithm for evaluation. It takes k and a value $x \in \mathcal{D}_{\Pi_{\mathrm{prf}}}$ as input, and outputs $y \in \mathcal{R}_{\Pi_{\mathrm{prf}}}$.

Π_{prf} satisfies the following pseudorandomness. Specifically, we consider the following experiment $\mathsf{Exp}^{\mathsf{PRF}}_{\Pi_{\mathrm{prf}},\mathsf{A}}(\kappa)$ between a challenger C and any PPTA A.

$\boxed{\mathsf{Exp}^{\mathsf{PRF}}_{\Pi_{\mathrm{prf}},\mathsf{A}}(\kappa)}$ C chooses a random bit $b \in \{0, 1\}$, and runs $\mathsf{PRF.Gen}(1^\kappa)$ to get k. Let $\mathcal{X}$ be an empty set. A is allowed to repeatedly make an evaluation query $x \in \mathcal{D}_{\Pi_{\mathrm{prf}}}$ and sends C it. If $b = 0$, C returns $y \leftarrow \mathsf{PRF.Eval}(\mathsf{k}, x^\star)$ to A. Otherwise, C randomly chooses y from $\mathcal{R}_{\Pi_{\mathrm{prf}}}$ and returns it to A. Finally, A outputs b'. If $b' = b$, C sets 1 as the output of $\mathsf{Exp}^{\mathsf{PRF}}_{\Pi_{\mathrm{prf}},\mathsf{A}}(\kappa)$; otherwise, C sets 0. A's advantage is defined by $\mathsf{Adv}^{\mathsf{PRF}}_{\Pi_{\mathrm{prf}},\mathsf{A}}(\kappa) := |\Pr[\mathsf{Exp}^{\mathsf{PRF}}_{\Pi_{\mathrm{prf}},\mathsf{A}}(\kappa) = 1] - 1/2|$.

Definition 6 (Pseudorandom Functions): We say Π_{prf} is a pseudorandom function (PRF) if for any PPTA A*, it holds that $\mathsf{Adv}^{\mathsf{PRF}}_{\Pi_{\mathrm{prf}},\mathsf{A}}(\kappa) < \mathsf{negl}(\kappa)$ for all sufficiently-large $\kappa \in \mathbb{N}$.*

B. UF-CMA-*SECURE AND* ANO-*SECURE CONSTRUCTION*

We construct an ABA scheme $\Pi = (\mathsf{Setup}, \mathsf{Join}, \mathsf{Auth}, \mathsf{Vrfy})$ from any MAC Π_{mac} and any PRF Π_{prf}, where:

- $\Pi_{\mathrm{mac}} = (\mathsf{MAC.Gen}, \mathsf{MAC.Auth}, \mathsf{MAC.Vrfy})$;
- $\Pi_{\mathrm{prf}} = (\mathsf{PRF.Gen}, \mathsf{PRF.Eval})$ such that $\mathsf{PRF.Eval}_{\mathsf{k}} : \mathcal{D}_{\Pi_{\mathrm{prf}}} \rightarrow \mathcal{R}_{\Pi_{\mathrm{prf}}}$, where $\mathsf{k} \leftarrow \mathsf{PRF.Gen}(1^\kappa)$;

$\boxed{\mathsf{Setup}(1^\kappa, N, N) \rightarrow \mathsf{ak}}$ Run $\mathsf{k} \leftarrow \mathsf{PRF.Gen}(1^\kappa)$ and output $\mathsf{ak} := \mathsf{k}$.

$\boxed{\mathsf{Join}(\mathsf{ak}, \mathsf{id}) \rightarrow \mathsf{vk}_{\mathsf{id}}}$ Run $r_{\mathsf{id}} \leftarrow \mathsf{PRF.Eval}(\mathsf{k}, \mathsf{id})$. Compute $\mathsf{K}_{\mathsf{id}} \leftarrow \mathsf{MAC.Gen}(1^\kappa; r_{\mathsf{id}})$ and output $\mathsf{vk}_{\mathsf{id}} := \mathsf{K}_{\mathsf{id}}$.

$\boxed{\mathsf{Auth}(\mathsf{ak}, \mathsf{m}, \mathcal{S}) \rightarrow \mathsf{cmd}_{\mathcal{S}}}$ Let $n := |\mathcal{D}|$ be the number of devices currently involved in the protocol, and suppose that $\mathsf{vk}_{\mathsf{id}_1}, \ldots, \mathsf{vk}_{\mathsf{id}_n}$ are generated by Join. For all $j \in [n]$, run $\mathsf{Join}(\mathsf{ak}, \mathsf{id}_j)$ to get $\mathsf{K}_{\mathsf{id}_j}$, and then compute the following:

$$\begin{cases} \tau_j \leftarrow \mathsf{MAC.Auth}(\mathsf{K}_{\mathsf{id}_j}, 1\|\mathsf{m}), & \text{if } \mathsf{id}_j \in \mathcal{S}, \\ \tau_j \leftarrow \mathsf{MAC.Auth}(\mathsf{K}_{\mathsf{id}_j}, 0\|\mathsf{m}), & \text{if } \mathsf{id}_j \notin \mathcal{S}. \end{cases}$$

Output $\mathsf{cmd}_{\mathcal{S}} := (\mathsf{m}, \mathsf{id}_1\|\tau_1, \ldots, \mathsf{id}_n\|\tau_n)$.

$\boxed{\mathsf{Vrfy}(\mathsf{vk}_{\mathsf{id}}, \mathsf{cmd}_{\mathcal{S}}) \rightarrow \mathsf{m}/\bot}$ Parse $\mathsf{vk}_{\mathsf{id}}$ and $\mathsf{cmd}_{\mathcal{S}}$ as K_{id} and $(\mathsf{m}, \mathsf{id}_1\|\tau_1, \ldots, \mathsf{id}_n\|\tau_n)$, respectively. Find j such that $\mathsf{id} = \mathsf{id}_j$. Run $\mathsf{MAC.Vrfy}(\mathsf{K}_{\mathsf{id}}, 1\|\mathsf{m}, \tau_j)$ and return m if the output is $\top$; return $\bot$ otherwise.

Theorem 1: If Π_{mac} is a PP-MAC and Π_{prf} is a PRF, the construction of Π described above is UF-CMA*-secure and* ANO-CMA*-secure.*

Proof: First, we show the proposed construction is UF-CMA-secure. Namely, we show that if there exists a PPTA A that breaks the UF-CMA security, then there exists a PPTA B that breaks the security of the PRF or a PPTA F that breaks the unforgeability of the MAC. For any PPTA A, we consider the following game sequences.

- G_0: This is exactly the same as $\mathsf{Exp}^{\mathsf{CMA}}_{\Pi,\mathsf{A}}(\kappa, N, N)$. In the following, we arbitrarily fix κ and $N = \mathsf{poly}(\kappa)$ for some polynomial $\mathsf{poly}(\cdot)$.
- G_1: This is the same as G_0 except that all outputs of Π_{prf} are directly and uniformly chosen from the range $\mathcal{R}_{\Pi_{\mathrm{prf}}}$ of Π_{prf}. Moreover, the challenger C randomly guesses an index $i^\star \in [N]$ at the beginning of the game.
- G_2: This is the same as G_1 except that C requires the following condition for $\mathsf{flag} = 1$ in addition to 1) – 3):
  4) $\mathsf{id}_{i^\star} = \min \mathcal{S}$ holds.

For $i \in \{0, 1, 2\}$, let S_i be an event that $\mathsf{flag} = 1$ occurs in G_i. We show that $\mathsf{Adv}^{\mathsf{CMA}}_{\Pi,\mathsf{A}}(\kappa, N, N) \leq |\Pr[\mathsf{S}_0] - \Pr[\mathsf{S}_1]| + \Pr[\mathsf{S}_1] \leq 2\mathsf{Adv}^{\mathsf{PRF}}_{\Pi_{\mathrm{prf}},\mathsf{B}}(\kappa) + N \cdot \mathsf{Adv}^{\mathsf{UF}}_{\Pi_{\mathrm{mac}},\mathsf{F}}(\kappa)$.

Lemma 1: $\left|\Pr[\mathsf{S}_0] - \Pr[\mathsf{S}_1]\right| \leq 2\mathsf{Adv}^{\mathsf{PRF}}_{\Pi_{\mathrm{prf}},\mathsf{B}}(\kappa)$.

Proof: We construct a simulator B that wins $\mathsf{Exp}^{\mathsf{PRF}}_{\Pi_{\mathrm{prf}},\mathsf{B}}(\kappa)$ by using A that breaks G_0 or G_1. B simulates G_0 or G_1 depending on a bit b chosen by the challenger C of $\mathsf{Exp}^{\mathsf{PRF}}_{\Pi_{\mathrm{prf}},\mathsf{B}}(\kappa)$. Specifically, when receiving the key-generation query id from A, B makes evaluation queries id to obtain r_{id}, respectively, and uses them to generate $\mathsf{vk}_{\mathsf{id}} := \mathsf{K}_{\mathsf{id}}$. If $b = 0$, then r_{id} is the outputs of Π_{prf}; otherwise, they are uniformly chosen from $\mathcal{R}_{\Pi_{\mathrm{prf}}}$. Namely, B simulates G_0 if $b = 0$ or G_1 otherwise. Finally, B outputs $b' = 0$ if A submits $(\mathsf{m}, \mathcal{S}, \mathsf{cmd}_{\mathcal{S}})$ that satisfies the conditions for $\mathsf{flag} := 1$; $b' = 1$ otherwise. Hence, we have $2 \cdot \mathsf{Adv}^{\mathsf{PRF}}_{\Pi_{\mathrm{prf}},\mathsf{B}}(\kappa) = |\Pr[b' = 0 \mid b = 0] - \Pr[b' = 0 \mid b = 1]| = |\Pr[\mathsf{S}_0] - \Pr[\mathsf{S}_1]|$. Note that although the index $i^\star$ is guessed in G_1, it does not affect A's view. □

Lemma 2: $\Pr[\mathsf{S}_1] = N \cdot \Pr[\mathsf{S}_2]$.

Proof: For $i \in [N]$, let id_i be an i-th key-generation query. Let G be an event that for an verification query

$(\mathsf{m}, \mathcal{S}, \mathsf{cmd}_{\mathcal{S}})$, $\mathsf{id}_{i^\star}$ satisfies $\mathsf{id}_{i^\star} = \min \mathcal{S}$ and the conditions for $\mathsf{flag} = 1$ hold (see the description of G_2). The difference between G_1 and G_2 is that A only wins when C's guess is correct in G_2. Therefore, we have

$$\begin{aligned}\Pr[\mathsf{S}_2] &= \Pr[\mathsf{S}_2 \wedge \mathsf{G}] + \Pr[\mathsf{S}_2 \wedge \neg\mathsf{G}] \\ &= \Pr[\mathsf{S}_1 \wedge \mathsf{G}] + \Pr[\neg\mathsf{G}]\Pr[\mathsf{S}_2 \mid \neg\mathsf{G}] \qquad (1)\\ &= \Pr[\mathsf{S}_1]\Pr[\mathsf{G} \mid \mathsf{S}_1] \qquad (2)\\ &= \frac{1}{N}\Pr[\mathsf{S}_1], \qquad (3)\end{aligned}$$

where (1) follows from $\Pr[\mathsf{S}_2 \wedge \mathsf{G}] = \Pr[\mathsf{S}_1 \wedge \mathsf{G}]$, (2) follows from $\Pr[\mathsf{S}_2 \mid \neg\mathsf{G}] = 0$, and (3) follows from $\Pr[\mathsf{G} \mid \mathsf{S}_1] = 1/N$. □

Lemma 3: $\Pr[\mathsf{S}_2] = \mathsf{Adv}^{\mathsf{UF}}_{\Pi_{\mathrm{mac}},\mathsf{F}}(\kappa)$.

Proof: We construct a simulator F that wins $\mathsf{Exp}^{\mathsf{UF}}_{\Pi_{\mathrm{mac}},\mathsf{F}}(\kappa)$ by using A that breaks G_2. B randomly guesses $i^\star \in [N]$ and implicitly sets $\mathsf{vk}_{\mathsf{id}_{i^\star}} := \mathcal{K}_{\mathsf{id}_{i^\star}}$, where $\mathcal{K}_{\mathsf{id}_{i^\star}}$ is a MAC key generated by the challenger C of $\mathsf{Exp}^{\mathsf{UF}}_{\Pi_{\mathrm{mac}},\mathsf{F}}(\kappa)$. For $i \in [N] \setminus \{ i^\star \}$, B runs Join to create $\mathsf{vk}_{\mathsf{id}_i}$ for an i-th key-generation query id_i. When receiving an authentication query $(\mathsf{m}, \mathcal{S})$ from A, B computes $\mathsf{cmd}_{\mathcal{S}}$ as follows: for any $\mathsf{id}_j \in \mathcal{D} \setminus \{ \mathsf{id}_{i^\star} \}$, B generates τ_j as in the Auth algorithm. If $\mathsf{id}_{i^\star} \in \mathcal{S}$, B makes an authentication query $1\|\mathsf{m}$ to get $\tau_{i^\star}$; otherwise, B makes an authentication query $0\|\mathsf{m}$ to get $\tau_{i^\star}$. Note that if $\mathsf{id}_{i^\star} \notin \mathcal{D}$ (i.e., before issuing the $i^\star$-th key-generation query), B just generates $\mathsf{cmd}_{\mathcal{S}}$ as in the Auth algorithm. Suppose that G occurs when receiving a verification query $(\mathsf{m}, \mathcal{S}, \mathsf{cmd}_{\mathcal{S}})$ from A. It means that $(1\|\mathsf{m}, \tau_{i^\star})$ is a message/authenticator pair that breaks the unforgeability of the MAC. Therefore, we have $\Pr[\mathsf{S}_2] = \mathsf{Adv}^{\mathsf{UF}}_{\Pi_{\mathrm{mac}},\mathsf{F}}(\kappa)$. □

We next show the proposed construction is $\mathsf{ANO\text{-}CMA}$-secure. Namely, we show that if there exists a PPTA A that breaks the $\mathsf{ANO\text{-}CMA}$ security, then there exists a PPTA B that breaks the security of the PRF or a PPTA F that breaks the confidentiality of the MAC. For any PPTA A, we consider the following game sequences.

- G_0: This is exactly the same as $\mathsf{Exp}^{\mathsf{ANO}}_{\Pi,\mathsf{A}}(\kappa, N, N)$. In the following, we arbitrarily fix κ and $N = \mathsf{poly}(\kappa)$ for some polynomial $\mathsf{poly}(\cdot)$.
- G_1: This is the same as G_0 except that all outputs of Π_{prf} are directly and uniformly chosen from the range $\mathcal{R}_{\Pi_{\mathrm{prf}}}$ of Π_{prf}.
- $\mathsf{G}_{2,i}$ ($i \in [Q]$): Let Q be an upper bound of key-generation queries A can make during the game. This game is the same as $\mathsf{G}_{2,i-1}$ except that a challenge query $(\mathsf{m}, \mathcal{S}_0, \mathcal{S}_1)$, an authenticator τ_i of the i-th identifier id_i (i.e., an identifier generated by an i-th key-generation query) is generated as follows: C generates $\tau_i \leftarrow \mathsf{MAC.Auth}(\mathsf{K}_{\mathsf{id}_i}, 0\|\mathsf{m})$ if $\mathsf{id}_i \in (\mathcal{S}_0 \triangle \mathcal{S}_1) \setminus \mathcal{W}$. Note that $\mathsf{G}_{2,0} := \mathsf{G}_1$.

For $i \in \{0, 1\}$, let S_i be an event that $b' = b$ occurs in G_i. Similarly, let $\mathsf{S}_{2,i}$ be an event that $b' = b$ occurs in $\mathsf{G}_{2,i}$ for $i \in [Q]$. We show that $\mathsf{Adv}^{\mathsf{ANO}}_{\Pi,\mathsf{A}}(\kappa, N, N) \le |\Pr[\mathsf{S}_0] - \Pr[\mathsf{S}_1]| + \sum_{i=1}^{Q} |\Pr[\mathsf{S}_{2,i-1}] - \Pr[\mathsf{S}_{2,i}]| + |\Pr[\mathsf{S}_{2,Q}] - 1/2| \le 2\mathsf{Adv}^{\mathsf{PRF}}_{\Pi_{\mathrm{prf}},\mathsf{B}}(\kappa) + 2Q \cdot \mathsf{Adv}^{\mathsf{PP}}_{\Pi_{\mathrm{mac}},\mathsf{F}}(\kappa)$.

We have $\left|\Pr[\mathsf{S}_0] - \Pr[\mathsf{S}_1]\right| \le 2\mathsf{Adv}^{\mathsf{PRF}}_{\Pi_{\mathrm{prf}},\mathsf{B}}(\kappa)$ as in Lemma 1. Moreover, it obviously holds that $\left|\Pr[\mathsf{S}_{2,Q}] - \frac{1}{2}\right| = 0$ since all authenticators for all identifiers in $\mathcal{D} \setminus \mathcal{W}$ (which includes $(\mathcal{S}_0 \triangle \mathcal{S}_1) \setminus \mathcal{W}$) are generated for a message $0\|\mathsf{m}$ and A cannot distinguish $\mathcal{S}_0$ and $\mathcal{S}_1$ more than probability $1/2$. Therefore, the rest of the proof follows from the following lemma.

Lemma 4: $\left|\Pr[\mathsf{S}_{2,i-1}] - \Pr[\mathsf{S}_{2,i}]\right| \le 2\mathsf{Adv}^{\mathsf{PP}}_{\Pi_{\mathrm{mac}},\mathsf{F}}(\kappa)$ for $i \in [Q]$.

Proof: We construct a simulator F that wins $\mathsf{Exp}^{\mathsf{PP}}_{\Pi_{\mathrm{mac}},\mathsf{F}}(\kappa)$ by using A that breaks $\mathsf{G}_{2,i-1}$ or $\mathsf{G}_{2,i}$. F simulates $\mathsf{G}_{2,i-1}$ or $\mathsf{G}_{2,i}$ depending on a bit b chosen by the challenger C of $\mathsf{Exp}^{\mathsf{PP}}_{\Pi_{\mathrm{mac}},\mathsf{F}}(\kappa)$. Specifically, F implicitly sets a MAC key generated by C as $\mathsf{K}_{\mathsf{id}_i}$ at the beginning of the game, where id_i is an i-th key-generation query, and chooses a random bit $d \in \{0, 1\}$. Now, suppose that id_i is not queried as corruption queries. Then, F can respond to any key-generation queries and corruption queries. F can also respond to any authentication query $(\mathsf{m}, \mathcal{S})$ as follows: If $\mathsf{id}_i \in \mathcal{S}$, F makes an authentication query $(1\|\mathsf{m}, 1\|\mathsf{m})$ to get τ_i; otherwise, F makes an authentication query $(0\|\mathsf{m}, 0\|\mathsf{m})$. For other identifiers $\mathsf{id} \in \mathcal{D} \setminus \{ \mathsf{id}_i \}$, F makes an authenticator as in the Auth algorithm. For a challenge query $(\mathsf{m}, \mathcal{S}_0, \mathcal{S}_1)$, F computes $\mathsf{cmd}_{\mathcal{S}_d}$ as follows. If $\mathsf{id}_i \in (\mathcal{S}_0 \triangle \mathcal{S}_1) \setminus \mathcal{W}$, F makes an authentication query $(0\|\mathsf{m}, 1\|\mathsf{m})$ to get τ_i. If $b = 0$, then τ_i is generated as $\tau_i \leftarrow \mathsf{MAC.Auth}(\mathsf{K}_{\mathsf{id}_i}, 0\|\mathsf{m})$, which simulates $\mathsf{G}_{2,i}$; otherwise, $\tau_i \leftarrow \mathsf{MAC.Auth}(\mathsf{K}_{\mathsf{id}_i}, 1\|\mathsf{m})$, which simulates $\mathsf{G}_{2,i-1}$. F then generates $\tau_j \leftarrow \mathsf{MAC.Auth}(\mathsf{K}_{\mathsf{id}_j}, 1\|\mathsf{m})$ for every $\mathsf{id} \in \mathcal{S}_d \setminus \{ \mathsf{id}_i \}$ and $\tau_j \leftarrow \mathsf{MAC.Auth}(\mathsf{K}_{\mathsf{id}_j}, 0\|\mathsf{m})$ for every $\mathsf{id} \in \mathcal{D} \setminus \mathcal{S}_d$, and returns $\mathsf{cmd}_{\mathcal{S}_d}$ to A. Note that if $\mathsf{id} \notin (\mathcal{S}_0 \triangle \mathcal{S}_1) \setminus \mathcal{W}$, F just computes $\tau_i \leftarrow \mathsf{MAC.Auth}(\mathsf{K}_{\mathsf{id}_i}, 0\|\mathsf{m})$. After A submits d', F outputs $b' = 0$ if $d' = d$; $b' = 1$ otherwise. Therefore, we have $2 \cdot \mathsf{Adv}^{\mathsf{PP}}_{\Pi_{\mathrm{mac}},\mathsf{F}}(\kappa) = |\Pr[b' = 0 \mid b = 0] - \Pr[b' = 0 \mid b = 1]| = |\Pr[\mathsf{S}_{2,i} \wedge \neg\mathsf{Cor}_i] - \Pr[\mathsf{S}_{2,i-1} \wedge \neg\mathsf{Cor}_i]| = |\Pr[\mathsf{S}_{2,i}] - \Pr[\mathsf{S}_{2,i-1}]|$, where Cor_i is an event that id_i is corrupted and the last equality follows from that $\Pr[\mathsf{S}_{2,i} \wedge \mathsf{Cor}_i] = \Pr[\mathsf{S}_{2,i-1} \wedge \mathsf{Cor}_i]$. □

It completes the proof of Theorem 1. □

C. UF-CMA-SECURE AND ANO-eq-SECURE CONSTRUCTION

We next show an $\mathsf{ANO\text{-}eq}$-secure ABA scheme from the same primitives by slightly modifying the above Auth algorithm. In addition to a MAC Π_{mac} and PRF Π_{prf}, we use another PRF $\overline{\Pi}_{\mathrm{prf}} = (\overline{\mathsf{PRF.Gen}}, \overline{\mathsf{PRF.Eval}})$ such that $\overline{\mathsf{PRF.Eval}}_{\overline{\mathsf{k}}} : \mathcal{D}_{\overline{\Pi}_{\mathrm{prf}}} \to \mathcal{R}_{\overline{\Pi}_{\mathrm{prf}}}$, where $\overline{\mathsf{k}} \leftarrow \overline{\mathsf{PRF.Gen}}(1^\kappa)$, and random permutation σ, where $\mathcal{D}_{\overline{\Pi}_{\mathrm{prf}}} = \mathcal{R}_{\overline{\Pi}_{\mathrm{prf}}}$. The second construction requires the underlying MAC to be only $\mathsf{UF\text{-}CMA}$-secure, whereas the MAC in the first construction should meet both $\mathsf{UF\text{-}CMA}$ security and confidentiality.

$\boxed{\mathsf{Setup}(1^\kappa, N, N) \to \mathsf{ak}}$ It is the same as that in the first construction.

$\boxed{\mathsf{Join}(\mathsf{ak}, \mathsf{id}) \to \mathsf{vk}_{\mathsf{id}}}$ Run $r_{\mathsf{id}} \leftarrow \mathsf{PRF.Eval}(\mathsf{k}, 0\|\mathsf{id})$ and $y_{\mathsf{id}} \leftarrow \overline{\mathsf{PRF.Eval}}(\mathsf{k}, 1\|\mathsf{id})$. Compute $\mathsf{K}_{\mathsf{id}} \leftarrow \mathsf{MAC.Gen}(1^\kappa; r_{\mathsf{id}})$ and output $\mathsf{vk}_{\mathsf{id}} := (\mathsf{K}_{\mathsf{id}}, y_{\mathsf{id}})$.

$\boxed{\mathsf{Auth}(\mathsf{ak}, \mathsf{m}, \mathcal{S}) \to \mathsf{cmd}_{\mathcal{S}}}$ Suppose $\mathcal{S} = \{\mathsf{id}_1, \ldots, \mathsf{id}_k\}$. Compute $\overline{\mathsf{k}} \leftarrow \overline{\mathsf{PRF.Gen}}(1^\kappa)$ and a random permutation $\sigma : [|\mathcal{S}|] \to [|\mathcal{S}|]$. For all $j \in [k]$, run $\mathsf{Join}(\mathsf{ak}, \mathsf{id}_j)$ to get $(\mathsf{K}_{\mathsf{id}_j}, y_{\mathsf{id}_j})$, and then compute the following:

$$(1)\ \gamma_j \leftarrow \overline{\mathsf{PRF.Eval}}(\overline{\mathsf{k}}, y_{\mathsf{id}_j}),\quad (2)\ \tau_j \leftarrow \mathsf{MAC.Auth}(\mathsf{K}_{\mathsf{id}_j}, \mathsf{m}).$$

Output $\mathsf{cmd}_{\mathcal{S}} := (\mathsf{m}, \overline{\mathsf{k}}, \gamma_{\mathsf{id}_{\sigma(1)}} \| \tau_{\sigma(\mathsf{id}_{\sigma(1)})}, \ldots, \gamma_{\mathsf{id}_{\sigma(k)}} \| \tau_{\mathsf{id}_{\sigma(k)}})$.

$\boxed{\mathsf{Vrfy}(\mathsf{vk}_{\mathsf{id}}, \mathsf{cmd}_{\mathcal{S}}) \to \mathsf{m}/\bot}$ Parse $\mathsf{vk}_{\mathsf{id}}$ and $\mathsf{cmd}_{\mathcal{S}}$ as $(\mathsf{K}_{\mathsf{id}}, y_{\mathsf{id}})$ and $(\mathsf{m}, \overline{\mathsf{k}}, \gamma_1 \| \tau_1, \ldots, \gamma_{|\mathcal{S}|} \| \tau_{|\mathcal{S}|})$, respectively. Compute $\overline{\gamma}_{\mathsf{id}} \leftarrow \overline{\mathsf{PRF.Eval}}(\overline{\mathsf{k}}, y_{\mathsf{id}})$ and find j such that $\gamma_j = \overline{\gamma}_{\mathsf{id}}$. Run $\mathsf{MAC.Vrfy}(\mathsf{K}_{\mathsf{id}}, \mathsf{m}, \tau_j)$ and return m if the output is $\top$; return $\bot$ otherwise.

Theorem 2: If Π_{mac} is $\mathsf{UF\text{-}CMA}$ secure, Π_{prf} is a PRF, and σ is a random permutation, the construction of Π described above is $\mathsf{UF\text{-}CMA}$-secure and $\mathsf{ANO\text{-}eq\text{-}CMA}$-secure.

Proof: Since we can prove this theorem as in the proof of Theorem 1, we omit the proof and only describe the difference from the proof of Theorem 1. In the second construction, the order of MAC authenticators in each $\mathsf{cmd}_{\mathcal{S}}$ is randomized by the random permutation σ. The reason why we need this randomization is that even if the order is deterministic, an adversary A can easily determine b' in $\mathsf{Exp}^{\mathsf{ANO\text{-}eq}}_{\Pi,\mathsf{A}}(\kappa, N, \ell)$ as follows. Suppose that three identifiers $\mathsf{id}_1, \mathsf{id}_2, \mathsf{id}_3$, where the order of them is $\mathsf{id}_1 \le \mathsf{id}_2 \le \mathsf{id}_3$ (in some order), and that the order of MAC authenticators in each $\mathsf{cmd}_{\mathcal{S}}$ are deterministically defined by the order of identifiers. A corrupts id_2 and gets $\mathsf{vk}_{\mathsf{id}_2}$, and then submits a challenge query $(\mathsf{m}, \{\mathsf{id}_1, \mathsf{id}_2\}, \{\mathsf{id}_2, \mathsf{id}_3\})$. A receives $\mathsf{cmd}_{\mathcal{S}_b} = (\mathsf{m}, \overline{\mathsf{k}}, \gamma \| \tau, \gamma' \| \tau')$. If $\mathsf{MAC.Vrfy}(\mathsf{vk}_{\mathsf{id}_2}, \mathsf{m}, \tau) \to \top$, A determines $b' = 1$; if $\mathsf{MAC.Vrfy}(\mathsf{vk}_{\mathsf{id}_2}, \mathsf{m}, \tau') \to \top$, A determines $b' = 0$. A wins the game with probability one. Hence, we need the random permutation σ in the construction. Moreover, we do not require PP-MACs since each *MAuth* takes m as input, not $0\|\mathsf{m}$ or $1\|\mathsf{m}$. □

D. DISCUSSIONS

1) ABA SCHEMES FROM ONE-WAY FUNCTIONS

It is known that (PP-)MACs are strictly weaker than PRFs [42]. Hence, we can also obtain ABA schemes by replacing Π_{mac} with Π_{prf} in our constructions. Since PRFs and (PP-)MACs can be constructed from one-way functions [44], [45], the constructions can be viewed as ABA schemes from one-way functions. Furthermore, if we change Setup so that it runs $\mathsf{MAC.Gen}$ N times and outputs N MAC keys as ak, the constructions turn to an ABA scheme from only (PP-)MACs.

2) RELATION TO CHAFFING-AND-WINNOWING

In our second construction, each device with an identifier id can find $\gamma_{\mathsf{id}} \| \tau_{\mathsf{id}}$, which is a pair of a random string and MAC tag associated with the device, from $\mathsf{cmd}_{\mathcal{S}}$ only if the device is designated (i.e., $\mathsf{id} \in \mathcal{S}$). This can be seen that it stems from the same idea as Chaffing-and-Winnowing [46], [47], [48], which realizes an encryption scheme from authentication schemes. In Chaffing-and-Winnowing, a sender sends a lot of plaintexts with a MAC tag for a plaintext that the sender wants to send, and only a receiver can find the correct one by verifying the tag.

V. EXTENSIONS

We give discussions on how we can improve efficiency and security of our proposed constructions.

A. EXTENSIONS IN TERMS OF EFFICIENCY

We discuss how we can further improve the efficiency of our constructions.

In the first construction, authenticated commands $\mathsf{cmd}_{\mathcal{S}}$ contains all identifiers, which are used to make each identifier efficiently find the corresponding authenticator in Vrfy. if we allow the manager to maintain a counter $\mathsf{ctr}_{\mathsf{Join}}$ (initialized as zero) in ak, they can be removed from $\mathsf{cmd}_{\mathcal{S}}$. Specifically, Join increments $\mathsf{ctr}_{\mathsf{Join}}$ and outputs $\mathsf{vk}_{\mathsf{id}} := (\mathsf{ctr}_{\mathsf{Join}}, \mathsf{K}_{\mathsf{id}})$. Auth outputs $\mathsf{cmd}_{\mathcal{S}} := (\mathsf{m}, \tau_1, \ldots, \tau_n)$. Vrfy chooses i-th authenticator τ_i and runs $\mathsf{MAC.Vrfy}(\mathsf{K}_{\mathsf{id}}, 1\|\mathsf{m}, \tau_j)$, where $\mathsf{vk}_{\mathsf{id}} = (i, \mathsf{K}_{\mathsf{id}})$.

In the second construction, $\overline{\Pi}_{\mathrm{prf}}$ is used so that each identifier id_j efficiently finds their authenticator τ_j, and $\overline{\Pi}_{\mathrm{prf}}$ does not contribute any security aspects of the constructions. Specifically, each id_j runs $\mathsf{MAC.Vrfy}$ only once thanks to $\overline{\Pi}_{\mathrm{prf}}$. The reason why we adopt $\overline{\Pi}_{\mathrm{prf}}$ in the constructions is that we want to keep the construction simple (i.e,, we use only two cryptographic primitives, MACs and PRFs). We here discuss a variant of the construction by replacing $\overline{\Pi}_{\mathrm{prf}}$ with a cryptographic hash function H such as SHA-256. Specifically, in the Auth algorithm, (1) is replaced with the following: (1') $\gamma_j \leftarrow \mathsf{H}(\mathsf{m}\|y_{\mathsf{id}_j})$. Auth finally outputs $\mathsf{cmd}_{\mathcal{S}} := (\mathsf{m}, \gamma_{\mathsf{id}_{\sigma(1)}} \| \tau_{\sigma(\mathsf{id}_{\sigma(1)})}, \ldots, \gamma_{\mathsf{id}_{\sigma(|\mathcal{S}|)}} \| \tau_{\mathsf{id}_{\sigma(|\mathcal{S}|)}}))$. Vrfy computes $\overline{\gamma}_{\mathsf{id}} \leftarrow \mathsf{H}(\mathsf{m}\|y_{\mathsf{id}})$, instead of $\overline{\gamma}_{\mathsf{id}} \leftarrow \overline{\mathsf{PRF.Eval}}(\overline{\mathsf{k}}, y_{\mathsf{id}})$. This modification removes a PRF key $\overline{\mathsf{k}}$ from $\mathsf{cmd}_{\mathcal{S}}$, and improves computational costs by choosing a faster hash function.

B. EXTENSIONS IN TERMS OF SECURITY LEVELS

We discuss how we construct ABA schemes that achieve a stronger and/or additional security notions.

1) TO ACHIEVE STRONG ANONYMITY

In the both constructions, if the underlying PP-MAC has a probabilistic $\mathsf{MAC.Auth}$ algorithm [43], the resultant ABA schemes meet sANO security and $\mathsf{sANO\text{-}eq}$ security, respectively. We here consider $\mathsf{IND\text{-}CMA}$, which is a stronger notion than Def/ 5, for *probabilistic MACs*, which equips a probabilistic $\mathsf{MAC.Auth}$.

$\boxed{\mathsf{Exp}^{\mathsf{IND\text{-}MAC}}_{\Pi_{\mathrm{mac}},\mathsf{A}}(\kappa)}$ C randomly chooses $b \in \{0, 1\}$, and runs $\overline{\mathsf{MAC.Gen}}(1^\kappa)$ to get K. Let $\widehat{\mathsf{m}}$ be an arbitrary fixed message, say 0. A may make an authentication query $\mathsf{m} \in \mathcal{M}$ and sends C it. If $b = 0$, C returns an authenticator

$\tau \leftarrow \mathsf{MAC.Auth}(\mathsf{K}, \widehat{\mathsf{m}})$ for $\widehat{\mathsf{m}}$; otherwise, C returns $\tau \leftarrow \mathsf{MAC.Auth}(\mathsf{K}, \mathsf{m})$ for m. At some point, A outputs b'. If $b' = b$, C sets 1 as the output of $\mathsf{Exp}^{\mathsf{IND\text{-}MAC}}_{\Pi_{\mathrm{mac}},\mathsf{A}}(\kappa)$; otherwise, C sets 0.

Definition 7 ($\mathsf{IND\text{-}CMA}$ *for MAC [43]): A MAC* Π_{mac} *said to be* $\mathsf{IND\text{-}CMA}$*-secure if for any PPTA* A*, it holds that* $\mathsf{Adv}^{\mathsf{IND\text{-}MAC}}_{\Pi_{\mathrm{mac}},\mathsf{A}}(\kappa) < \mathsf{negl}(\kappa)$ *for all sufficiently-large* $\kappa \in \mathbb{N}$*, where* $\mathsf{Adv}^{\mathsf{IND\text{-}MAC}}_{\Pi_{\mathrm{mac}},\mathsf{A}}(\kappa) := |\Pr[\mathsf{Exp}^{\mathsf{IND\text{-}MAC}}_{\Pi_{\mathrm{mac}},\mathsf{A}}(\kappa) = 1] - 1/2|$.

It is obvious that as long as $\mathsf{MAC.Auth}$ is deterministic, A can determine b' such that $b' = b$ by issueing queries on two distinct message m, m' and checking whether or not the corresponding authenticators are equivalent.

We obtain the following corollaries, which can be proved in a similar way to Theorems 1 and 2.

Corollary 1: If Π_{mac} *is* $\mathsf{UF\text{-}CMA}$*-secure and* $\mathsf{IND\text{-}CMA}$*-secure and* Π_{prf} *is a PRF, the construction of* Π *in Section IV-C is* $\mathsf{UF\text{-}CMA}$*-secure and* $\mathsf{sANO\text{-}CMA}$*-secure.*

Corollary 2: If Π_{mac} *is* $\mathsf{UF\text{-}CMA}$*-secure and* $\mathsf{IND\text{-}CMA}$*-secure,* Π_{prf} *is a PRF, and* σ *is a random permutation, the construction of* Π *in Section IV-B is* $\mathsf{UF\text{-}CMA}$*-secure and* $\mathsf{sANO\text{-}eq\text{-}CMA}$*-secure.*

2) TO ACHIEVE CONFIDENTIALITY OF COMMANDS

We here consider confidentiality of commands, which guarantees that $\mathsf{cmd}_{\mathcal{S}}$ leaks no information on the corresponding message m. We consider an experiment $\mathsf{Exp}^{\mathsf{IND\text{-}CMA}}_{\Pi,\mathsf{A}}(\kappa, N, \ell)$ between a challenger C and an adversary A as follows.

$\boxed{\mathsf{Exp}^{\mathsf{IND\text{-}CMA}}_{\Pi,\mathsf{A}}(\kappa, N, \ell)}$ C randomly chooses $b \in \{0, 1\}$, and runs $\mathsf{Setup}(1^\kappa, N, \ell)$ to get ak. A may make an authentication query $(\mathsf{m}_0, \mathsf{m}_1, \mathcal{S}) \in \mathcal{M}^2 \times 2^{\mathcal{D}}_{\leq \ell}$ and sends C it. C returns $\mathsf{cmd}_{\mathcal{S}_b} \leftarrow \mathsf{Auth}(\mathsf{ak}, \mathsf{m}_b, \mathcal{S})$ to A. At some point, A outputs b'. If $b' = b$, C sets 1 as the output of $\mathsf{Exp}^{\mathsf{IND\text{-}CMA}}_{\Pi,\mathsf{A}}(\kappa, N, \ell)$; otherwise, C sets 0 as the output of the experiment.

Definition 8 (Confidentiality of Commands): Π *said to be* $\mathsf{IND\text{-}CMA}$*-secure, or meets confidentiality if for any PPTA* A*, it holds that* $\mathsf{Adv}^{\mathsf{IND\text{-}CMA}}_{\Pi,\mathsf{A}}(\kappa, N, \ell) < \mathsf{negl}(\kappa)$ *for all sufficiently-large* $\kappa \in \mathbb{N}$*, all* $N \in \mathbb{N}$*, and all* ℓ $(\leq N)$*, where* $\mathsf{Adv}^{\mathsf{IND\text{-}CMA}}_{\Pi,\mathsf{A}}(\kappa, N, \ell) := |\Pr[\mathsf{Exp}^{\mathsf{IND\text{-}CMA}}_{\Pi,\mathsf{A}}(\kappa, N, \ell) = 1] - 1/2|$.

We can easily modify our constructions so that they meet $\mathsf{IND\text{-}CMA}$ security by replacing the underlying MAC Π_{mac} with any authenticated encryption scheme that meets $\mathsf{IND\text{-}CPA}$ and $\mathsf{INT\text{-}PTXT}$ security.[2]

C. COUNTERMEASURES AGAINST REPLAY ATTACKS AND ITS IMPLEMENTATION

Our constructions surely meet $\mathsf{UF\text{-}CMA}$ security (Def. 1), while the definition does not allow an adversary A to use message previously queried as an authentication query for a verification query (see the third restriction of verification queries). It means that $\mathsf{UF\text{-}CMA}$ security does not provide security against replay attacks, i.e., an adversary that uses the same message for multiple queries. We emphasize that this restriction is not so strange since one can see the same restriction in the unforgeability definition of MACs. In practice, one can indeed realize the restricted situation by combining a message with a timestamp. Namely, the replay attacks can be avoided by computing $\mathsf{cmd}_{\mathcal{S}} \leftarrow \mathsf{Auth}(\mathsf{ak}, \mathtt{t}\|\mathsf{m}, \mathcal{S})$, where $\mathtt{t}$ is a timestamp. However, in general, digital signatures, which are well-known *public-key* primitives, are used to synchronize timestamps. Therefore, We here discuss how to avoid the replay attacks with only *symmetric-key* primitives. We allow the manager and each device id to maintain their own counters ctr and $\mathsf{ctr}_{\mathsf{id}}$, respectively, to detect replay attacks. Let us explain the details through the first construction as an example. Setup and Join are the same as those in the construction. Every time the manager runs Auth, the manager increments ctr and broadcasts $(\mathsf{ctr}, \mathsf{cmd}_{\mathcal{S}} := (\mathsf{m}, \mathsf{id}_1\|\tau_1, \ldots, \mathsf{id}_n\|\tau_n))$, where for each $j \in [n]$, τ_j is computed as follows:

$$\begin{cases} \tau_j \leftarrow \mathsf{MAC.Auth}(\mathsf{K}_{\mathsf{id}_j}, 1\|\mathsf{m}\|\mathsf{ctr}), & \text{if } \mathsf{id}_j \in \mathcal{S}, \\ \tau_j \leftarrow \mathsf{MAC.Auth}(\mathsf{K}_{\mathsf{id}_j}, 0\|\mathsf{m}\|\mathsf{ctr}), & \text{if } \mathsf{id}_j \notin \mathcal{S}. \end{cases}$$

As described above, each device id maintains its own counter $\mathsf{ctr}_{\mathsf{id}}$. When receiving $(\mathsf{ctr}, \mathsf{cmd}_{\mathcal{S}})$, each device checks whether $\mathsf{ctr} > \mathsf{ctr}_{\mathsf{id}}$ holds. If not, the device sets the output of Vrfy to $\bot$. Otherwise, the device runs Vrfy. If $\mathsf{MAC.Vrfy}(\mathsf{K}_{\mathsf{id}}, 1\|\mathsf{m}\|\mathsf{ctr}, \tau_j) \rightarrow \top$, it returns m; otherwise, it returns $\bot$. Finally, id updates $\mathsf{ctr}_{\mathsf{id}} := \mathsf{ctr}$ if $\mathsf{cmd}_{\mathcal{S}}$ is valid (i.e., Vrfy outputs m).

The above modification is secure against the replay attacks. Intuitively, each ctr, which is incremented each time Auth is executed, is associated with its corresponding authenticated command $\mathsf{cmd}_{\mathcal{S}}$ by $\mathsf{MAC.Auth}$. Therefore, the adversary A cannot forge any pair $(\mathsf{ctr}, \mathsf{cmd}_{\mathcal{S}})$ due to unforgeability of the underlying MAC. Moreover, each device checks if $\mathsf{ctr} > \mathsf{ctr}_{\mathsf{id}}$ holds. It ensures that a pair $(\mathsf{ctr}, \mathsf{cmd}_{\mathcal{S}})$ is a fresh one, not a reused one. Note that $\mathsf{ctr}_{\mathsf{id}}$ is updated if and only if $\mathsf{MAC.Vrfy}$ outputs $\top$, i.e., the counter ctr that the device received is valid.

In fact, implementations of our ABA schemes in the next section employ this countermeasure (without synchronizing timestamps).

VI. EXPERIMENTS

In this section, we describe experimental evaluations of the proposed schemes. First, we describe our implementation of the proposed schemes. Next, we show experimental results on a laptop PC and then discuss their consideration. Primarily, we also estimate the performance, including communication between a laptop PC and a Raspberry Pi, of the entire process over a typical network on the consideration. On the system model in Section II, the laptop PC corresponds to an operation, and the Raspberry Pi corresponds to an IoT device since the device has become popular and widely used

[2]We omit the details of authenticated encryption. See [40] for the definitions of **IND-CPA** and **INT-PTXT**.

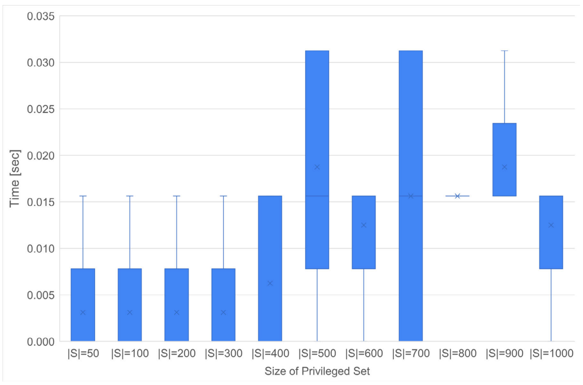

FIGURE 2. **The computational time for the Auth algorithm of UF-CMA-secure and ANO-secure construction: The box-and-whisker plot represents the maximum and minimum values and the quartiles for each size of privileged size.**

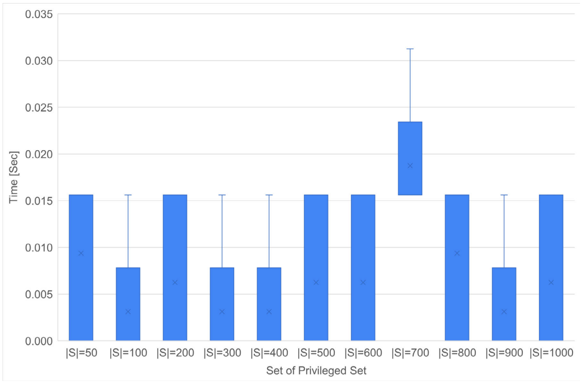

FIGURE 3. **The computational time for the Vrfy algorithm of the UF-CMA-secure and ANO-secure construction: The setting is common with Fig. 2.**

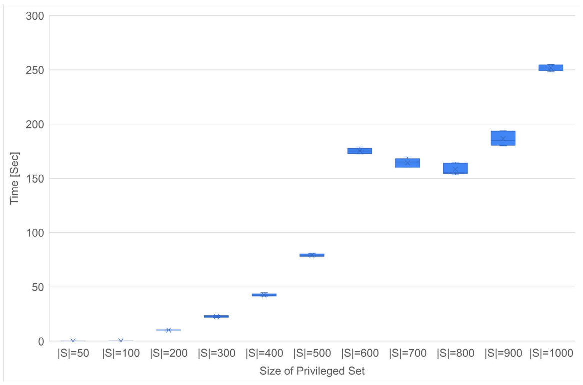

FIGURE 4. **The computational time for the Auth algorithm of the UF-CMA-secure and ANO-eq-secure construction: The setting is common with Fig. 2.**

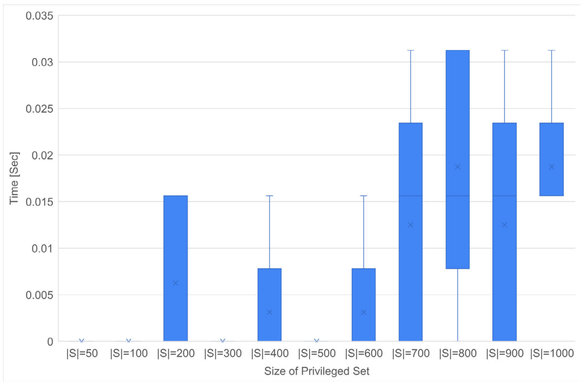

FIGURE 5. **The computational time for the Vrfy algorithm of the UF-CMA-secure and ANO-eq-secure construction: The setting is common with Fig. 2.**

by most users. The estimation thus gives us insight into remote-controlling devices in the real world.

A. IMPLEMENTATIONS

We implemented the proposed schemes in Section IV-C in the C language with the OpenSSL library version 1.1.1. Specifically, PRF and MAC were implemented via HMAC. In the ANO-eq-secure ABA scheme, the random permutation σ was implemented the Fisher-Yates algorithm [49], where random numbers were generated as AES ciphertexts. We note that the Fisher-Yates algorithm is a classic algorithm for random permutation, and the random number generation described above provides the distribution whose statistical distance is close to the uniform distribution [50]. The counter ctr to prevent replay attacks in Section V-C was also implemented. Our source code is available via GitHub as described in Code Availability for reproducibility and as a reference for subsequent works.

B. EXPERIMENTAL SETTING

We evaluate the feasibility of our ABA schemes for the remote control system described in Section II. To this end, the computational performance is measured on a laptop PC. As a part of consideration, we also measure the computational performance on Raspberry Pi3 as an IoT device in order to measture the performance on IoT environments. The environment of the laptop PC is Ubuntu 18.04.5 LTS on the Windows Subsystem for Linux over Windows 10 and is with Intel Core i7-8565U and 16 gigabytes memory. On the other hand, Raspberry Pi3 is with Raspbian GNU/Linux 10 (buster). The entire performance is then estimated by including the communication over a typical wireless network setting.

The above performance was measured by executing all the algorithms on the laptop PC memory and the Vrfy algorithms on Raspberry Pi3, where an authentication key and all verification keys are generated for every execution. Each algorithm is executed five times for performance stability. The byte length of commands to be authenticated is 32 bytes.

C. RESULT

We demonstrate the experimental results in this section. We first show the results of all the algorithms on the laptop PC and then show those of the Vrfy algorithm on the Raspberry Pi3.

1) DEPLOYMENT TO LAPTOP PC

The results on the UF-CMA-secure and ANO-secure construction for the size of privileged sets are shown in Fig. 2 and Fig. 3, and those on the UF-CMA-secure and ANO-eq-secure construction are shown in Fig. 4 and Fig. 5, respectively. Here, most computations of the proposed schemes are based on HMAC in our implementation as described in Section VI-A, and the results were measured on the laptop PC. Also, a verification key is generated on about 30 microseconds per user on the Join algorithm for the both schemes.

According to Fig. 2, the generation of authenticated commands by the Auth algorithm of the UF-CMA-secure and ANO-secure construction is about 0.015 seconds on average. The verification of these commands is common according to Fig. 3 because the exact computation is executed on the Vrfy algorithm by verification of HMAC.

On the other hand, as shown in Fig. 4, the generation of authenticated commands on the UF-CMA-secure and ANO-eq-secure construction increased significantly. Most parts of the computational time on the figure is caused by random permutation as described in detail later. Although the size of authenticated commands on the UF-CMA-secure and ANO-eq-secure construction is independent of the size of privileged sets, it also caused the trade-off between the computational time and the communication overhead. In other words, the communication overhead of the UF-CMA-secure and ANO-eq-secure construction is drastically smaller than the UF-CMA-secure and ANO-eq-secure construction, and the computational time is much heavier.

Meanwhile, the verification time of the authenticated commands on the UF-CMA-secure and ANO-eq-secure construction is almost identical to the UF-CMA-secure and ANO-secure construction, i.e., average is about 0.016 seconds. We consider that the difference between Fig. 3 and Fig. 5 depends on the valuation of PRF, which is implemented with HMAC. Namely, the UF-CMA-secure and ANO-eq-secure construction needs two HMACs per user while the UF-CMA-secure and ANO-secure construction needs a single HMAC per user.

2) DEPLOYMENT TO IoT DEVICE

We discuss the performance of the proposed schemes on a Raspberry Pi as an IoT device. In particular, on Raspberry Pi3 with Raspbian GNU/Linux 10 (buster), we measured the Vrfy algorithm of only UF-CMA-secure and ANO-secure construction for the size of privileged sets as 1000. We believe that, although the current implementation is based on HMAC, cheaper devices such as microcontrollers will be available by introducing a more lightweight authentication scheme [51] instead of HMAC. We also expect that similar performance can be obtained on the UF-CMA-secure and ANO-eq-secure construction as well because the same level performance was obtained as described in Section VI-C.

The results are shown in Fig. 6. According to the figure, the Raspberry Pi can verify the given authenticated commands

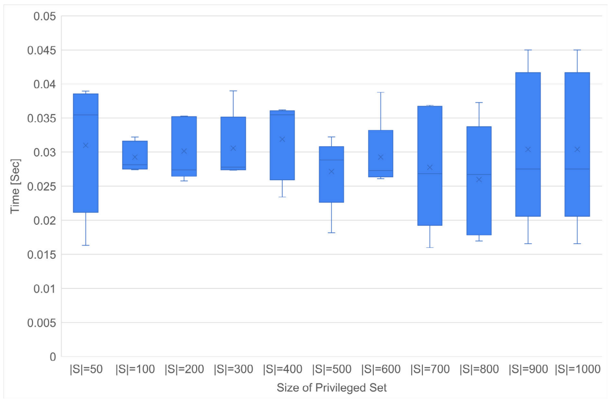

FIGURE 6. The computational time on a Raspberry Pi for the Vrfy algorithm of the UF-CMA-secure and ANO-secure construction: The setting is common with Fig. 2.

for 1000 devices on average 0.03 seconds. The performance is then estimated as 0.03 milliseconds per IoT device. For instance, authenticated commands for 100 thousand devices will be verified in three seconds.

D. CONSIDERATION

Based on the experimental results described in the previous section, we discuss the performance of the proposed schemes in detail. In particular, we first discuss the pre-processing to reduce the computational time. We further evaluate the entire performance by estimating the communication overhead over a typical wireless network environment.

1) PRE-PROCESSING FOR IMPROVING COMPUTATIONAL TIME

We discuss how the performance of the UF-CMA-secure and ANO-eq-secure construction is improved by the pre-processing. To this end, we analyze the computational time of the Auth algorithm and show the detail in Table 1.

Most parts of the computational time then depended on the random permutation. Only 0.016 seconds of the computational time for $|S| = 1000$ in Fig. 4, i.e., only about 0.006%, was for generating HMAC of authenticated commands, and the remaining time was for the random permutation. The random permutation was implemented by the Fisher-Yates algorithm [49] with unique random numbers via AES ciphertexts. These random numbers often collided when the size of privileged sets is large, and the resulting computational time became quite long due to repeating the random number generation.

We consider that the random permutation can be pre-processed independently of the generation of authenticated commands. More specifically, generating random numbers for the random permutation in advance leads to only computing HMACs for authenticated commands. The performance of the Auth algorithm would then be improved to the same level as the UF-CMA-secure and ANO-secure construction.

TABLE 1. Detail of computational time for the Auth algorithm of the UF-CMA-secure and ANO-eq-secure construction.

Process	Time [sec]
Total	266.172
HMAC	0.016
Random	266.156

We denote the entire computational time by Total, the generation of HMAC for authenticated commands by HMAC, and the random permutation by Random.

2) ENTIRE PERFORMANCE

We discuss the entire performance of the proposed schemes. As a reference value for an acceptable execution time, we consider one second as an upper bound by following the existing cryptographic protocol [52]. When the same setting as our experiments is utilized, the communication overhead of the UF-CMA-secure and ANO-secure construction is 512 bits per user in addition to 256 bits for a command. Likewise, the communication overhead of the UF-CMA-secure and ANO-eq-secure construction is 512 bits per user and 512 bits for a command and a PRF key, respectively. In the following estimation, the pre-processing described in Section VI-D1 is introduced in the implementation of the UF-CMA-secure and ANO-eq-secure construction. In other words, the computational time of the Auth algorithm becomes almost the same as the UF-CMA-secure and ANO-secure construction. The Auth algorithm is executed on a laptop PC, whereas the Vrfy algorithm is executed on a Raspberry Pi, where the performance of both machines is common with those in our experiments.

Based on the above setting, we first discuss the entire performance of the UF-CMA-secure and ANO-secure construction. If we use LoRa with its maximum transmission speed of 250 kbit/sec, the communication time from the laptop PC to the Raspberry Pi is less than 0.82 seconds until the number of devices is 400. The computational time on that scale is about 0.04 seconds for the laptop PC and the Raspberry Pi in total. Thus, the entire process is finished in about 0.86 seconds.

Next, we discuss the entire performance of the UF-CMA-secure and ANO-eq-secure construction. The communication overhead of the construction is quite limited because it is independent of the number of the whole number of devices. For instance, on the use of LoRa with $|\mathcal{S}| = 50$, the communication time from the laptop PC to the Raspberry Pi is about 0.82 seconds for 400 devices. The computational time on that scale is about 0.02 seconds for the laptop PC and the Raspberry Pi in total. The entire process on the above setting is then finished within one second.

We conclude that the proposed schemes provide acceptable executions in a practical setting. A more significant number of devices would be available for both constructions by deploying over a higher bandwidth network.

VII. CONCLUSION

In this paper, we considered a basic system for securely remote-controlling IoT devices, and discussed its requirements. We then put forward an anonymous broadcast authentication (ABA) scheme as its core cryptographic primitive. We formalized a model and security notions of ABA and showed two provably-secure constructions. The proposed schemes consist of only symmetric-key cryptographic primitives. Throughput is thus expected to be fast since our ABA schemes can be implemented with only symmetric-key cryptographic primitives, say, HMAC or AES. We also implemented the ABA constructions and evaluated their performance. Under a typical wireless network setting between a laptop PC and a Raspberry Pi as an IoT device, we confirm that the entire process of ABA could be finished within one second.

Future work would be to design a real-world application of ABA (and hence the extension of the remote control system discussed in the paper), including more resource-constrained IoT devices.

CODE AVAILABILITY

Our source code is available via GitHub (https://github.com/naotoyanai/anonymous_broadcast_authentication) for reproducibility and as a reference for subsequent works.

ACKNOWLEDGMENT

The authors would like to thank Hirokazu Kobayashi and Tatsuya Takehisa for their valuable comments.

REFERENCES

[1] Cisco, "The Internet of Things reference model," San Jose, CA, USA, Tech. Rep., 2014.

[2] X. Wang, Y. Han, V. C. M. Leung, D. Niyato, X. Yan, and X. Chen, "Convergence of edge computing and deep learning: A comprehensive survey," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 2, pp. 869–904, 2nd Quart., 2020.

[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 *Proc. USENIX Secur.*, 2017, pp. 1093–1110.

[4] M. B. M. Noor and W. H. Hassan, "Current research on Internet of Things (IoT) security: A survey," *Comput. Netw.*, vol. 148, pp. 283–294, Jan. 2019.

[5] M. A. Al-Garadi, A. Mohamed, A. K. Al-Ali, X. Du, I. Ali, and M. Guizani, "A survey of machine and deep learning methods for Internet of Things (IoT) security," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 3, pp. 1646–1685, 3rd Quart., 2020.

[6] E. Bertino and N. Islam, "Botnets and Internet of Things security," *Computer*, vol. 50, no. 2, pp. 76–79, Feb. 2017.

[7] D. Naor, M. Naor, and J. Lotspiech, "Revocation and tracing schemes for stateless receivers," in *Proc. CRYPTO*, in Lecture Notes in Computer Science, vol. 2139. Berlin, Germany: Springer, 2001, pp. 41–62.

[8] D. Boneh, C. Gentry, and B. Waters, "Collusion resistant broadcast encryption with short ciphertexts and private keys," in *Proc. CRYPTO*, in Lecture Notes in Computer Science, vol. 3621. Berlin, Germany: Springer, 2005, pp. 258–275.

[9] H. Chan and A. Perrig, "Round-efficient broadcast authentication protocols for fixed topology classes," in *Proc. IEEE Symp. Secur. Privacy*, May 2010, pp. 257–272.

[10] A. Perrig, "The BiBa one-time signature and broadcast authentication protocol," in *Proc. CCS*, 2001, pp. 28–37.

[11] A. Perrig, R. Canetti, J. D. Tygar, and D. Song, "Efficient authentication and signing of multicast streams over lossy channels," in *Proc. IEEE S&P*, May 2000, pp. 56–73.

[12] K. Shim, "BASIS: A practical multi-user broadcast authentication scheme in wireless sensor networks," *IEEE Trans. Inf. Forensics Security*, vol. 12, no. 7, pp. 1545–1554, Jul. 2017.

[13] R. Safavi-Naini and H. Wang, "Broadcast authentication for group communication," *Theor. Comput. Sci.*, vol. 269, nos. 1–2, pp. 1–21, Oct. 2001.

[14] B. Libert, K. G. Paterson, and E. A. Quaglia, "Anonymous broadcast encryption: Adaptive security and efficient constructions in the standard model," in *Proc. PKC*, in Lecture Notes in Computer Science, vol. 7293. Berlin, Germany: Springer, 2012, pp. 206–224.

[15] N. Fazio and I. M. Perera, "Outsider-anonymous broadcast encryption with sublinear ciphertexts," in *Public Key Cryptography—PKC 2012*, M. Fischlin, J. Buchmann, and M. Manulis, Eds. Berlin, Germany: Springer, 2012, pp. 225–242.

[16] A. Kiayias and K. Samari, "Lower bounds for private broadcast encryption," in *Proc. IH*, in Lecture Notes in Computer Science, vol. 7692. Berlin, Germany: Springer, 2013, pp. 176–190.

[17] J. Li and J. Gong, "Improved anonymous broadcast encryptions," in *Proc. ACNS*, in Lecture Notes in Computer Science, vol. 10892. Cham, Switzerland: Springer, 2018, pp. 497–515.

[18] A. Costin, A. Zarras, and A. Francillon, "Automated dynamic firmware analysis at scale: A case study on embedded web interfaces," in *Proc. ASIACCS*, 2016, pp. 437–448.

[19] A. Costin, J. Zaddach, A. Francillon, and D. Balzarotti, "A large-scale analysis of the security of embedded firmwares," in *Proc. USENIX Secur.*, 2014, pp. 95–110.

[20] E. Fernandes, J. Jung, and A. Prakash, "Security analysis of emerging smart home applications," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2016, pp. 636–654.

[21] E. Ronen, A. Shamir, A. Weingarten, and C. O'Flynn, "IoT Goes nuclear: Creating a ZigBee chain reaction," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2017, pp. 195–212.

[22] E. Fernandes, J. Paupore, A. Rahmati, D. Simionato, M. Conti, and A. Prakash, "FlowFence: Practical data protection for emerging IoT application frameworks," in *Proc. USENIX Secur.*, 2016, pp. 531–548.

[23] Z. B. Celik, G. Tan, and P. McDaniel, "IoTGuard: Dynamic enforcement of security and safety policy in commodity IoT," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2019, pp. 1–15.

[24] J. Fan, Y. He, B. Tang, Q. Li, and R. Sandhu, "Ruledger: Ensuring execution integrity in trigger-action IoT platforms," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, May 2021, pp. 1–10.

[25] M. P. Andersen, S. Kumar, M. AbdelBaky, G. Fierro, J. Kolb, H.-S. Kim, D. E. Culler, and R. A. Popa, "WAVE: A decentralized authorization framework with transitive delegation," in *Proc. USENIX Secur.*, 2019, pp. 1375–1392.

[26] A. L. M. Neto, A. L. F. Souza, I. Cunha, M. Nogueira, I. O. Nunes, L. Cotta, N. Gentille, A. A. F. Loureiro, D. F. Aranha, H. K. Patil, and L. B. Oliveira, "AoT: Authentication and access control for the entire IoT device life-cycle," in *Proc. SenSys*, 2016, pp. 1–15.

[27] S. Kumar, Y. Hu, M. P. Andersen, R. A. Popa, and D. E. Culler, "JEDI: Many-to-many end-to-end encryption and key delegation for IoT," in *Proc. USENIX Secur.*, 2019, pp. 1519–1536.

[28] T. D. Nguyen, S. Marchal, M. Miettinen, H. Fereidooni, N. Asokan, and A. Sadeghi, "DÏoT: A federated self-learning anomaly detection system for IoT," in *Proc. IEEE 39th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jul. 2019, pp. 756–767.

[29] Y. J. Jia, Q. A. Chen, S. Wang, A. Rahmati, E. Fernandes, Z. M. Mao, and A. Prakash, "Contexlot: Towards providing contextual integrity to appified IoT platforms," in *Proc. NDSS*, 2017, pp. 1–15. [Online]. Available: https://www.ndss-symposium.org/ndss2017/ndss-2017-programme/contexlot-towards-providing-contextual-integrity-appified-iot-platforms/

[30] S. Raza, L. Wallgren, and T. Voigt, "SVELTE: Real-time intrusion detection in the Internet of Things," *Ad Hoc Netw.*, vol. 11, no. 8, pp. 2661–2674, Nov. 2013.

[31] X. Lei, G.-H. Tu, C.-Y. Li, T. Xie, and M. Zhang, "SecWIR: Securing smart home IoT communications via Wi-Fi routers with embedded intelligence," in *Proc. MobiSys*, 2020, pp. 260–272.

[32] M. Xu, M. Huber, Z. Sun, P. England, M. Peinado, S. Lee, A. Marochko, D. Mattoon, R. Spiger, and S. Thom, "Dominance as a new trusted computing primitive for the Internet of Things," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2019, pp. 1415–1430.

[33] K. Suzaki, A. Tsukamoto, A. Green, and M. Mannan, "Reboot-oriented IoT: Life cycle management in trusted execution environment for disposable IoT devices," in *Proc. Annu. Comput. Secur. Appl. Conf.*, Dec. 2020, pp. 428–441.

[34] H. Liang, J. Wu, S. Mumtaz, J. Li, X. Lin, and M. Wen, "MBID: Micro-blockchain-based geographical dynamic intrusion detection for V2X," *IEEE Commun. Mag.*, vol. 57, no. 10, pp. 77–83, Oct. 2019.

[35] H. Yao, T. Mai, J. Wang, Z. Ji, C. Jiang, and Y. Qian, "Resource trading in blockchain-based industrial Internet of Things," *IEEE Trans. Ind. Informat.*, vol. 15, no. 6, pp. 3602–3609, Jun. 2019.

[36] Y. Lu, X. Huang, K. Zhang, S. Maharjan, and Y. Zhang, "Blockchain empowered asynchronous federated learning for secure data sharing in Internet of Vehicles," *IEEE Trans. Veh. Technol.*, vol. 69, no. 4, pp. 4298–4311, Apr. 2020.

[37] H. Liang, J. Wu, X. Zheng, M. Zhang, J. Li, and A. Jolfaei, "Fog-based secure service discovery for Internet of Multimedia Things: A cross-blockchain approach," *ACM Trans. Multimedia Comput., Commun., Appl.*, vol. 16, no. 3s, pp. 1–23, Oct. 2020.

[38] Y. Watanabe, N. Yanai, and J. Shikata, "Anonymous broadcast authentication for securely remote-controlling IoT devices," in *Advanced Information Networking and Applications*. Cham, Switzerland: Springer, 2021, pp. 679–690.

[39] Y. Zhauniarovich, I. Khalil, T. Yu, and M. Dacier, "A survey on malicious domains detection through DNS data analysis," *ACM Comput. Surv.*, vol. 51, no. 4, pp. 1–36, Jul. 2019.

[40] M. Bellare and C. Namprempre, "Authenticated encryption: Relations among notions and analysis of the generic composition paradigm," in *Advances in Cryptology—ASIACRYPT 2000*, T. Okamoto, Ed. Berlin, Germany: Springer, 2000, pp. 531–545.

[41] M. Bellare, O. Goldreich, and A. Mityagin, "The power of verification queries in message authentication and authenticated encryption," Cryptol. ePrint Arch., IACR, Tech. Rep. 2004/309, 2004, p. 20.

[42] M. Bellare, "New proofs for NMAC and HMAC: Security without collision-resistance," in *Proc. CRYPTO*, in Lecture Notes in Computer Science, vol. 4117. Berlin, Germany: Springer, 2006, pp. 602–619.

[43] Y. Dodis, E. Kiltz, K. Pietrzak, and D. Wichs, "Message authentication, revisited," in *Advances in Cryptology—EUROCRYPT 2012*, D. Pointcheval and T. Johansson, Eds. Berlin, Germany: Springer, 2012, pp. 355–374.

[44] O. Goldreich, S. Goldwasser, and S. Micali, "How to construct random functions," *J. ACM*, vol. 33, no. 4, pp. 792–807, Aug. 1986.

[45] J. Håstad, R. Impagliazzo, L. A. Levin, and M. Luby, "A pseudorandom generator from any one-way function," *SIAM J. Comput.*, vol. 28, no. 4, pp. 1364–1396, Jan. 1999.

[46] R. L. Rivest, "Chaffing and winnowing: Confidentiality without encryption," *CryptoBytes, RSA Laboratories*, vol. 4, no. 1, pp. 12–17, 1998.

[47] G. Hanaoka, Y. Hanaoka, M. Hagiwara, H. Watanabe, and H. Imai, "Unconditionally secure chaffing-and-winnowing: A relationship between encryption and authentication," in *Proc. AAECC*, in Lecture Notes in Computer Science, vol. 3857. Berlin, Germany: Springer, 2006, pp. 154–162.

[48] W. Kitada, G. Hanaoka, K. Matsuura, and H. Imai, "Unconditionally secure chaffing-and-winnowing for multiple use," in *Proc. ICITS*, in Lecture Notes in Computer Science, vol. 4883. Berlin, Germany: Springer, 2009, pp. 133–145.

[49] R. A. Fisher and F. Yates, *Statistical Tables for Biological, Agricultural, and Medical Research*. New York, NY, USA: Hafner, 1953.

[50] M. Bellare, T. Ristenpart, P. Rogaway, and T. Stegers, "Format-preserving encryption," Cryptol. ePrint Arch., IACR, Tech. Rep. 2009/251, 2009. [Online]. Available: https://ia.cr/2009/251

[51] N. Mouha, B. Mennink, A. Van Herrewege, D. Watanabe, B. Preneel, and I. Verbauwhede, "Chaskey: An efficient MAC algorithm for 32-bit microcontrollers," in *Proc. SAC*, in Lecture Notes in Computer Science, vol. 8781. Cham, Switzerland: Springer, 2014, pp. 306–323.

[52] G. Hartung, M. Hoffmann, M. Nagel, and A. Rupp, *BBA+: Improving the Security and Applicability of Privacy-Preserving Point Collection*. New York, NY, USA: ACM, 2017, pp. 1925–1942.

KAZUHIKO MINEMATSU received the B.E., M.E., and Dr.S. degrees from Waseda University, in 1996, 1998, and 2008, respectively. Since 1998, he has been with NEC, where he is currently a Research Fellow in the field of symmetric-key cryptography and its application systems. Since 2020, he has been a Visiting Professor with the Institute of Advanced Sciences, Yokohama National University, Yokohama, Japan. He served as a member for several committees, such as the Cryptography Research and Evaluation Committee (CRYPTREC), a Japanese organization for cryptography standardization and evaluation. He received multiple awards, including the Best Paper Award from FSE 2015 and CRYPTO 2019 from the International Association for Cryptologic Research (IACR).

JUNJI SHIKATA (Member, IEEE) received the B.S. and M.S. degrees in mathematics from Kyoto University, Kyoto, Japan, in 1994 and 1997, respectively, and the Ph.D. degree in mathematics from Osaka University, Osaka, Japan, in 2000. From 2000 to 2002, he was a Postdoctoral Fellow with the Institute of Industrial Science, The University of Tokyo, Tokyo, Japan. Since 2002, he has been with the Graduate School of Environment and Information Sciences, Yokohama National University, Yokohama, Japan. From 2008 to 2009, he was a Visiting Researcher with the Department of Computer Science, Swiss Federal Institute of Technology (ETH), Zürich, Switzerland. He is currently a Professor with Yokohama National University. His research interests include cryptology, information theory, theoretical computer science, and computational number theory. He received several awards, including the Wilkes Award from the British Computer Society, in 2006, and the Young Scientists' Prize, the Commendation for Science and Technology from the Minister of Education, Culture, Sports, Science and Technology, Japan, in 2010.

YOHEI WATANABE (Member, IEEE) received the B.E., M.E., and Ph.D. degrees in information science from Yokohama National University, in 2011, 2013, and 2016, respectively. He is currently an Assistant Professor with The University of Electro-Communications. He is also an Invited Adviser with NICT, a Collaborative Researcher with AIST, and a Research Fellow with Japan Datacom Company Ltd. His research interests include cryptography and information security. He is a member of IEICE, IPSJ, and IACR.

NAOTO YANAI (Member, IEEE) received the B.E. degree from the National Institution of Academic Degrees and University Evaluation, Japan, in 2009, and the M.S.Eng. and Ph.D. degrees in engineering from the Graduate School of Systems and Information Engineering, University of Tsukuba, Japan, in 2011 and 2014, respectively. From 2014 to 2021, he was an Assistant Professor with Osaka University, Osaka, Japan, where he is currently an Associate Professor. His research interest includes information security.