ITUPublications

International Telecommunication Union
Standardization Sector

ITU-T Technical Report

(09/2023)

TR.ba-iot

Broadcast authentication scheme for Internet of things (IoT) system

Technical Report ITU-T TR.ba-iot

Broadcast authentication scheme for Internet of things (IoT) system

Summary

In order to specify the broadcast authentication (BA) schemes for an Internet of things (IoT) system, this Technical Report provides a conceptual model of the target BA system and clarifies the security characteristics and requirements required therein. In addition, this Technical Report specifies a message authentication code (MAC)-based BA authentication scheme and a digital signature (DS)-based BA authentication scheme as methods to achieve the security requirements and shows the convenience and usage of each method.

Keywords

Authentication scheme, broadcast authentication, IoT systems, remote control system, security.

Note

This is an informative ITU-T publication. Mandatory provisions, such as those found in ITU-T Recommendations, are outside the scope of this publication. This publication should only be referenced bibliographically in ITU-T Recommendations.

Table of Contents

Technical Report ITU-T TR.ba-iot

Broadcast authentication scheme for Internet of things (IoT) system

1 Scope

This Technical Report presents a scheme of broadcast authentication (BA) to realize a secure teleoperation system by specifying a subset of arbitrary Internet of things (IoT) devices in the target BA system and broadcasting the commands for securely controlling the IoT devices.

2 References

None.

3 Definitions

3.1 Terms defined elsewhere

This Technical Report uses the following term defined elsewhere:

3.1.1 message authentication code (MAC) [b-ITU-T X.813]: A cryptographic check value that is used to provide data origin authentication and data integrity.

3.2 Terms defined in this Technical Report

This Technical Report defines the following terms:

3.2.1 message authentication: A property that guarantees that a message has not been modified while in transit to ensure data integrity, and allows the receiving party to verify the source of the message.

3.2.2 BA system: Internet of things (IoT) system with broadcast authentication (BA) scheme implemented.

4 Abbreviations and acronyms

This Technical Report uses the following abbreviations and acronyms:

AMQS	Approximate Membership Query (Data) Structure
BA	Broadcast Authentication
DDoS	Distributed Denial of Service
DS	Digital Signature
IoT	Internet of Things
MAC	Message Authentication Code
mMTC	massive Machine-Type Communication

5 Conventions

None.

6 Conceptual model

6.1 General

A broadcast authentication system (or BA system for short) is a basic system in which a central entity (e.g., a sender) has the ability to control a number of arbitrary target entities (i.e., IoT devices) remotely and simultaneously.

Note that due to the often numerous IoT devices concerned, one-to-many communications is considered, rather than individual communication between the sender and each IoT device (see Figure 1). In this system, a sender broadcasts a command to all IoT devices so that only designated devices can execute the command. Each IoT device checks the validity of the command, and executes the command if it is regarded as valid. This simplified model of the remote control system captures characteristics of the IoT era and massive machine-type communication (mMTC) in the context of (Beyond) 5G.

In this document, a message authentication code (MAC)-based anonymous broadcast authentication and digital signature (DS)-based broadcast authentication are considered. MAC-based anonymous broadcast authentication (BA) is a BA system with anonymity, and it conceals information on target entities from any device except for the information for which the device itself is the target. DS-based BA is a BA system using a digital signature and is characterized by the fact that the required data size does not depend on the number of IoT devices to be controlled. This allows a trade-off between the data size compression ratio and the false positive probability in control (the probability that a non-target device can accept a command), which allows flexible parameter settings.

To date, no security solution exists that efficiently and securely controls a subset of the large number of IoT devices. There are several protocols, commonly referred to as broadcast authentication, but they do not have the ability to specify a subset of recipients for authentication that the BA system described above has. A cryptographic protocol closely related to this is broadcast encryption, in which the sender encrypts plaintext and specifies an arbitrary subset of recipients so that only the specified recipients can decrypt the encrypted plaintext. However, the BA scheme is an authentication scheme that performs one-to-many authentication and is suitable for IoT control, which is a quite different functionality from broadcast encryption, see [b-ISO/IEC 29192-7].

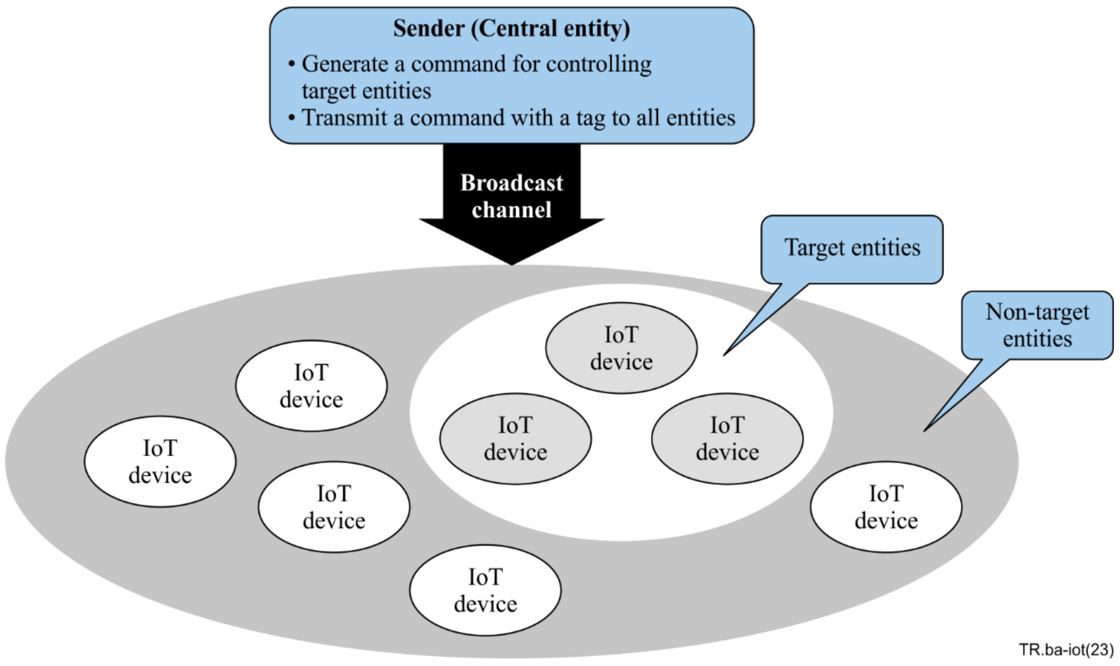
Sender (Central entity)
• Generate a command for controlling target entities
• Transmit a command with a tag to all entities
Broadcast channel
Target entities
Non-target entities
IoT device
TR.ba-iot(23)

Figure 1 – Broadcast authentication system with one-to-many communications

It should be noted that this scheme basically requires that the IoT devices to be controlled be identifiable at all times by the controlling entity in order to control them remotely. If, for example, the network identifier changes due to dynamic roaming by the IoT device, it becomes difficult to identify it as a network level, but this scheme can be applicable in such cases by always maintaining the correspondence between the upper layer identifier of the IoT device and the network identifier.

Some use cases and implementations of this scheme are described in Appendix I of this document. For example, if a target IoT device is infected with an IoT malware and the device is expected to become completely inoperable, by attaching an adapter for broadcast authentication (BA) to the IoT device and using a separate network, this BA scheme can be used in such an unusual situation. However, in general, when an IoT device is infected with IoT malware, many of the functions of the IoT device are retained normally and the device is then often used as a proxy for malicious activity. In such cases pre-installing BA functionality on the IoT device makes it possible to use the BA without an adapter scheme.

6.2 A conceptual model of the broadcast authentication system

A conceptual model of the BA system is basically composed of four processes namely set-up, key generation, tagging, and verification as follows:

1) Set-up takes a security parameter as input and outputs an authentication key, which is a secret key for a sender, (and an optional public parameter).

2) Key generation takes the authentication key and an identity ID of an IoT device (and the optional public parameter) as input and outputs a verification key for the ID. The verification key may be kept secret (depending on whether the construction is based on MAC or DS).

3) Tagging takes the authentication key, the identities of multiple IoT devices, and a command as input and produces a tag (or a signature) as output.

4) Verification takes a verification key of an identity ID, a command, and its corresponding tag and the optional public parameter as input, and outputs 1 if the pair of the command and tag is valid and the ID was designated when the tag was generated. Otherwise, verification outputs 0.

7 Security requirements

The scenario where a sender broadcasts an authenticated command (i.e., a command with a tag) to all devices via a broadcast channel should be considered. In this situation, there is an adversary who can eavesdrop, insert, delay, and modify all the transmitted information but does not remove it from the channel or send jam signals. In addition, the adversary may carry out the attacks after stealing the verification keys of some devices by corrupting them. It is assumed that the main purpose of the adversary is to maliciously modify an authenticated command so that some of designated devices do not execute the command and/or some of the non-designated devices execute the command. Furthermore, all devices receive the same information; if an authenticated command is modified, all devices receive the modified command.

In this case, the BA system is required to meet the security requirements of completeness, integrity, and anonymity. Completeness and integrity are fundamental requirements and anonymity is optional.

7.1 Completeness

Only designated devices execute a command unless the corresponding authenticated command is (maliciously or accidentally) modified. In other words, any non-designated device does not execute the command as long as it receives the authenticated command as it is.

7.2 Integrity

If an authenticated command is (maliciously or accidentally) modified, any device can detect it and reject it. Even if an adversary corrupts some of devices and steals verification keys of the devices, all the non-corrupted devices can detect a maliciously modified authenticated command and reject it.

7.3 Anonymity

Anonymity is optional in BA systems. An anonymous BA system provides the property that an authenticated command does not leak any information on the designated devices except for the number of them. This property is satisfied in the anonymous BA, even if some of the devices are corrupted by an adversary.

8 BA schemes

This document provides two basic schemes of the BA systems: an anonymous BA scheme based on message authentication codes (MAC) and a non-anonymous BA protocol based on digital signatures (DS). The MAC-based anonymous BA scheme is constructed from MAC and random permutations. The DS-based non-anonymous BA construction consists of DS and approximate membership query structure (AMQS).

8.1 MAC-based anonymous BA constructions

In this clause, a construction of an anonymous BA scheme is provided from a message authentication code (MAC) and a random permutation: a MAC is used to meet the integrity requirement, while a random permutation is utilized to satisfy that of anonymity. If a MAC-based BA scheme without anonymity is required, it is possible to eliminate the use of the random permutation in the construction. The feature of the construction lies in the efficient running time, but the data-size required in it is proportional to the number of target controlled devices. It should be noted that completeness is satisfied in the construction.

8.1.1 Specific notation

id	ID of a device.

m	Command (message).
r	(Pseudo)Random number.
Σ	(Pseudo)Random permutation over a set. If Σ is a random permutation over the set $\{1,2,\ldots,s\}$, where s is a positive integer larger than one, the output sequence $(\Sigma(1),\Sigma(2),\ldots,\Sigma(s))$ is random such that every integer in $\{1,2,\ldots,s\}$ appears only once.
ak	Authentication key of sender.
K_{id}	Verification key of a device id. For simplicity, K_i denotes the secret key corresponding to id_i instead of K_{id_i}.
$F()$	MAC function which takes a secret key and a message as input and outputs a MAC tag.
$\|$	concatenation

8.1.2 Set-up

This algorithm takes a security parameter as input. For each id, this algorithm generates a random key, which is a verification key of a device id. Let $ak = \{K_{id}\}$ be collection of verification keys of devices. It outputs ak as an authenticated key.

8.1.3 Key generation

This algorithm takes an authenticated key $ak = \{K_{id}\}$, and id as an ID as input, and outputs the corresponding verification key K_{id} from the collection of verification keys $\{K_{id}\}$.

8.1.4 Tagging

This algorithm takes an authentication key $ak = \{K_{id}\}$, a set of identities of multiple IoT devices $S = \{id_1, id_2, \ldots, id_s\}$, and a command m as input. It outputs a tag σ after the following procedure:

Step 1: Take a (pseudo-)random number r. For each $id \: in \: S$, compute a MAC tag t_{id} which is computed by the MAC function $F(K_{id}, m \parallel r)$.

Step 2: Arrange the positive integers $1,2,\ldots,s$ in a random order. For doing it, take a (pseudo-)random permutation Σ over the set $\{1,2,\ldots,s\}$, and compute a random sequence $(\Sigma(1),\Sigma(2),\ldots,\Sigma(s))$. Then, based on the random sequence $(\Sigma(1),\Sigma(2),\ldots,\Sigma(s))$, arrange the MAC tags as well, and the resulting sequence of MAC tags is $(t_{id_{\Sigma(1)}}, t_{id_{\Sigma(2)}}, \ldots, t_{id_{\Sigma(s)}})$. This algorithm finally outputs the random number and the random sequence of MAC tags as a tag σ, namely a tag $\sigma = (r, t_{id_{\Sigma(1)}}, t_{id_{\Sigma(2)}}, \ldots, t_{id_{\Sigma(s)}})$.

8.1.5 Verification

Verification takes a verification key of a device id denoted by K_{id}, a command m, and a tag σ as input. This algorithm outputs 1 if and only if there is a valid MAC tag in a tag σ. Specifically, for the tag $\sigma = (r, t_1, t_2, \ldots, t_s)$, do the following procedure.

```
Step 1: $j \leftarrow 1$.
Step 2: If $t_j = F(K_{id}, m \parallel r)$, output 1 and terminate the process.
Step 3: If $j = s$, output 0 and terminate the process.
Otherwise, set $j \leftarrow j + 1$, and go to Step 2.
```

8.2 DS-based non-anonymous BA construction

In this clause, a construction of a DS-based non-anonymous BA construction is provided from a digital signature (DS) and approximate membership query structure (AMQS): a DS is used to meet the integrity requirement, while AMQS is utilized to realize the compact size of a tag. The feature of the construction lies in that the data-size required in it does not depend on the number of target controlled devices, and flexible parameter setting is possible by allowing a trade-off between data-size compression rate and the false positive probability in controlling (i.e., the probability that a non-target device might accept the command).

8.2.1 Specific notation

id	ID of a device.
M	Command (message).
ak	Authentication key of sender.
Pp	Public parameter
K_{id}	Verification key of a device *id*. For simplicity, K_i denotes the secret key corresponding to id_i instead of K_{id_i}.
DS=(*SigGen*, *SigSign*, *SigVer*)	Digital signature *DS* consisting of a key generation algorithm *SigGen*, a signing algorithm *SigSign*, and a verification algorithm *SigVer*.
(*sigk*, *verk*)	A pair of the signing key *sigk* and the verification key *verk* generated by a key generation algorithm *SigGen* of *DS*.
AMQS=(*Gen*, *Insert*, *Lookup*)	Approximate membership query (data) structure *AMQS* over a whole set *U* that consists of a generation algorithm *Gen*, an inserting algorithm *Insert*, and a lookup algorithm *Lookup*. (See Note 1); *Gen* takes the set *U* and a parameter *par* as input, and it outputs an initial structure and auxiliary information, where *par* depends on an instantiation of AMQS; *Insert* takes a data structure, an element of U, and auxiliary information as input, and it outputs an updated data structure; *Lookup* takes a data structure, an element of U, and auxiliary information as input, and it outputs *true* or *false*, where *true* means the element of *U* is contained in the data structure while *false* means it is not.
Assign ( )	Function in AMQS over the whole set U, which takes a positive integer ℓ and a device *id* and outputs a set of ℓ elements in U such that $\mathrm{Assign}(\ell, id_1) \cap \mathrm{Assign}(\ell, id_2) = \emptyset$ for any distinct id_1 and id_2. This function uniquely assigns multiple elements in *U* into an arbitrary ID. (See Note 2)
$\|$	concatenation

NOTE 1 – There are many instantiations of AMQS such as the Bloom filter, cuckoo filter, and vacuum filter.

NOTE 2 – For example, the function Assign() is constructed as follows: Suppose that ID is represented by a binary string with γ bits and let $U = \{0,1\}^{\gamma+\lfloor\log_2 \ell\rfloor+1}$; Then, define $\mathrm{Assign}(\ell, id) = \{(id \| \beta_1), (id \| \beta_2), \ldots, (id \| \beta_\ell)\}$, where β_i is binary representation of i for $1 \leq i \leq \ell$.

8.2.2 Set-up

This algorithm takes a security parameter and generates a signing key *sigk* and a verification key *verk* by using a key generation algorithm *SigGen* of *DS*. The algorithm also chooses a positive integer ℓ, which will determine a parameter of AMQS and outputs an authentication key $ak = (sigk, \ell)$ and a public parameter $pp = (verk, \ell)$.

8.2.3 Key generation

This algorithm takes the positive integer ℓ which is a part of a public parameter, and an ID id as input. By the integer ℓ, parameters of AMQS including the description of a whole set U and the function Assign() are determined (see Note 2 in clause 8.2.1). The algorithm outputs a verification key K_{id} which is consisting of ℓ elements in U and computed by using the function Assign() in AMQS, namely $K_{id} = \text{Assign}(\ell, id)$.

8.2.4 Tagging

This algorithm takes an authentication key $ak = (sigk, \ell)$, a set of identities of multiple IoT devices $S = \{id_1, id_2, \ldots, id_s\}$, and a command m as input. Basically, this algorithm transforms all the ID data of S into $s\ell$ elements in U (i.e., the set $\widehat{K}$ below), and generates a signature that shows validity of all the data. Specifically, this algorithm outputs a tag σ after the following procedure:

Step 1: Derive a parameter par of AMQS from the set of identities S. Then, get an initial structure T_0 of AMQS and auxiliary information aux by computing $Gen(U, par)$ in AMQS.

Step 2: For each i with $1 \leq i \leq s$, compute a verification key $K_i = \text{Assign}(\ell, id_i)$ which consists of ℓ elements in U. Suppose the set K_i is denoted by $K_i = \{k_{(i-1)\ell+1}, k_{(i-1)\ell+2}, \ldots, k_{i\ell}\}$ for every i with $1 \leq i \leq s$. Then, from the property of the function Assign() , the set $\widehat{K} = \bigcup_{i=1}^{s} K_i = \{k_1, k_2, k_3, \ldots, k_{s\ell-1,} k_{s\ell}\}$ consists of different $s\ell$ elements in U.

Step 3: Update a data structure starting from the initial one T_0 until all elements k_j ($1 \leq j \leq s\ell$) in U are inserted into the data structure. Namely, for $j = 1,2, \ldots, s\ell$, compute $T_j \leftarrow Insert(T_{j-1}, k_j, aux)$. The final data structure $T_{s\ell}$ guarantees that all elements of $\widehat{K}$ transformed from all the ID data of S are included.

Step 4: Generate a signature σ_{DS} by computing $SigSign(sigk, m \parallel T_{s\ell} \parallel aux)$ in order to show the validity of the data ($m \parallel T_{s\ell} \parallel aux$). Then, a tag is defined as $\sigma = (T_{s\ell}, aux, \sigma_{DS})$.

8.2.5 Verification

Verification algorithm takes a public parameter $pp = (verk, \ell)$, a verification key K_{id} of a device whose ID is id, a command m, and a tag σ as input. Each device can check if its own ID is included in the data structure as follows: the verification key consists of ℓ elements in U that are transformed from its own ID; Verification algorithm regards that the target ID is included in the data structure if and only if all the corresponding ℓ elements in U are included in the data structure. Specifically, for the tag $\sigma = (T_{s\ell}, aux, \sigma_{DS})$, Verification algorithm outputs 0 if σ_{DS} is invalid, namely $0 \leftarrow \text{SigVer}(\text{verk}, (\text{m} \parallel T_{s\ell} \parallel aux))$. If $\text{Lookup}(T_{s\ell}, k, \text{aux})$ outputs *true* for every $k \; in \; K_{id}$, the verification algorithm outputs 1, and otherwise outputs 0.

Appendix I

Assumed applications (use cases) for BA schemes

Introduction

When utilizing the BA scheme, use cases that possess the following features and characteristics should be selected and considered for its utilization:

- the IoT system has a very large number of IoT devices;
- the IoT devices in use require a high level of security;
- access to the IoT devices in use is difficult;
- the administrator of the IoT system utilizing the IoT devices is ambiguous.

This appendix provides the following three use cases that possess the above features and characteristics and explains them for utilizing the BA scheme with "remote emergency stop", "remote update", "life cycle control" and so on.

Use case 1: IoT devices for smart factories

Smart factories can achieve high productivity and continuous improvement by monitoring and analysing work contents with a wide variety of IoT devices. In smart factories, a huge number of IoT devices are used in a certain area, and there are various types of IoT devices, such as cameras, motion sensors, or actuators. With the expansion of smart factories, the number of IoT devices is expected to increase sharply. IoT devices used in smart factories are also subject to threats such as hacking, and malware infection just like general communication devices. In smart factories, threats are directly linked to serious incidents, such as reduced productivity, fatal accidents, or leaking of confidential data as a result of unauthorized operations by insiders or malware infection. The organizations operating smart factories are required to properly manage IoT devices against such threats.

However, it is difficult to fully establish a security management system in smart factories due to the huge variety of IoT devices, multiple types of threats, lack of administrators, and the fact that security measures are not directly linked to profits. Under these circumstances, smart factories are often the target of cyber-attacks, and in order to minimize the risk of threats, a mechanism that can safely, easily, and instantly control a huge number of diverse IoT devices is required.

A BA system as presented in this document that can remotely and safely control only a specific set of devices from a huge number of IoT devices is one of the ideal mechanisms for managing IoT devices in a smart factory. In particular, from the viewpoint of preventing the spread of damage caused by malware infection or internal threats, it is effective to use a MAC-based authentication scheme which is anonymous such as BA schemes constructed from MAC and random permutation.

For example, in a smart factory in the manufacturing industry, IoT devices are often managed by the IT department, so the BA scheme will also be operated by the IT department. The IT department implements a BA system on IoT devices installed at the manufacturing site and sets an ID and authentication key for each. The receivers of the BA system can be retrofitted to all IoT devices as additional hardware, so it can be installed in all types of IoT devices. When a movement that leads to malicious activity, such as unauthorized operation due to malware infection or taking out of an IoT device, is detected, the IT department generates a message and tag as a sender and broadcasts it to all IoT devices. In this case, the content of the message can be an emergency stop or non-functionalization. Then, the receivers of the BA system implemented in the IoT devices receive the message, verify the validity and legitimacy of the message, and safely stop it only when the validity and legitimacy of the message is proved. Control through the BA system is not only legitimate but also highly anonymous, allowing it to respond undetected by attackers and malicious insiders. Also, in smart factories, gateways often control multiple IoT devices, so the administrator can stop damage spread by applying the BA scheme to the gateway. In this way, by managing IoT devices safely and easily using the BA system in the smart factory, it is possible to minimize the impacts from malware infection, insider threat and so on.

The target smart factories are factories that require a high degree of security, such as factories that handle security-related or high-tech products. By introducing the BA system, these factories can improve security, satisfy motivations such as stable factory operation and reduction of unauthorized communication, and obtain safe operation and customer satisfaction.

Use case 2: IoT devices for infrastructure monitoring

In recent years, there have been many cases where IoT devices are widely used for monitoring infrastructure, such as monitoring deterioration of bridges or water pipes, investigating water levels in rivers or dams, and prediction of volcanic activities. By installing a large number of IoT devices for infrastructures, it is possible not only to acquire more detailed data, but also to install it in places that are difficult for people to access and so 0reducing management costs. For these reasons, the number of IoT devices for infrastructure monitoring is expected to increase.

While, many IoT devices have an expiration date due to the device life cycle and need to be replaced or disposed at an appropriate time, due to the large number of devices and the difficulty of accessing their locations, IoT devices for infrastructure monitoring can be left abandoned even after their lifetime. In addition, since many infrastructure monitoring IoT devices are powered by solar power, they will not turn off even after their lifetime, and abandoned IoT devices can continue to emit unnecessary communications. To make matters worse, when IoT devices are infected with malware, the abandoned IoT devices can become a hotbed for distributed denial of service (DDOS) attacks due to springboard attacks.

In order to solve these problems, installing the BA system for infrastructure monitoring IoT devices is one of the best solutions. In particular, a non-anonymous BA protocol based on DS is preferable for remotely managing a large number of IoT devices for infrastructure that exist over a wide area.

For example, it is assumed that IoT devices that monitor the aging of water pipes buried underground are installed over a wide area throughout the town. Furthermore, in order to physically access these IoT devices, it is necessary to dig them up from the ground, so if this device reaches the expiration date, it will cost a lot to collect and may be left unattended. To avoid this difficulty, the administrator installs the BA system to infrastructure monitoring IoT devices in advance, and sets the ID and lifetime of each IoT device. If there are IoT devices that have expired, the BA can safely and remotely shut down the IoT devices without digging them up by broadcasting messages and tags to all IoT devices. In addition, the administrator can order the backup device to boot through the BA system, allowing the expired IoT devices to be replaced in sequence without physical access. In another example, suppose that some of the large number of monitoring IoT devices installed in a mountain river have been infected with malware. The administrator orders the BA system to stop the monitoring IoT devices that have infected, and the sender broadcasts to all monitoring IoT devices located in the mountain river. All IoT devices receive the message, and the infected device to be stopped is safely stopped remotely by turning off the power after confirming the validity of the message.

In this way, it does not require a great deal of effort to replace several devices from a large number of devices, and all instructions are executed after ensuring their legitimacy, so safety is also ensured.

Use case 3: IoT devices for smart home

In recent years, home appliances with telecommunication functions have become widespread. A smart home that creates a new lifestyle through home appliances connected to the Internet will be realized using a large number of IoT devices. While the number of IoT devices for smart homes is increasing rapidly, there are major concerns about their security threats. Threats in smart homes can affect the physical space such as leakage of personal information, unlocking of smart lock systems, and unauthorized operation of home appliances.

Despite this situation, some IoT devices can be left connected to the Internet after the expiration date, or security patches may not be updated. In addition, there is a threat that can be a springboard by hijacking smart home IoT devices. Those threats are partly due to the fact that IoT devices for smart homes are used in ordinary homes, and there is no dedicated administrator with knowledge of cyber security. Another reason is that IoT devices used in smart homes are diverse and it takes a lot of effort to take security measures for each. Furthermore, since various stakeholders such as residents, device makers, and smart home service providers are involved in smart homes, the ambiguity of their responsibilities is another cause of the problem. Therefore, it is necessary to have a system that manages the security of IoT devices for smart homes in a unified manner which considers updates and safety.

It is preferable to adopt the BA system for smart home IoT devices that have such problems. The operation of BA systems primarily involves IoT device manufacturers, smart home service providers, and residents. Device manufacturers need to ensure security from the factory and for initial set-up of IoT devices, so they implement the BA system and assign IDs and authentication keys for each IoT device. A smart home service provider that installs and manages IoT devices in a house can keep track of the ID of each IoT device, monitor lifetime and abnormal communication, and send commands to the target device through the BA system when necessary. Residents need to understand the management of the BA system and agree on emergency stop and automatic remote updates.

For example, if a new security patch is distributed by the manufacturer to the IoT device used in the smart home, the smart home service provider will obtain the patch and send an update program after specifying the device that needs to be updated through the BA system. In another example, if a device that is not under the control of a service provider that is individually installed by a resident reaches the expiration date, the manufacturer can safely shut down all manufactured IoT devices through the BA system directly. The MAC-based authentication scheme is preferable for emergency stop when an IoT device that handles highly confidential data such as cameras and biological monitoring is infected with malware. On the other hand, the DS-based authentication scheme is considered preferable for less urgent items such as general IoT home appliance updates.

Bibliography

[b-ITU-T X.813] Recommendation ITU-T X.813 (1996), *Information technology – Open Systems Interconnection – Security frameworks for open systems: Non-repudiation framework*.

[b-ISO/IEC 29192-7] International Standard ISO/IEC 29192:2019, *Information security – Lightweight cryptography – Part 7: Broadcast authentication protocols*.