An ATM Security Measure to Prevent Unauthorized Deposit with a Smart Card

Hisao OGATA†,††, Member, Tomoyoshi ISHIKAWA†, Norichika MIYAMOTO†, Nonmembers, and Tsutomu MATSUMOTO††, Member

SUMMARY Recently, criminals frequently utilize logical attacks to Automated Teller Machines (ATMs) and financial institutes’ (FIs’) networks to steal cash. We proposed a security measure utilizing peripheral devices in an ATM for smart card transactions to prevent “unauthorized cash withdrawals” of logical attacks, and the fundamental framework as a generalized model of the measure in other paper. As the measure can prevent those logical attacks with tamper-proof hardware, it is quite difficult for criminals to compromise the measure. However, criminals can still carry out different types of logical attacks to ATMs, such as “unauthorized deposit”, to steal cash. In this paper, we propose a security measure utilizing peripheral devices to prevent unauthorized deposits with a smart card. The measure needs to protect multiple transaction sub-processes in a deposit transaction from multiple types of logical attacks and to be harmonized with existing ATM system/operations. A suitable implementation of the fundamental framework is required for the measure and such implementation design is confusing due to many items to be considered. Thus, the measure also provides an implementation model analysis of the fundamental framework to derive suitable implementation for each defense point in a deposit transaction. Two types of measure implementation are derived as the result of the analysis.

key words: ATM, security, malware, network, cryptography, device

1. Introduction

Recently, criminals frequently carry out logical attacks to Automated Teller Machines (ATMs) and Financial Institute’s (FI’s) networks to steal cash. Two kinds of logical attacks to steal cash are considered; one is “unauthorized cash withdrawal" that is cash withdrawal without debiting an FI’s account [1]–[8], and the other is “unauthorized deposit” that is a fraudulent increase of account balance without the equivalent amount of cash. Unauthorized cash withdrawals have occurred in more than 30 countries [9], and resulted in serious social impacts. Criminals attack any layers of ATM systems with network-based [1], malware-based [1]–[5], and hardware-based [1], [6]–[8] logical attacks. Existing security measures [1], [8], [10]–[13], such as the EUROPOL’s guidance [1] with more than 30 requirements, primarily try to protect the whole ATM, especially the PC of an ATM containing a number of files and huge data. However, such security measures trying to protect a number of files and huge data, which can be intrusion routes of malware and increase vulnerable risks, are not so effective or efficient due to the following reasons.

In general, an ATM consists of a PC running the Windows® Operating System (OS) and peripheral devices such as a card reader. Although a vulnerable ATM platform [14] commonly installed in the PC to control peripheral devices is frequently abused for unauthorized cash withdrawal, it is difficult to drastically modify the platform specifications because of the interoperability of existing ATM applications using the platform. Not only external crimes but also internal crimes should be supposed to take measures [5]. That is, frequent physical/logical accesses inside an ATM are required in existing operations, which increases risks of logical attacks to the ATM. For instance, typically cash replenishment/collection is once a few days or once a week, and updating software/contents in the PC is once a quarter. Regarding files in the PC, in some cases, the number of the executable files to protect in a Windows® 7-based ATM is more than twenty thousand, and the total number of the files to protect is more than millions or tens of millions if hundreds or thousands of ATMs are managed. Even whitelisted anti-malware software does not always work well to protect the files since it might be disabled by directly manipulating the PC. In this way, since the PC is not protected by tamper-proof hardware, tightly protecting such a huge number of files would bring quite heavy management workloads to FIs, and may result in breaking rules, human errors, and rather poor management by ATM operation staffs. Thus, we proposed an ATM security measure utilizing peripheral devices to prevent unauthorized cash withdrawal in the other paper [15], in which a peripheral device itself verifies the authenticity of a command accessing property sent from the PC. We also proposed the fundamental framework as a generalized model of the measure. Securing firmware in peripheral devices is a key point of the measure. The firmware can be protected with tamper-proof hardware since the number of the firmware, typically one or two, is much smaller than that of the files in the PC and existing secure elements can be applied for it. As just described, measures to prevent unauthorized cash withdrawals have been delivering well.

On the other hand, measures for unauthorized deposit have not been argued sufficiently, in which sub-processes in a deposit transaction is manipulated using malware and malicious hardware to fraudulently increase account balance...
with no actual cash or less cash. And then, criminals can withdraw cash with ATMs from the accounts. Unauthorized deposit is highly likely to occur in the near future because of the following strong incentives to criminals.

“Easiness”: (a) few existing measures explicitly focus on unauthorized deposit, (b) vulnerable ATM platform can be abused, and (c) new logical attacks can be easily created by modeling for widespread existing frauds using a physical trap such as “Transaction Reversal Fraud” [16] and “Cash Trap” [17], [18], which are explained in Sect. 2.2.

“Expandability”: criminals can also conduct fraudulent international remittance if the deposit accounts are linked with internet banking accounts. Since remittance services on ATMs are usually limited to domestic accounts, international remittance using internet banking is more preferable for criminals due to difficulty of a trace. The transferred money can be spent even to buy cryptocurrencies. Criminals can fraudulently open bank accounts linked with internet banking accounts or buy existing accounts to make accounts with a large balance utilizing ATMs. In this way, they can unlimitedly create internet banking accounts with a large balance utilizing ATMs without strenuous efforts to search such accounts on the internet.

“Covertness”: the average amount per deposit is much higher than that per withdrawal, typically double to eight times for personal and business deposit, and even more than twenty times for business deposit in some cases [19]. This difference is because of the limit set by FIs; a cash deposits has no limit or the limit is much higher than a cash withdrawal limit. Thus, criminals can easily conduct unauthorized large amount of transactions as if they are usual transactions.

In this paper, we propose a security measure utilizing peripheral devices to prevent unauthorized deposits with a smart card. Other logical/physical attacks such as card data skimming are out of the discussion. We focus on a deposit transaction to simplify the discussion since a deposit transaction can be a remittance transaction if the deposit account number in the transaction message is replaced with a beneficiary account number. The proposing measure needs to protect multiple transaction sub-processes in a deposit transaction from multiple types of logical attacks, and also needs to be harmonized with existing ATM operations. Since such requirements are not covered in the other paper [15], to design suitable implementation of the fundamental framework meeting those requirements is confusing due to many items to be considered. Therefore, the proposing measure also provides implementation model analysis of the fundamental framework to derive suitable implementation meeting the requirements for each defense point in a deposit transaction.

The remainder of this paper is organized as follows. Chapter 2 addresses the issues of ATM systems, operations, and existing measures. Chapter 3 presents the proposing measure. Chapter 4 describes the implementations of the proposing measure. Chapter 5 is the conclusion.
processes of an existing deposit with a smart card [20], [21]. The related logical attacks for unauthorized deposits are also illustrated in Fig. 3 and Table 1, whose details are described in the following section. The malicious device is a circuit board installed between the PC and a peripheral device to manipulate communication data on RS-232C/USB. A point of smart card transactions is that the authenticity of a transaction message is assured by a Message Authentication Codes (MACs) generated by either a smart card or the host computer. A cryptographic key for MACs is shared between a smart card and the host computer in conformity to the EMV® specification [20]. The key is also linked with the Primary Account Number (PAN) on the smart card, which is preliminarily assigned to the card by the FI to identify the user. It is supposed that the multi-vendor application includes “transaction application” (hereinafter called “transaction AP”) to process transaction messages and “cash handling application” (hereinafter called “cash handling AP”) to control a CRM. The ATM platform and the OS are omitted in the figure. An example of an existing transaction sequence is described as follows:

S1: A user inserts a smart card into the card reader in an ATM. The card reader reads a PAN from the smart card. And then the card reader transfers the PAN to the transaction AP using the CEN/XFS API.

S2: The user puts cash into the cash pocket of the CRM (Fig. 4). Cash is transported to the bill validator to verify the authenticity of the cash and to count the cash amount. The cash is further transported to the intermediate stacker. The CRM outputs the cash amount to the cash handling AP using the CEN/XFS API.

S3: The cash handling AP sends the cash amount to the transaction AP.

S4: The transaction AP creates a “transaction request message” (hereinafter called “RQMSG”) from the PAN and the cash amount for the deposit. The transaction AP sends the RQMSG to the smart card through the card reader using the CEN/XFS API.

S5: The smart card generates a MAC to the RQMSG called “MAC1” and sends the MAC1 back to the transaction AP using the CEN/XFS API.

S6: The transaction AP sends the RQMSG and the MAC1 to the host computer. The host computer verifies the RQMSG with the MAC1 and then decides whether the transaction is authorized or not.

| No | Attack objective | Attack name | Attack surface | Outline of possible logical attack |
|----|------------------|-------------|----------------|-----------------------------------|
| A1 | Manipulation of transaction request message | Malicious device for data manipulation | RS-232C/USB | A malicious device on RS-232C/USB cable manipulating a transaction request message and a cash amount for unauthorized deposit. |
| A2 | Malware for data manipulation | Malware for data manipulation | PC | Malware manipulates a transaction request message and a cash amount for unauthorized deposit. |
| B1 | Malicious device for unauthorized cash return | Malicious device for unauthorized cash return | RS-232C/USB | A malicious device on RS-232C/USB cable manipulating data and a command for unauthorized cash return despite the deposit transaction was authorized. |
| B2 | Malware for unauthorized cash return | Malware for unauthorized cash return | PC | Malware forces the ATM to return cash in intermediate stacker despite the deposit transaction was authorized. |
| C1 | Malicious device for replay attack | Malicious device for replay attack | RS-232C/USB | A cash returning command sent to the CRM is temporally held by a malicious device on the USB cable and then sent to the CRM again after a user leaves from the ATM. |
| C2 | Malware for replay attack | Malware for replay attack | PC | A cash returning command sent to the CRM is temporally held by malware and then sent to the CRM again after a user leaves from the ATM. |
S7: The host computer creates a “transaction response message” (hereinafter called “RESMSG”) and generates a MAC to the RESMSG called “MAC2”. The host computer sends them back to the transaction AP.

S8: The transaction AP forwards them to the smart card through the card reader using the CEN/XFS API.

S9: The smart card verifies the RESMSG with MAC2 and returns the RESMSG verification result to the transaction AP. The value of the verification result varies, depending on the host computer’s decision [21]. It is noted that the response verification result is plain data as the smart card and the transaction AP do not share any cryptographic key.

S10: The transaction AP provides the cash handling AP with a cash storing request if the transaction is authorized in the response verification result. Otherwise, the transaction AP provides a cash returning request.

S11: The cash handling AP sends either a cash storing command or a cash returning command to the CRM through the CEN/XFS API. The CRM transports the cash in the intermediate stacker into the cash units to store cash in the safe if the CRM receives a cash storing command. Or the CRM transports cash in the stacker back into the cash pocket to return the cash to the ATM user.

2.2 Issues of Existing Security Measures

The logical attacks for unauthorized deposit target data flow and process, which are related to ATM system structure and media such as cash and a smart card, in a deposit transaction. Security targets of an ATM system are (1) the communication between the smart card and the host computer, (2) the PC, (3) USB/RS-232c cable between the PC and a peripheral device, and (4) the peripheral devices. Since (1) and (4) are protected by tamper-proof hardware, the proposing measure considers (2) and (3), which are shown in Fig. 3 and Table 1. The tamper-proof hardware of (4) is covered in the proposed measure of the other paper [15], which is explained in Chapter 1. In the standard security measures, the CRM is secure against unauthorized physical manipulation because it is physically protected by a safe. The tamper-proof hardware of (1) is implemented in the standard security measures shown in Fig. 2. (1) is protected logically and by a tamper-proof mechanism in conformity to the EMV specifications and regulations [20], [21]. Even if S1 PAN is altered by any logical attack, such altered S1 PAN can be detected by the EMV specifications [20] since a MAC key is linked with the PAN. It is noted that “Denial of Service” attacks, such as resource excessive command sending to the CRM, are out of the discussion since they do not result in fraudulently increased balance and unauthorized cash return. Excessive resource consumption of the CRM simply results in suspension of the mechanical action.

Regarding logical attacks to (2) (3), A1 and A2 in Table 1 are attacks to manipulate an RQMSG before a smart card generates a MAC1 for the RQMSG on USB/RS-232c cable and the PC, respectively. A manipulated RQMSG and a MAC1 are sent to the host computer in order to fraudulently increase an account balance with no actual cash or less cash. And then, malicious persons withdraw cash from the fraudulently increased account using ATMs. B1 and B2 are attacks to manipulate data flow and processes when the cash stored in the intermediate stacker is transported into the cash units of the CRM after the host computer authorized the deposit transaction. These attacks force an ATM to return the cash in the stacker to a malicious user. Manipulating an RQMSG is not required in the attacks. The malicious user can increase an account balance unlimitedly by repeating a cycle of sending a verified RQMSG to the host computer and conducting unauthorized cash return after the host computer’s authorization. Such unauthorized cash return is application of an existing fraud called “Transaction Reversal Fraud” [16], in which a criminal induces a fault at an ATM during a cash dispense operation such that the transaction AP reverses the cash withdrawal transaction, i.e. retracting to debit the account, although the criminal removes dispensed cash from the ATM with some trick.

C1 and C2 are replay attacks to manipulate data flow and processes to return cash in the stacker to not a legitimate user but a malicious person when the host computer rejects the deposit transaction. The malicious person utilizes malware or a malicious device for the attacks, which temporally holds a cash returning command transferred from the PC to the CRM. Such an attack causes false trouble to the ATM to make the user recognize the ATM has trouble. And then, the malicious person steals cash returned from the ATM operated by the malware or the malicious device after the user left from the ATM. Such cash stealing is an application of an existing fraud called “Cash Trap” [17], [18] targeting cash withdrawal, in which a device inserted inside the cash dispenser traps cash before the cash is presented to the ATM user while fooling the user into thinking that the cash shutter has not opened. The average amount per deposit is much higher than per withdrawal and even more than twenty times for business deposit as described in chapter 1. The replay attacks in deposit transactions can be much more effective and efficient for criminals than Cash Trap. A replay attack to S9 in Fig. 5 are not supposed to bring fraudulent cash return since it results in “Out of service” of the ATM as system trouble. Thus the attack is omitted here.

As depicted in Fig. 5, existing measures [1], [8], [10]–[13] try to protect information property, including software and data, in the PC against A2, B2, and C2. Furthermore, the existing measures try to cryptographically protect the USB/RS232c cable from A1, B1, C1. However, it is difficult to prevent C1 with cryptographic communication since it is a replay attack. It is noted that cryptographic protection of the USB/RS-232c cable also depends on the PC’s security because the cryptographic keys are stored in the PC. In this way, the existing measures trying to protect the whole ATM system. Such measures are neither so effective nor so efficient because there is a critical issue to tightly protect the PCs containing a number of files and huge data without tamper-proof hardware as explained in chapter 1.
The tight protection of the PCs requires FIs quite heavy management workloads, and these heavy workloads may result in breaking rules, human errors, and rather poor management by ATM operation staff. A security measure should also be harmonized with interoperability and compatibility of existing ATM applications, and stable ATM system availability as a social infrastructure.

3. The Proposing Measure

3.1 Fundamental Framework of the Proposing Measure

Figure 6 shows a comparison between the objectives of existing measures and the proposing measure. As shown in Fig. 6 (a), the PC totally controls a peripheral device on the basis of a master-slave model in an existing ATM as well as other device control systems. Such a control system works securely if the PC is secure while it is not always secure in the actual situations. The objective of the existing measures is to protect information property in the PC so that the PC does not send unauthorized commands and data to the peripheral device (Fig. 6 (b)). On the other hand, the objective of the proposing measure is for a peripheral device itself to verify the authenticity of a received command and data (Fig. 6 (c)). The function of the device (i) is to extract verification information (ii) verifying authenticity of a command in order to protect property from unauthorized access. The function of the device (v) is to verify the authenticity of a command with the information received from the device (i) in order to protect property from unauthorized access. The two kinds of devices work as follows. The device (i) extracts information to protect property (vii) from the input/output data of the device (i). Then the device (i) generates verification information (ii) to verify the authenticity of a command accessing the property (iv). And then, the device (i) securely sends the information (ii) to the device (v). If the device (v) successfully verifies the authenticity of the command (iv) received from the control unit (iii) with the information (ii), the device (vi) accesses the property (vii). In this way, protecting the control unit (iii) is not a necessary condition to prevent the protected property from unauthorized access in the fundamental framework.

3.2 Implementation Model Analysis

The implementation model analysis of the fundamental framework is described here to select candidate implementation models that are consistent with existing systems, operations, and characteristics of command authenticity verification. Figure 8 outlines the implementation models of the fundamental framework. Each peripheral device consists of tamper-proof hardware as a secure domain and an existing control mechanism including firmware. The authenticity of the firmware is supposed to be assured by digital signatures stored in the tamper-proof hardware although the signatures are not shown in the figure. “Verification information extracting module” extracts verification information from not
only input/output data but also physical objects put-in/out of the peripheral device, and the physical objects (cash) are added to be applied to deposit. In Fig. 8(a), the verification information is securely transferred from the device (i) to the “command verification module” in the device (v) through the two cryptographic modules. The module verifies a command received from the control unit (iii) and outputs the verified command to the firmware of the control mechanism in the device (v). Figure 8(b) shows the implementation model 2 that the device (v) on the right side of Fig. 8(a) is split into two devices: a command verifying device and a command executing device. Figure 8(c) outlines the implementation model 3 that the device (i) and the command verifying device in Fig. 8(b) are integrated into one device. Figure 8(d) depicts the implementation model 4 that all devices in Fig. 8(b) are integrated into one device.

Table 2 summarizes the features of each implementation model in Fig. 8. The implementation model 1 and model 3 are recommended to implement the fundamental framework for deposit with a smart card. Model 4 is a
preferable model but not recommended since it is difficult to apply the model to smart card transactions. Regarding the preferable point, some SIers install a critical device, such as a secure element in the card reader, inside the safe wired with the card reader, to physically protect it from being stolen by staff and criminals even if the secure element is tamper-proof. Concerning the not recommended point, a smart card, the card reader and the CRM need to be integrated into one device in a safe to verify an RQMSG with cash amount outputted by the CRM. However, it is difficult to install a smart card in the safe since a smart card must be returned to an ATM user. The model 2 is not recommended from a viewpoint of minimizing cryptographic communication to reduce risks of cryptographic key management. When a troubled peripheral device equipped with cryptographic functions is replaced with a new device, a tightly controlled cryptographic key configuration is required because such configuration could be an attack target [22]. Some maintenance staffs try to identify a troubled device by replacing devices in an ATM with new devices one by one. Thus such tightly controlled operations for cryptographic functions should be minimized to reduce both management workloads and opportunities of malicious operations by staff [5].

It is difficult to verify the authenticity of a command transfer time to detect a replay attack in model 3. Even if a verified command is cryptographically protected in communication between peripheral devices, the command could be maliciously delayed. And the peripheral device on the right side is not equipped with any module verifying command authenticity. Thus, implementation model 3 can be applied to verify the authenticity of a command except for a viewpoint of a command transfer time. On the other hand, the implementation model 1 can verify the authenticity of a command transfer time. Since (v) peripheral device is equipped with “module verifying command authenticity”, the module can detect delay of a command transferred from the control unit by confirming time difference between the command arriving time and a reference time. The reference time can be included in (ii) information to verify the authenticity of command accessing the property.

3.3 Conditions to Prevent Logical Attacks

To securely protect deposit from A1 to C2, the following four conditions “R1 to R4” should be confirmed in secure domains of an ATM system.

(R1) The cryptographic keys of cryptographic USB/RS-232c communication in an ATM should be protected in secure domains of an ATM.

(R2) The cash amount received from CRM is equal to (Cash amount included in RQMSG).

(R3) CRM accepts a cash returning command if the transaction is rejected by the host computer, otherwise, CRM rejects the cash returning command.

(R4) (CRM’s receiving time of a cash returning command) - (card reader’s receiving time of the RESMSG verification result) is less than a threshold.

R1 is an existing key management condition to protect USB/RS-232c communication from logical attacks to them. R2 is the first defense point to protect an RQMSG from being manipulated before the RQMSG is cryptographically protected with MAC1. R2 is used to confirm consistency between the cash amount outputted from the CRM and the cash amount included in the RQMSG. R3 is the second defense point to prevent an unauthorized cash returning command by confirming consistency between a cash returning command and the RESMSG verification result. R4 is the third defense point to protect a cash returning command from replay attacks. R4 is used to confirm whether the cash returning command received by the CRM is significantly delayed or not. Table 3 shows the correspondence relation between the conditions and prevented logical attacks. In the proposing measure, the secure domains are created in peripheral devices to meet R1, which was proposed in the other paper [15], we focus R2, R3, and R4 to prevent logical attacks specific to unauthorized deposit.

| Conditions | A1 | A2 | B1 | B2 | C1 | C2 |
|------------|----|----|----|----|----|----|
| R1         | ✓  |    | ✓  |    | ✓  | ✓  |
| R2         | ✓  |    | ✓  |    | ✓  | ✓  |
| R3         |    | ✓  |    | ✓  | ✓  | ✓  |
| R4         |    |    |    |    | ✓  | ✓  |

4. Implementations

4.1 Implementation Outline

The outline of the implementation model 1 and the implementation model 3 to verify the conditions “R2, R3 and R4” is depicted in Fig. 9. The implementation model 3 can be applied to R2 because R2 is not a condition to verify a command transferring time. Secure elements as tamper-proof hardware are supposed to be installed in the proposing card reader and the proposing CRM. Figure 9 (a) shows the implementation model 1 to verify a MAC generating a command for a REQMSG with R2 in the card reader using the cash amount from the CRM. The CRM, the card reader, the put-in cash, the MAC generating command for a REQMSG, and the MAC key in the smart card correspond to the device (i), (v), the physical objects, the command, and the protected property in Fig. 8 (a), respectively. Since the authenticity of a command can be replaced with the authenticity of a REQMSG, the following description focuses on the REQMSG. If the authenticity is verified, the verified REQMSG is forwarded to the smart card to generate a MAC for the verified REQMSG. Figure 9 (b) shows the implementation model 3 to verify a REQMSG with R2 in the CRM using the cash amount stored in the CRM. “Cash amount
OGATA et al.: AN ATM SECURITY MEASURE TO PREVENT UNAUTHORIZED DEPOSIT WITH A SMART CARD

Figure 9 illustrates the implementation outline of the proposing measure.

Fig. 9 Implementation outline of the proposing measure.

Fig. 10 Implementation data flow of the proposing measure.

4.2 Detailed Data Flows of Implementation

The data flow of the implementation examples of the proposing measure is described in this section. There is not any physical communication cable such as USB/RS-232c cable directly connecting between existing peripheral devices. A cryptographic communication between the peripheral devices is implemented by utilizing the existing USB/RS-232c cable between a peripheral device and the PC. “Data Transfer Library” (hereinafter called “DTL”) is newly introduced in the PC to simply provide a communication path transferring encrypted data among the peripheral devices. Even if the DTL is infected with malware, the integrity of encrypted data transferred in the DTL is still assured. The DTL is supposed to be installed in a layer below the CEN/XFS APIs. Figure 10 illustrates the data flow of the implementation outlines shown in Fig. 9. The cryp-
tographic key management and a session creation for each cryptographic communication are supposed to conform to either the PCI requirements [23]–[25] or the EMV specifications [20] to meet confidentiality, integrity, and authenticity. A session of each cryptographic communication is supposed to be preliminarily created. The detailed process flows of each implementation of Fig. 10 are described as follows. Only modified processes and the related processes are explained here. Figure 10 (a) shows the modified data flow corresponding to Fig. 9 (a).

S2: The CRM counts a cash amount and returns it to the cash handling AP. And then the CRM stores the cash amount in it.

S4: The transaction AP sends an RQMSG to the card reader using a CEN/XFS API and the DTL.

S4-1: Once the DTL receives the RQMSG, the DTL requests the CRM to send the encrypted cash amount and forwards it to the card reader. The card reader decrypts the encrypted cash amount and verifies the RQMSG with R2 using the cash amount.

S4-2: If the RQMSG is successfully verified, the card reader forwards the RQMSG to the smart card.

Figure 10 (b) shows the modified data flow corresponding to Fig. 9 (b). S2 is omitted since it is the same as in Fig. 10 (a).

S4: The transaction AP sends an RQMSG to the CRM using a CEN/XFS API and the DTL. The CRM verifies the RQMSG with R2 using the cash amount stored in the CRM.

S4-3: The DTL requests the CRM the encrypted RQMSG and forwards it to the card reader.

S4-4: The card reader decrypts the encrypted RQMSG and forwards it to the smart card.

Figure 10 (c) shows the modified data flow corresponding to Fig. 9 (c).

S8: The card reader receives the RESMSG and MAC2.

S9: The card reader receives a RESMSG verification result from the smart card, and stores a RESMSG verification result receiving time in it. And then the card reader generates an authorization/rejection flag from the result and stores the flag in it.

S11: The cash handling AP sends either a cash storing command or a cash returning command to the CRM through a CEN/XFS API and the DTL following to the request from the transaction AP.

S11-1: Once the DTL receives either command, the DTL requests the card reader to send the encrypted authorization/rejection flag and the encrypted RESMSG verification result receiving time and then forwards them to the CRM. The CRM decrypts the encrypted data and verifies the cash storing/returning command with R3 and R4 using the decrypted data. If the command is successfully verified, the CRM executes the verified command.

4.3 Architecture of the Proposing Peripheral Devices

The architecture examples of the proposing peripheral devices are depicted in Fig. 11. In general, an existing card reader is equipped with a slot to install a secure element for mutual authentication between a smart card and a terminal. A secure element to achieve the proposing measure can be installed in the slot. The contact point I/F (interface) to communicate with a smart card is also equipped with a secure element. Those two secure elements are cryptographically connected in order to protect contents transferred from a card to the PC and from the PC to the card even in the card reader. Additionally, the firmware in the controller is also supposed to be protected from unauthorized manipulation with digital signatures installed in the secure element. The firmware running on the RAM in the controller is supposed to be still secure by self-tests with the digital signatures. For example, the firmware hash is calculated once every day in the controller. The hash is transferred to the secure element and verified with the digital signatures. Such structure and processes are practical since the PCI requirements [23], [25] define similar requirements.

An existing CRM is equipped with a serial interface to expand the functions in many cases. A circuit board implementing a secure element can be installed on the serial interface. The firmware in the controller is also supposed to be protected from unauthorized manipulation even during running on the RAM with digital signatures installed in the secure element as well as the card reader. Furthermore, the whole CRM is protected from unauthorized physical accesses by a tightly controlled safe. Thus the firmware is logically and physically protected.

4.4 Validation of the Proposing Measure through Implementation

In this section, it is described from qualitative and quantitative viewpoints with the implemented measure, that the proposing measure in Chapter 3 prevents the logical attacks
effectively in an actual ATM operation. Regarding the qualitative viewpoint, preventing the logical attacks effectively requires meeting the conditions of (A), (B), and (C) as described in the other paper [15]. These conditions were derived from the issues of ATM systems and operations, and show that a measure does not overload ATM operation. A practical security effect of a measure without tamper-proof hardware highly depends on operational protection, namely management. And the security effect can be largely decreased if the measure requires heavy management workloads as explained in Chapter 1. The proposing measure for cash withdrawal transactions with a smart card was shown to meet three conditions in the other paper. The proposing measure for deposit transactions with a smart card is shown to meet the conditions in the following description.

(A) A security measure should not significantly impact management workloads of existing ATM operations.

The proposing measure does not significantly impact the management workloads for the PCs containing so many files to protect since the measure relies on the peripheral devices equipped with tamper-proof hardware. The number of firmware in the peripheral devices is typically one or two and much smaller than that of executable files in the PC. Such a small number of firmware can be protected with existing secure elements. In this way, tight protection of the PCs causing quite heavy management workloads is not a critical issue to prevent unauthorized deposit.

(B) A security measure should not significantly impact ATM system availability.

The proposing measure does not rely on the PC but on the peripheral devices equipped with tamper-proof hardware. Thus frequent OS updating/hardening for a security patch, which would significantly impact ATM system availability, is not a necessary condition in the proposing measure. FIs can take enough time to comprehensively test so many software components in the PC before releasing them to prevent occasional system troubles while mitigating zero-day attack risks of unauthorized deposit.

(C) The logical attacks cannot be successful even though the integrity of all software related to device control commands is compromised.

The proposing measure can prevent the logical attacks for unauthorized deposit without relying on the integrity of all software of the PC. Even if the integrity of the DTL is compromised in the PC, the logical attacks cannot still be successful because the DTL just provides a communication pass to transfer encrypted data. The proposing measure can work as a defense in depth in cases that the PC is compromised.

The requirements of the EUROPOL’s guidance [1] as representative of existing measures do not meet the three conditions as explained in the other paper [15].

Concerning the quantitative viewpoint, let us estimate each measure’s annual numbers of potential unauthorized access to files to protect in order to compare the practical effect of the EUROPOL’s guidance and the implemented proposing measure. These numbers are correlating to practical management workloads to prevent unauthorized access to the files in ATM operation. A comparison of the initial costs is omitted here since the cost of the EUROPOL’s guidance is much more than the proposing measure. Costly measures: encrypted communication between the PC and the CRM, whitelisting-based anti-malware, sand boxing, hard disk encryption, various ATM monitoring systems and so forth are required in the EUROPOL’s guidance. Regarding the proposing measure, the proposed card readers and CRMs based on existing devices, and the secure elements can be developed and provided at a reasonable cost as described in the other paper [15]. Either Fig. 10 (a) and (c), or Fig. 10 (b) and (c) can be used for the estimation. The assumption is that the number of ATMs is three thousand and that cash replenishment/collection is conducted once a week, i.e. 52 times per year for each ATM. The number of the executable files to protect in the PC is twenty thousand for the EUROPOL’s guidance, while the number of the files to protect in the peripheral devices is two for the proposing measure, namely, a file of firmware is implemented for the card reader and the CRM. The DTL is not counted as described in the condition (C). The result is shown in Table 4. The number is 3,120 million for the EUROPOL’s guidance while zero for the proposing measure. “Number of executable files to protect by management” in Table 4 is zero for the proposing measure since the firmware is protected by not management but tamper-proof hardware. In this way, our proposal is much better than the EUROPOL’s guidance.

5. Conclusion

In this paper, we propose an ATM security measure utilizing the peripheral devices to prevent unauthorized deposits with a smart card. The measure needs to protect multiple transaction sub-processes in a deposit transaction from multiple types of logical attacks and needs to be harmonized with ex-
isting ATM operations. Thus, the measure provides an implementation model analysis of the fundamental framework of the measure to achieve suitable implementation for each defense point in a deposit transaction. In detail, the features of each implementation model are compared and candidate models are selected to conform to existing systems and operations. And then, the suitable implementation models can be selected among the candidates to meet the requirements at each defense point. Two types of measure implementation are proposed in the sub-process before communication between the smart card and the host computer, and one type of measure implementation is proposed in the sub-process after the communication. As the proposing measure relies on the security of the peripheral devices, it can work as a defense in depth when the PC is compromised. The proposing measure can also meet the three conditions so as not to impose on FIs heavy burden to tightly control the PCs. We expect that the primary concept of the proposing measure can also be applied to ATM transactions with magnetic stripe cards, contactless cards, smartphones, and QR codes. Securely protecting the new types of transactions are remaining issues to be addressed in future works.

*1 Windows is either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

*2 EMV is a trademark in the U.S. and other countries and an unregistered trademark elsewhere. The EMV trademark is owned by EMVCo.

References

[1] European law enforcement agency, “Guidance and recommendations regarding logical attacks on ATMs,” https://www.ncr.com/content/dam/ncrcom/content-type/brochures/EuroPol-Guidance-Recommendations-ATM-logical-attacks.pdf

[2] Symantec, “Backdoor.Padpin,” Press Release, Symantec Security Response, Oct. 2014, https://www.symantec.com/security_response/writeup.jsp?docid=2014-051213-0525-99&tabid=2

[3] Kaspersky Lab, “Tyupkin Virus (Malware) | ATM Security,” https://www.kaspersky.com/resource-center/threats/tyupkin-malware-atm-security-malware

[4] Symantec Official Blog: Backdoor.Ploutos Relaoded – Ploutos Leaves Mexico, http://www.symantec.com/connect/blogs/backdoorploutos-relaoded-ploutos-leaves-mexico

[5] The Times of India, “ATM jackpot with malware,” TIMES NATION | Politics & Policy, May 2015, http://www.pressreader.com/india/the-times-of-india-mumbai-edition/2015/05/09/282003260992233

[6] EUROPOL, “27 arrested in successful hit against ATM Black Box attacks,” Press Release, May 2017, https://www.europol.europa.eu/newsroom/news/27-arrested-in-successful-hit-against-atm-black-box-attacks

[7] The European Association for Secure Transactions (EAST), “EAST reports 2016 crime stats for Europe’s ATMs: black box attacks up 287 percent,” April 2017, https://www.atmmarketplace.com/news/east-reports-2016-crime-stats-for-europe-s-atms-black-box-attacks-up-287-percent/

[8] NCR, “ATM security exploit a inning attack vectors, defense strategies and solutions,” 2018, https://www.ncr.com/content/dam/ncrcom/content/whitepapers/12518fin-b-atm_security_attack_vectors_and_solutions_update-fin-web.pdf

[9] G.A. Aad, NCR, “Security and customer experience in self service,” Forth Information Security Conference for the Financial Sector - Qatar Central Bank, Nov. 2017, https://www.isfs.qcb.gov.qa/en/maroon/home/Presentation/Gaby%20Aad%20NCR.pdf

[10] China Zhijian Publishing House, “GA 1280-2015, Security requirements for automatic teller machines,” (in Simplified Chinese), https://www.spc.org.cn/online/GA%252F201280-2015/

[11] ATM marketplace, “ATMs left behind as Windows XP support ends,” April 2014, http://www.atmmarketplace.com/articles/atms-left-behind-as-windows-xp-support-ends/

[12] J. Bräuer, B. Gmeiner, and J. Sametinger, “A Risk Assessment of Logical Attacks on a CEN/XFS-based ATM Platform,” International Journal on Advances in Security, vol. 9, no. 3&4, pp. 122–132, ISSN 1942-2636, Dec. 2016.

[13] S. Kai, T. Ishiakawa, H. Ogata, N. Miyamoto, and T. Sanada, “Accelerating Global Business through ATMs Security Practices,” IPSJ Digital Practice, vol.9, no.3, pp. 700–715, July 2018, https://ipsj.ipsg.nii.ac.jp/el?action=pages_view_main&active_action=repository_view_main_item_detail&item_id=190389&item_no=1&page_id=1&block_id=8

[14] CEN, “Extensions for Financial Services (XFS) interface specification Release 3.30 - Part 1: Application Programming Interface (API) - Service Provider Interface (SPI) - Programmer’s Reference,” European Committee for Standardization, Aug. 2015, ftp://ftp.cen.eu/CWA/CEN/WS-XFS/CWA16926/CWA16926-1.pdf

[15] H. Ogata, T. Ishiakawa, N. Miyamoto, and T. Matsumoto, "An ATM security measure for smart card transactions to prevent unauthorized cash withdrawal," IEICE Trans. Inf. & Syst., vol.E102-D, no.3, pp.559–567, http://search.ieice.org/bin/summary.php?e=d=2013-d_559&category=E&year=2015&lang=E&abst=

[16] NCR, “Transaction Reversal Fraud - Global,” July 2018, https://www.ncr.com/content/dam/ncrcom/content-type/brochures/NCR%20Security%20Alert%20-%202018-06%20Transaction%20Reversal%20Fraud.pdf

[17] NCR, “Cash Trapping “Type 1” Attacks in Spain,” Dec. 2016, https://www.ncr.com/content/dam/ncrcom/content-type/brochures/nr_security_alert_-_2016-14_cash_trapping_in_spain_0/pdf

[18] The European ATM Security Team, "European ATM crime report 2014 period: January to December,” https://www.association-secure-transactions.eu/files/EAST-ATM-Crime-Report-2014.pdf

[19] RBR, “Deposit Automation and Recycling 2016,” https://www.rhcdn.com/research/deposit/

[20] EMVCo, LLC, “EMV Integrated Circuit Card Specifications for Payment Systems Book 2 Security and Key Management Version 4.3,” Nov. 2011, https://www.emvco.com/terms-of-use/?wp-content/uploads/documents/EMV_v4.3_Book_2_Security_and_Key_Management_20120607061923900.pdf

[21] EMVCo, “EMV Integrated Circuit Card Specifications for Payment Systems Book 3 Application Specification, Version 4.3,” Nov. 2011, https://www.emvco.com/wp-content/uploads/documents/EMV_v4.3_Book_3_Application_Specification_20120607062110791.pdf

[22] IOActive, Inc., “IOActive Security Advisory,” July 2017, https://ioactive.com/pdfs/ATM-security-advisory_FINAL_v4-davis_cm.pdf

[23] PCI SSC, “Payment Card Industry (PCI) PIN Transaction Security (PTS) Point of Interaction (POI) Modular Security Requirements Version 5.1,” March 2018, https://www.pcisecuritystandards.org/documents/PCIPTS_PoISecureRequirements_v5.1.pdf

[24] PCI SSC, “Payment Card Industry (PCI) PIN Transaction Security (PTS) Hardware Security Module (HSM) Modular Security Requirements Version 3.0,” June 2016, https://www.pcisecuritystandards.org/documents/PCI_HSM_Security_Requirements_v3.0_final.pdf

[25] PCI SSC, “Payment Card Industry (PCI) Point-to-Point Encryption: Solution Requirements and Testing Procedures Version 3.0,” December 2019, https://www.pcisecuritystandards.org/documents/P2PE_v3.0_Standard.pdf
Hisao Ogata received the B.S. and M.S. degrees from Kyusyu University in 1987 and 1989, respectively. During 1989-2004, he worked for Hitachi, Ltd. researching neural networks and pattern recognition. He has been working for Hitachi-Omron Terminal Solutions, Corp. developing biometrics and ATM security since 2004. He is a Ph.D. candidate of Yokohama National University, a member of IEICE and ISO/IEC JTC1 SC37.

Tomoyoshi Ishikawa received the B.S. degree in Informatics and Mathematical Science from Kyoto University in 2001. During 2001-2004, he worked for OMRON Corporation, developing ATM software. He has been working for Hitachi-Omron Terminal Solutions, Corp. designing ATM software and its security since 2004. He is a member of CEN Workshop on Extensions for Financial Services (CEN/XFS).

Norichika Miyamoto received the B.S. and M.S. degrees from Kyoto University in 1984 and 1986, respectively. During 1986-1987, he worked for Nippon Kokan Corporation, developing process control system for steel manufacturing. During 1987-2004, he worked for OMRON Corporation, developing Unix workstation software and operating system for multiprocessor system. He has been working for Hitachi-Omron Terminal Solutions, Corp. developing ATM devices and constructing overseas business strategy since 2004.

Tsutomu Matsumoto is a professor of the Graduate School of Environment and Information Sciences, Yokohama National University and directing the Research Unit for Information and Physical Security at the Institute of Advanced Sciences. He received Doctor of Engineering from the University of Tokyo in 1986. Starting from Cryptography in the early 80’s, he has opened up the field of security measuring for logical and physical security mechanisms. Currently he is interested in research and education of Embedded Security Systems such as IoT Devices, Network Appliances, Mobile Terminals, In-vehicle Networks, Biometrics, and Artifact-metrics. He is serving as the IEICE Technical Committee on Hardware Security, the chair of Japanese National Body for ISO/TC68 (Financial Services), and a core member of the Cryptography Research and Evaluation Committees (CRYPTREC). He was a director of the International Association for Cryptologic Research (IACR) and the chair of the IEICE Technical Committee on Information Security and served as an associate member of the Science Council of Japan (SCJ). He received the IEICE Achievement Award, the DoCoMo Mobile Science Award, the Culture of Information Security Award, the MEXT Prize for Science and Technology, and the Fuji Sankei Business Eye Award.