Enhancing EMV Online PIN Verification

Danushka Jayasinghe*, Raja Naeem Akram*, Konstantinos Markantonakis*, Konstantinos Rantos† and Keith Mayes*
*Smart Card Centre, Information Security Group,
*Royal Holloway, University of London, Egham, Surrey, UK, TW20 0EX
*Email: {Danushka.Jayasinghe, R.N.Akram, K.Markantonakis, Keith.Mayes}@rhul.ac.uk
†Computer and Informatics Engineering Department,
†Eastern Macedonia and Thrace Institute of Technology, Greece
†Email: krantos@teiemt.gr

***Abstract*— EMV (Europay MasterCard Visa) is a globally accepted standard for chip card-based payment transactions, which benefits from the intrinsic security characteristics of chip cards. The EMV specification is relatively flexible and can be deployed in both online and offline card acceptance environments. In the offline environment, payment terminals and cards only communicate with each other in order to approve/decline the payment transactions, whereas in the online environment authorisation entities are also involved in the overall process. An authorisation entity can either be the Card Issuing Bank (CIB) or the payment scheme operator (e.g. Visa, Master-Card). Aside from the transaction authorisation, the EMV specifications define offline-PIN verification as one of the main cardholder verification methods. However, in an online authorisation environment, the PIN verification process is referred to as Online-PIN Verification (OPV). This process is the main focus of this paper. We discuss the OPV process that has placed indelible trust assumptions on the intermediary entities (subcontractors) between a payment terminal and a scheme operator/CIB. When this trust (assumption) is scrutinised, there is a potential attack scenario that an adversary can use to gain access to PIN data. This information can be used by an adversary to carry out an online PIN approved transaction without the involvement of the genuine cardholder but with the correct PIN. We then propose three solutions based on the existing OPV process as potential countermeasures that are then implemented to measure any incurred performance penalties and subjected to mechanical formal analysis using CasperFDR.**

***Keywords*—EMV, Online PIN Verification, Security, Cryptography, Implementation, Performance, CasperFDR.**

I. Introduction

Smart card-based payment (debit and credit) cards superseded magnetic stripe cards to reduce card-based fraud [1, 2]. In smart card-based systems [3], the sensitive data related to the consumer and his/her card is securely stored in a tamper-resistant chip [1, 4]. In addition, in this payment system, the association between the payment card and its authorised cardholder can also be established using a Personal Identification Number (PIN). Theoretically, only the genuine cardholder, the relevant smart card, the Card Issuing Bank (CIB) and the Scheme Operator (SO)[1] in stand-in process[2] should know the PIN.

At the Card Terminal at Point of Sales (CTPOS)[3], the cardholder inserts her smart card into a card terminal, after which the card and the terminal communicate with each other and try to determine common parameters based on their individual risk assessments in order to perform the transaction. If the common parameters require cardholder authentication, then the terminal will ask the consumer to enter her PIN. The input value can be validated either by the smart card or by the authorisation entity (i.e. scheme operator or CIB). If the PIN is verified successfully then the card-holder is authenticated and the transaction can proceed to the next step. Knowledge of the PIN is regarded as an authorisation for the transaction from the cardholder. It is an alternative to the cardholder's signature used to authorise a payment to the merchant. Payment card fraud can be reduced, to some extend, through the use of a smart card and it's corresponding PIN [2, 5].

The security of the PIN is paramount from both the cardholder's and the payment environment's point of view, as explained further in section I-A. For this reason payment terminals are designed to be secure and tamper-resistant in order to safeguard transaction details including the PIN. The PIN is also used in order to provide some assurance that the genuine cardholder was present at the time of the transaction. Of the two PIN verification scenarios that we discussed previously (PIN verified by the card or by the authorisation entity), the latter has a comparatively higher assurance value. The process in which the PIN is verified by the authorisation entity is referred as Online-PIN Verification (OPV), which is the core focus of this paper.

A. Problem Statement

In the OPV process, between the CTPOS and the scheme operator/card issuing bank, there might be a number of entities that handle the communication. The EVM Card Specification Book 4 recommends that this communication should be adequately protected [6]. Understanding the OPV process deployed in the ATM transactions [7] and elaborated in [8]–[10], it is clear that the payment terminal requests the PIN from the cardholder, then encrypts it and sends it to the authorisation entity. According to the explanation in section II-A, it can be assumed that during a transaction at a payment terminal the PIN is encrypted using a symmetric key (which can be a

[1]Scheme Operator is a trusted broker that manages the payment scheme. (e.g. Visa or MasterCard).

[2]Stand-in processing is a service where a scheme operator authenticates an EMV transaction on behalf of a card issuing bank.

[3]This is a payment terminal at the merchants premises that accepts card-based payments and it is referred to as CTPOS in this paper. For the purposes of this study, a CTPOS is not an Automated Teller Machine (ATM).

session key with a limited lifespan) that it shares with the next point of communication, which might not be the authorisation entity. The next point deploys a key translation mechanism and forwards the message to the next stage in the journey to the authorisation entity.

Furthermore, the transaction authorisation message[4] generated by the card, whose purpose is to get an online transaction approval from the authorisation entity, has no binding with the OPV process. This message consists of a number of EMV tags that are then encrypted using a shared key (between the card and the authorisation entity). One of the tags that make up the ARQC is the Cardholder Verification Method (CVM) mentioned in EMV Book 4, which indicates what method is used to verify the cardholder [6]. The CVM is a three-byte tag, with each byte representing the CVM performed, CVM conditions and CVM results [6, see: p49]. The only information in the CVM bytes regarding the OPV is a single binary value set to 1 if CVM was performed [11, see: p162]. In the CVM or ARQC, there is no tag that binds the OPV process and the respective ARQC. Also, during the OPV, only the payment terminal handles the PIN and the card is not informed of the PIN value entered on the terminal. Therefore, if an adversary compromises one of the intermediate entities that perform the key translation between the payment terminal and the authorisation entity, he can obtain the PIN number along with other transaction details. To perform this attack, the adversary will observe all the OPV messages that include the PINs and associated Primary Account Numbers (PAN). This will enable the adversary to perform the OPV-based transaction at a merchant's premises with a stolen card for which the adversary has previously obtained the relevant PIN.

Therefore, an attacker might be able to perform an EMV transaction that requires online approval at a CTPOS. The notion that an adversary can compromise an intermediary is not hypothetical or far-fetched, as indicated by these reports [12]–[14] and for the overall security of the payment scheme, such scenarios should not be underestimated.

B. Contributions

Based on the problem discussed in the previous sections, the following solutions are introduced to tackle the aforementioned problems:

1) To protect the PIN we have proposed an enhanced OPV process using:
   a) Card-based solution with symmetric cryptography.
   b) Card-based solution with asymmetric cryptography.
   c) Payment terminal-based solution with asymmetric cryptography.
2) Binding each OPV with the respective ARQC (online transaction authorisation).

C. Structure Of The Paper

The paper opens with a brief discussion of the EMV Online-PIN Verification process in section II. Section III details the assumptions regarding the payment networks operating environment, attacker's capabilities, and potential attack vectors that can be used by an adversary. In section IV, we provide three potential countermeasures to the attack vectors discussed in section III. We analyse our countermeasures in section V. Finally, in section VI we provide concluding remarks and potential future research directions.

[4]In the EMV specifications, the transaction authorisation message is referred to as an Authorisation Request Cryptogram (ARQC).

II. EMV Online PIN Verification

EMV supports both offline and online PIN verification methods. In offline PIN verification the cardholder PIN is sent to the card either in plaintext or enciphered format to be verified. In contrast to this, during an Online-PIN Verification (OPV) the PIN needs to be sent to the authorisation entity (e.g. scheme operator or CIB) for verification.

A. CTPOS Online PIN Verification Process

The EMV specifications [6, 11, 15, 16] do not specify the OPV process that may be supported by CTPOS devices. Similarly, there are, to the best of the authors knowledge, no publicly available documents that detail online PIN verification in CTPOS. However, literature in the public domain such as [8]–[10] highlights how OPV as a Cardholder Verification Method (CVM) is carried out at an Automated Teller Machine (ATM) during an ATM transaction. Considering how OPV is processed in ATM transactions and according to our understanding gained from publicly available documents such as [7], the process mentioned below outlines how OPV is carried out in a CTPOS transaction in the current architecture.

In OPV, the communication path between the payment terminal and the authorisation entity (i.e. SO/CIB) may consist of numerous intermediaries such as; the acquirer's subcontractors who operate CTPOS devices, third party Payment Terminal Operators (PTO) and other entities that engage in key translation (where messages are enciphered and deciphered from node to node). The current architecture has placed an indelible trust assumption on the intermediaries involved. Most of these intermediaries are bound by contracts with either the acquirer or SO/CIB, yet it is questionable whether this is sufficient to let intermediaries handle sensitive data related to the cards and cardholders. The EMV specification [6] states that "When the applicable CVM is online PIN, the Interface Device (IFD) shall not issue a Verify command. Instead the PIN pad shall encipher the PIN upon entry for transmission in the authorisation or financial transaction request".

This indicates that the PIN is sent in encrypted format from the CTPOS onwards but the specification does not mention PIN encipherment for OPV. Neither does it mention whether the CTPOS encrypts the PIN with a cryptographic key only shared with the authorisation entity or with the next entity on the communication path. The CTPOS devices are deployed by Payment Terminal Operators (PTO) as discussed in our operating environment in section III-A. PTOs can be merchants' acquirers or third parties. A single third party may deploy a significant number of CTPOS devices at different merchants and manage them. Depending on area to area and country to country, these third parties may differ and it is not practically feasible for issuers to get in contact with all the third parties globally, in order to share secret keys with each other. Considering the number of different merchants' acquirers, subcontractors, third parties and the number of

CTPOS devices, an authorisation entity sharing a specific key with each individual CTPOS would be logistically impractical. After all, one of the objectives of introducing EMV, was to achieve the interoperability between different entities without prior business relationships.

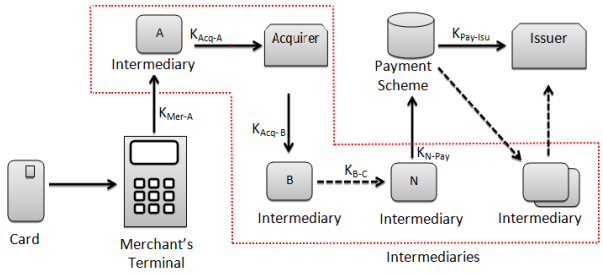

Fig. 1. Online PIN Verification Message Flow

Due to the need for these intermediaries to forward a cardholder's PIN in encrypted format in an OPV process from one intermediary to another until the scheme operator or CIB is reached, it can be assumed that a key translation mechanism similar to the one used in ATMs is used in OPV. A notable difference is that the ATM network is more trusted and the ATMs themselves would also be considered trusted. The process is carried out by intermediaries sharing different cryptographic keys with each other as illustrated in Figure 1. Considering the communication link between intermediary A and the issuer for instance, the PIN block which contains the PIN is first enciphered by intermediary A using a shared key K_{Acq-A} before it is send to the acquirer. Once received, acquirer deciphers the message to obtain the PIN block and subsequently enciphers it using the shared key K_{Acq-B} before forwarding to B. This process continues with each intermediary node until the message is received by the CIB.

III. Potential Concerns

In this section, we begin the discussion by describing the operating environment and associated assumptions. Subsequently, we discuss the attacker's capability, then examine two potential risk scenarios.

A. Operating Environment & Assumptions

Figure 2 shows an operating environment that is a representation of a payment network. A Payment Terminal Operator (PTO) might issue (or lease) their payment terminals to a number of customers (merchants). The PTO can be an acquiring bank but in our operating environment we assume two scenarios: the PTO is a third party that manages the terminals or it is an acquiring bank. The choice of who is the PTO does not affect our operating environment and the risk scenarios discussed later. There can be some additional nodes between the PTO and the scheme operators. One of these additional nodes must be the acquirer's bank (in the event that the PTO is a third party) or any additional number of entities that the communication has to go through before arriving with the scheme operator.

The scheme operator has a direct communication link with the CIB. Each of the arrows shown in the figure 2

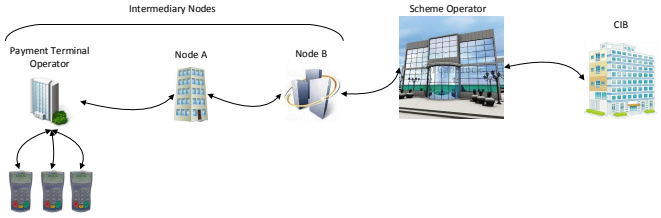

Fig. 2. OPV Operating Enviroment

connecting two entities represents the communication link and the arrowheads show the direction of communication. In addition, each of the arrows has a different session key used to secure the communication link. Each intermediary node and scheme operator will perform key translation until the OPV message is delivered to the CIB[5].

In this study, we consider that the smart cards, payment terminals, scheme operators and CIB are secure and trusted, whereas the intermediary nodes that might have a connection to the public internet have the potential to be compromised. Recently, we have seen reports in which banking sector services were successfully infiltrated by adversaries [13]. Therefore, the assumption that intermediary nodes can be compromised is reasonable [12]–[14]. Taking this operating environment and our assumptions regarding the intermediary nodes and other entities, we expand the discussion to the capabilities of our adversary in the next section.

B. Attacker's Capability

For the purposes of this discussion, the capabilities of an adversary are given below:

1) Has the capability to compromise any of the intermediary nodes.
2) Has the capability to access the OPV communication in plaintext on the compromised intermediary node. As in the operating environment discussed in section III-A, individual intermediary nodes perform a key translation process, which in essence decrypts the ingress message and then encrypts, with a new key, the egress message.
3) Can not break the standard[6] (strong) encryption algorithms.
4) Can not compromise the smart cards, payment terminals, scheme operator or CIB.
5) Might collude with other adversaries that steal smart cards from genuine cardholders.

Based on the capabilities listed above, we describe two risk scenarios in the subsequent section.

C. Two Potential Risk Scenarios

Based on the payment-networks operating environment, our assumptions and the adversary's capabilities, we discuss two potential risk scenarios that might result in a successful

[5]In certain configurations, the OPV validation can be performed by the scheme operator but in this paper we are not considering that option. However, this does not have any effect on the risk scenarios.

[6]Standardised in the relevant up to date statement

compromise of the OPV and online transaction authorisation. In both of these scenarios and in current transaction processes, it is challenging to detect a fraudulent transaction due to the adversary compromising the intermediary entity or a genuine cardholder claiming that he did not perform the transaction.

1) Correct PIN in OPV Message: The adversary has complete access to an intermediary node in our operating environment and he is observing all messages passing through it. In this risk scenario, the compromise of a payment transaction is carried out as follows:

1) The adversary observes the communication passing through the compromised intermediary node and builds a database of PIN numbers. This database contains the Primary Account Number (PAN) and associated PIN.
2) A malicious accomplice $\mathcal{M}$ of the adversary steals a smart card. The adversary matches the cards PAN with the database. If a match is found then the adversary and the accomplice know the associated PIN. Before the card gets blocked by the CIB, the accomplice presents the card to a (genuine) payment terminal, enters the correct PIN and can perform online or offline payment transactions.

This issue is exacerbated if the payment transaction is based on a magnetic stripe card and PIN. As in this case, the adversary gets most of the necessary information to create a clone magnetic stripe card and correct PIN. The clone card can then be used in countries where magnetic stripe cards are still in use.

2) OPV Response Message: In this risk scenario, a payment transaction is compromised in the following manner:

1) A malicious accomplice $\mathcal{M}$ of the adversary steals a smart card and then presents this card to a payment terminal that uses an intermediary node that is under total control of the adversary.
2) $\mathcal{M}$ selects the payment terminal in a manner that will opt in for the OPV process.
3) The payment terminal requests the cardholder ($\mathcal{M}$) to enter his PIN. $\mathcal{M}$ enters any random sequence at the payment terminal. The terminal then encrypts this entered PIN and sends it to the authorisation entity over the network.
4) The adversary captures this message. Whether it allows the message to go forward to the authorisation entity or not makes no difference[7]. The adversary then replays a successful OPV verification response message generated by the CIB (observed in previous genuine transactions) back to the terminal. The adversary has observed this successful message in previous runs of the OPV process and can just replay it to the payment terminal.
5) The payment terminal will receive a successful PIN verification message and proceeds to request the smart card to generate an online transaction authorisation message (i.e. ARQC) in the 1^{st} GENERATE AC command [16]. The card generated ARQC is then sent to the authorisation entity. The authorisation entity verifies the ARQC and does a credit check on the users account. If satisfied, the authorisation entity generates an Authorisation Response Code (ARC). The ARC is then Xored with the ARQC and then enciphered using the session key shared with the card to construct an Authorisation Response Cryptogram (ARPC) to approve the transaction. The ARPC is then sent back to the CTPOS. The ARPC could also be in the form of a Message Authentication Code (MAC).
6) The CTPOS forwards the ARPC to the card and request an outcome in the 2^{nd} GENERATE AC command [16]. The card after verifying the valid ARPC, generates a Transaction Certificate (TC) and sends it to the CTPOS.
7) CTPOS now accepts the card transaction and either sends the TC straight for payment processing or stores it for payment processing at a later time. The attack is possible due to lack of strong binding between the OPV and the ARQC.

[7]Authorisation entities (e.g. scheme operators and CIBs) do not link the OPV process with the online transaction authorisation (i.e. ARQC [6])

IV. Proposed Solutions

We propose three solutions that address the aforementioned concerns in section III and guarantee end-to-end security of OPV between the payment card and the CIB. Furthermore, the solutions introduce minimal or no changes to the intermediaries involved in the current EMV architecture between the payment terminal and scheme operator/CIB. To meet our objective, the changes we propose are only made to the payment cards, the CIB who has full ownership of the issued payment cards and their back-end authorisation systems, and to the CTPOS devices.

This makes the OPV process more secure within the threat model we discussed in section III-C by not needing a strong trust assumption for the involved intermediaries. The solutions are categorised into card-based and terminal-based solutions depending on which entity the PIN block encipherment occurs in during OPV.

There are no details provided in the EMV specifications in regard to the PIN block construction and encipherment for OPV [6, 11, 15, 16]. However, the process of PIN block construction and encipherment that happens at the CTPOS in an offline PIN verification where the enciphered PIN is sent to the card is specified in [16]. Standards such as ISO-9564-1 & ISO-9564-1 and similar guidelines given in [7] make recommendations on how PINs and associated account information need to be protected during transmission from one system to another. Since there is no publicly available standard on how PIN block construction and encipherment should be carried out in OPV, we have made reasonable assumptions in our PIN block construction and encipherment.

In order to explain the three, proposed solutions more clearly, we first introduce a generic OPV model on which our proposals are based. The protocol diagram shown in figure 3 llustrates this.

In the generic protocol, the CTPOS sends the payment card a generated session key S_{K_T} and the cardholder entered PIN either in plaintext or in enciphered format. If the PIN is sent to the card in enciphered format, the CTPOS may use the card's public key recovered from the card's public key certificate, or the CTPOS may use a card owned PIN encipherment public key (if the card contains a dedicated key pair for PIN encipherment) as specified in EMV standard [16, see: p81]. However, it must be noted that if the PIN is sent

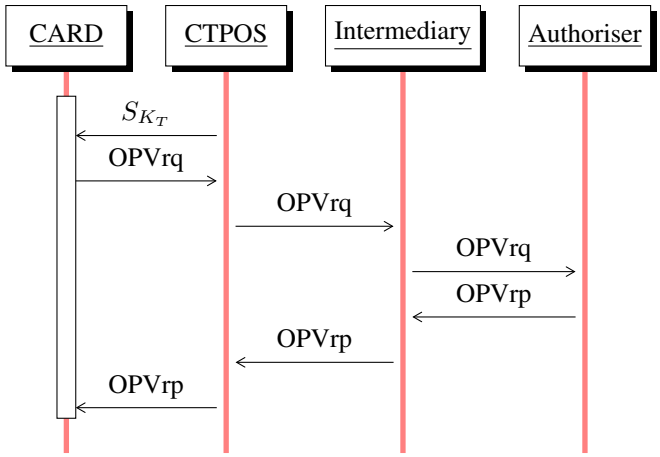

Fig. 3. Generic OPV Protocol Diagram

enciphered in a PIN block, the purpose of this message is for the card to retrieve the PIN, but not to respond to a VERIFY command carried out in offline PIN verification.

The payment card after receiving the message from CTPOS, constructs a PIN block which contains the PIN and details of the corresponding account number according to ISO-9564-1 & ISO-9564-1. The data included in the OPV PIN Block are shown in Table I. The Unpredictable Numbers (UN) mentioned in this study have the same properties as defined in the EMV specification [16]. The encipherment is done inside the card using a symmetric session key shared with the Authoriser, who may represent either the scheme operator or the CIB during OPV. We call this encipherment, OPV Enciphered PIN Block, that has the notation $e\{PB\}$.

For the encipherment of both the OVP PIN Block (PB) and the OPV PIN Result Block (PRB) using a symmetric encryption algorithm, we make our construction support three symmetric encipherment methods as detailed below;

1) **Basic Encryption:** Advanced Encryption Standard (AES) [17] as the symmetric encryption algorithm with Cipher Feedback Mode (CFB) as the mode of operation.
2) **Encrypt-then-MAC:** Advanced Encryption Standard (AES) as the symmetric encryption algorithm with Cipher Feedback Mode (CFB) as the mode of operation and a key based Message Authentication Code (MAC) computed using SHA256 to provide integrity.
3) **Authenticated Encryption:** Advanced Encryption Standard (AES) as the symmetric encryption algorithm and Galois/Counter Mode (GCM) [18, 19] as the mode of operation is used as a combined single operation to provide authenticated encryption.

Due to space restrictions, three separate messages detailing each of the symmetric encryption process are not included in this section but instead the symmetric encipherment of the OPV PIN Block which has the notation "$e\{PB\}$" represents any one of the three cryptographic processes detailed above depending on which one is selected. Depending on the symmetric encryption method used to create $e\{PB\}$, the CIB selects the same symmetric encryption method to encipher the OPV PIN Result Block (PRB) to create the enciphered OPV PIN Results Block "$e\{PRB\}$".

In our practical implementation as detailed in section V-B, however, we implement and measure the performance penalties of both symmetric and asymmetric encryption methods proposed in each of our solution. The findings are outlined in Tables VII.

In our generic protocol outlined here, for the symmetric encipherment of the OPV PIN Block, we use basic encryption mode with three blocks of 16 bytes each with a total length (L) of 48 bytes. In our practical implementation for this, we used 128bit - Advanced Encryption Standard (AES) [17] with Cipher Feedback Mode (CFB) as the mode of operation.

TABLE I. OPV PIN BLOCK (PB)

Data Header	:	1 Byte.
PIN Block	:	8 Bytes.
Card Unpredictable Number (CUN)	:	16 Bytes.
CTPOS Session Key (S_{K_T})	:	16 Bytes.
Random Padding ($L = 48 = length\ of\ encipherment$)	:	$L - 41$.

The payment card then constructs an Online PIN Verification Request (OPVrq) message addressed to the Authoriser which includes a concatenation of the personal account number (PAN) of the card (this could be a unique ID of that individual payment card), and the enciphered OPV PIN block $e\{PB\}$. The $e\{PB\}$ is enciphered using a session key shared between the smart card and the CIB.

$$OPVrq = PAN||e\{PB\}$$

The constructed OPVrq is then sent to the CTPOS, where the CTPOS device in this instance follows the existing EMV process and encrypts the PIN-related data it receives, using the key it shares with the next entity on the communication path, which may be the acquirer or an intermediary in the current EMV architecture. The OPVrq is forwarded in this way until it reaches the Authoriser.

The Authoriser uses the PAN to retrieve the shared session key and then decipher $e\{PB\}$. The PIN is then validated and the outcome is included in an OPV PIN Result Block (PRB). The PRB includes data as shown in Table II.

TABLE II. OPV PIN RESULT BLOCK (PRB)

Data Header	:	1 Byte.
Cardholder Verification Result(CVR)	:	5 Bytes.
Authoriser Unpredictable Number	:	16 Bytes.
Card Unpredictable Number	:	16 Bytes.

The Authoriser then constructs the Online PIN Verification Response (OPVrp) which is an encipherment, in this instance, of the PRB using the session key S_{K_T} of the CTPOS. The notation below is used to show the symmetric encryption of PRB using the key S_{K_T}.

$$OPVrp = e\{PRB\}$$

The constructed OPVrp is then sent to the same communication path, which may go through the same acquirer and intermediaries in the current EMV architecture until the OPVrp is reached by the CTPOS, which then deciphers the message to obtain the PIN verification results.

Once the CTPOS is satisfied with the OPVrp, it will transfer this message to the card. According to our threat model we have assumed that the payment card, CTPOS and scheme operator/CIB are trusted entities. EMV transaction authorisation continues following the OPV process described above.

A. Card Based Solutions

In our proposed card-based solution, the PB is enciphered inside the payment card before it is forwarded to the CIB (Authoriser in our generic model) to be verified. The card-based solutions are further sub-categorised, depending on whether a symmetric or an asymmetric cryptographic key is used to encipher the PB.

1) Card uses an online-PIN encipherment symmetric key of the CIB: The solution described here introduces an online-PIN encipherment symmetric key K_{OPV} that the payment card shares with the CIB. From the K_{OPV}, a session key $K_{S_{OPV}}$ is derived using a key derivation function similar to the one specified in EMV specification [16, see: p127 - p131] and also discussed in [4]. It is assumed that the session key derivation between the card and the CIB is synchronised.

When the card is inserted in the CTPOS, in this instance, the CTPOS sends the PIN that the cardholder enters to the payment card, either in plain text or in encrypted format. The CTPOS also sends the session key S_{K_T}. The OPV process in this solution follows exactly the same steps as described in the generic OPV model above. Considering the ownership of the payment card by the CIB, the necessary changes are under the control of the CIB.

2) Card uses an online-PIN encipherment public key of the CIB: The solution described here introduces an online-PIN encipherment public key $P_{OPV_{CIB}}$ of the CIB. It must be noted that this is not the CIB's public key that is recovered from the CIB's public key certificate residing in the card during EMV transactions. The $P_{OPV_{CIB}}$ in this context refers to a specific online-PIN encipherment public key introduced in our construction. This key is stored in the payment card by the CIB during card personalisation and is in the format of a public key certificate that has been signed by the CIB.

Similar to our generic model, the CTPOS first sends a session key S_{K_T} and the customer-entered PIN either in plain text or in enciphered format to the card. The card then constructs the PB-1. The data included in the PB-1 are shown in Table III. In this occasion, PB-1 is enciphered using the public key $P_{OPV_{CIB}}$ to generate the enciphered OVP PIN Block that has the notation $z\{PB\text{-}1\}$.

TABLE III. OPV PIN BLOCK -1 (PB-1)

$Data\ Header$	:	1 Byte.
$PIN\ Block$	:	8 Bytes.
$Card\ Unpredictable\ Number\ (CUN)$	:	16 Bytes.
$CTPOS\ Session\ Key\ (S_{K_T})$	:	16 Bytes.
$Random\ Padding\ (N = length\ of\ P_{OPV_{CIB}}\ )$	:	$N - 41$.

The payment card then constructs the OPVrq message which includes the PAN and the public key encipherment of PB-1.

$$OPVrq = PAN||z\{PB-1\}$$

The OPVrq is sent to the CTPOS, where the CTPOS device now follows the existing EMV process and encrypts the PIN-related data with the key it shares with the next entity on the communication path. The OPVrq is forwarded from entity to entity until it reaches the CIB. CIB then decrypts the PIN-related data and verifies whether the PIN entered by the cardholder is correct. The outcome of this verification is constructed in an OPV PIN Result Block (PRB-1) shown in Table IV

TABLE IV. OPV PIN RESULT BLOCK -1 (PRB-1)

$Data\ Header$	:	1 Byte.
$Cardholder\ Verification\ Result(CVR)$	:	5 Bytes.
$CIB\ Unpredictable\ Number$	:	16 Bytes.
$Card\ Unpredictable\ Number$	:	16 Bytes.

The CIB then constructs the OPVrp, which is the encipherment of the PRB-1 using the session key S_{K_T} of CTPOS.

$$OPVrp = e\{PRB-1\}$$

This message is then sent back using the same communication path until the CTPOS is reached. The CTPOS then deciphers $e\{PRB\text{-}1\}$ and verifies the CVR. Once satisfied, this is passed to the payment card. EMV transaction authorisation continues following the OPV process we described above.

B. Terminal-Based Solutions

In this solution, the PIN block is enciphered at the CTPOS before it gets sent forward to the CIB to be verified. In contrast to the previous two solutions, the cardholder-entered PIN is not sent to the card but instead is enciphered in the CTPOS.

1) Terminal uses an online-PIN encipherment public key of the CIB: An online-PIN encipherment public key $P_{OPV_{CIB}}$ of the CIB is introduced to carry out the OPV PIN Block encipherment at the CTPOS. The public key is stored in the payment card in a public key certificate, similar to the proposal in section IV-A2, and during a transaction the certificate is given to the CTPOS. During a transaction, the card provides; the CIB's public key certificate signed by a Certification Authority (CA) and the CIB's online-PIN encipherment public key certificate signed by the CIB. The CTPOS uses the CA's public verification key to verify that the CIB's public key was signed by the CA and then uses the CIB's public key to verify that $P_{OPV_{CIB}}$ was signed by the CIB. The CTPOS can then validate the public key $P_{OPV_{CIB}}$ recovered from the certificate and use this to create the public key encipherment of PB-2 that has the notation $z\{PB\text{-}2\}$. The OPV PIN Block PB-2 includes data mentioned in Table V.

TABLE V. OPV PIN BLOCK -2 (PB-2)

$Data\ Header$	:	1 Byte.
$PIN\ Block$	:	8 Bytes.
$CTPOS\ Unpredictable\ Number$	:	16 Bytes.
$CTPOS\ Session\ Key\ (S_{K_T})$	:	16 Bytes.
$Random\ Padding\ (N = length\ of\ P_{OPV_{CIB}}\ )$	:	$N - 41$.

The CTPOS then constructs the OPVrq message which includes the PAN and the public key encipherment of the OPV PIN Block as shown below.

$$OPVrq = PAN||z\{PB-2\}$$

The OPVrq is then sent to the next entity in the path of communication to the CIB. Once received, the CIB decrypts $z\{PB\text{-}2\}$ and verifies whether the PIN entered by the user is correct or not. The verification is included in a CVR. The CIB then constructs a OPV PIN Result Block (PRB-2) shown in Table VI

TABLE VI. OPV PIN RESULT BLOCK -2 (PRB-2)

$Data\ Header$	:	1 Byte.
$Cardholder\ Verification\ Result(CVR)$	:	5 Bytes.
$CIB\ Unpredictable\ Number$	:	16 Bytes.
$CTPOS\ Unpredictable\ Number$	:	16 Bytes.

The CIB then constructs the OPVrp which is the encipherment of the PIN result block using the session key S_{K_T} of CTPOS.

$$OPVrp = e\{PRB-2\}$$

The OPVrp is then sent back along the same communication path until it reaches the CTPOS, which then deciphers the encrypted PIN result block and retrieves the CVR. Once satisfied, the CTPOS will generate other commands if the transaction proceeds to transaction authorisation.

C. Binding of OPV and Transaction Authorisation

In the current EMV process there seems to be no direct linkage between the online PIN verification and the online transaction authorisation for a given EMV transaction. The two verifications are carried out separately, leaving space for replay attacks in which a harvested OPV Response Message could be replayed or injected by the compromised intermediary during an EMV transaction.

Both of our card-based and terminal-based proposals discussed in sections IV-A, IV-B help to eliminate the aforementioned attacks by making a minor change to the current EMV transaction authorisation message. In an online transaction authorisation process the payment card generates an Authorisation Request Cryptogram (ARQC) in response to the GENERATE AC command issued by the CTPOS [11, 16]. Here the payment card can include the *Authoriser Unpredictable Number* that is sent back to the CTPOS in the OPV PIN Result Block (PRB) as discussed in our generic OPV model in Section IV and shown in Table II, inside the ARQC, which is a symmetric encipherment using the shared key between the card and the Authoriser.

Once the Authoriser receives the ARQC and deciphers it, the Authoriser can use the *Authoriser Unpredictable Number* that it keeps a record of to link the previously verified OPV to the received transaction authorisation request. This gives assurance to the Authoriser that this is a genuine and timely transaction. The Authoriser may also include a combined verification result inside the Authorisation Response Cryptogram (ARPC) when the ARPC is sent back to the card.

V. ANALYSIS

In this section, we take the proposed solutions presented in the previous section and evaluate them for their security and performance. The security of the proposed solutions is analytically evaluated in relation to the attackers capability and performance measurements are taken to show the potential penalties for the existing process if they are adopted.

A. Analytical Analysis

Taking into account the adversary capabilities discussed in section III-B, a malicious user (adversary) can compromise an intermediary entity. The adversary cannot compromise the smart cards, payment terminals, scheme operator, or CIB. This limitation is imposed because if an adversary can successfully compromise any of these entities then almost no protection mechanism would be strong enough to protect against attacks on the OPV process. The rationale behind this is described below:

1) The smart card stores a copy of the authorised PIN, which it uses to compare with the value entered by the cardholder during an offline PIN-based transaction. If an adversary can break the tamper-resistant smart card to access the PIN, then he could potentially access any other information on it, thus rendering any countermeasure, including ours, redundant. Fundamentally, he can create a clone of the genuine smart card and perform transactions with the correct PIN.
2) If the adversary can successfully compromise a terminal, then he can access the authorised PIN whenever a genuine cardholder uses it. However, in this attack the adversary will only capture the PINs of the cards used on a single payment terminal.
3) An authorisation entity (e.g. scheme operator and CIB), similar to the payment cards, has to store copies of authorised PINs for authentication purposes. If an adversary compromises the authorisation entity then it is challenging to protect the PINs and the OPV process (even with our proposal).

In our proposed solutions, the smart card issued by the respective CIB either encrypts itself or gives the encryption (public) key to the payment terminal to be used for secure communication (OPVrq). The CIB, on receiving this message, decrypts the OPVrq and verifies the PIN. The response (OPVrp) of whether to indicate a successful PIN verification or not is sent back to the payment terminal. The payment terminal generates the session key that encrypts the OPVrp, which is included in the OPVrq by either the smart card or the terminal. The response message also includes a random number generated by the smart card or payment terminal (depending upon which proposed solution is selected), providing assurance of the freshness of the OPVrp message.

An adversary observing these messages can store them for the purpose of replaying them at some later stage. The adversary cannot see the PIN in plaintext as he does not have the capability of breaking a strong cryptographic algorithm. For the solution based on the symmetric key and encryption performed by the smart card, a replay of the OPVrq will be easily detected as the session key used for encrypting (and successful decryption) this message would have expired.

TABLE VII. PERFORMANCE PENALTIES ASSOCIATED WITH EACH SOLUTION

Proposed Solutions	Basic Encryption		Encrypt-then-MAC		Authenticated Encryption	
	Java Card 1	Java Card 2	Java Card 1	Java Card 2	Java Card 1	Java Card 2
Symmetric Key Card	64ms	86ms	138ms	156ms	122ms	136ms
Asymmetric Key Card	138ms	159ms	196ms	218ms	-	-
Asymmetric Key Payment Terminal[8]	34ms		48ms		-	
CIB OPVrp (footnote 7)	16ms		28ms		22ms	

However, for solutions based on an asymmetric cryptosystem (e.g. using public key of the CIB) the smart card or the payment terminal generates a session (symmetric) key that is then used by the CIB to encrypt the OPVrp. If the adversary replays the OPVrq message, generated using the public key of the CIB, the session key part of the replayed message would be different. As a result, the payment terminal may not be able to decrypt the OPVrp message properly, avoiding a successful replay of the OPVrq message.

To provide binding between the OPV process and the online transaction authorisation, we have proposed, including a random number generated by the authorisation entity in the online transaction authorisation. This provides a countermeasure against an adversary taking advantage of the lack of binding between the OPV process and the online transaction authorisation.

Consider a potential scenario in which an adversary creates a "Yes" card [1]. In this scenario, the OPV process has to execute the first two proposed solutions (based on the smart card). The payment terminal communicates the PIN entered by the malicious user to the "Yes" card and it generates an OPVrq message. The intention is that when the authorisation entity tries to verify the OPVrq and fails, it will send a PIN verification decline result back in the OPVrp. As this response goes back to the "Yes" card, it simply discards this message and tells the payment terminal that the PIN verification was successful. However, this scenario requires that the OPVrq is completely isolated from the payment terminal. In our proposal the symmetric key used to encipher the OPVrp message is generated by the payment terminal (and communicated to the authorisation entity in OPVrq). Therefore, the payment terminal can also decrypt the message and verify whether the PIN was verified or not. Hence, this potential scenario might not be able to circumvent the protection provided by our proposals.

Another concern that could be raised would be the use of a standard Initialisation Vector (IV) [20] for the symmetric cryptosystem. We are discussing this as there is the potential that patterns in the ciphertext might reveal some information regarding the PIN. To avoid this (even when the IV is a predefined value), in the OPVrq message, we append a random number generated by the smart card to the data header and then append the PIN value. This way, the first 16 byte (plaintext) block to be used by the symmetric algorithm (i.e. AES [17]) will have 15 random bytes and the first byte of the second block will also be random (the 16 byte random number is spread over the first two plaintext blocks). This randomness in the plaintext of the first block avoids any patterns being detected in the second block, which contains the PIN. Furthermore, the session keys are unique for the symmetric key-based proposals, making it difficult for an adversary to gain any additional information from the OPVrq or OPVrp message about the PIN or the associated decision.

B. Practical Implementation

In this section, we describe our implementation of the proposed solutions with the aim of providing potential performance penalties the existing OPV process has to bear. For our test bed, we selected two 32bit Java Cards [21] connected with a Microsoft Windows 7 machine running on 2.3GHz, 2GB RAM as a CTPOS and CIB (for OPVrp). For the symmetric key based solution, we selected Advanced Encryption Standard (AES) [17], 128bit key. In basic encryption and Encrypt-then-MAC we use the Cipher Feedback Mode (CFB). Whereas, for authenticated encryption we have opted for Galois/Counter Mode (GCM) [18, 19]. For the generation of the random number [22] we selected the HMAC-based Pseudorandom Number Generator (PRNG)[9] .

For the implementation of the public key-based solutions, we selected 1048bit RSA (Rivest, Shamir and Adleman) [23] with random padding generated by the selected PRNG. The session keys used in this solution were also generated using the same selected PRNG. The performance penalty is illustrated in the table VII and each measurement is represented in milliseconds (ms). A point to note is that there is no authenticated encryption mode (similar to GCM) for asymmetric cryptosystems (i.e. RSA). Therefore, there is not performance measurement for it included in the table VII.

Our implementation does not emulate the complete EVM specifications. It only implements the proposed modifications to the OPVrq and OPVrp. The performance measurement should be taken as an additional execution cost that the EVM process has to bear to implement the proposed solutions in this paper.

C. Mechanical Formal Analysis

In this section, we subject the proposed modifications to a mechanical formal analysis based on the CasperFDR tool.

The CasperFDR approach was adopted to test the soundness of the proposal under the defined security properties. In this approach, the Casper compiler [24] takes a high-level description of the protocol, together with its security requirements. It then translates the description into the process algebra of Communicating Sequential Processes (CSP) [25]. The CSP description of the protocol can be machine-verified using the Failures-Divergence Refinement (FDR) model checker [26]. The intruder's capability modelled in the Casper script (appendices A and B) for the proposed protocol is:

1) Intruder can masquerade any entity in the network
2) Intruders can read the messages transmitted in the network
3) Intruder cannot influence the internal process of an entity in the network

[9]PRNGs based on different cryptographic algorithms can give different performances; a detailed discussion of this can be found in [22]

The security specification for which the CasperFDR evaluates the network is as shown below. The listed specifications are defined in the #Specification section of appendices A and B:

1) The protocol run is fresh and both applications were alive,
2) The key used for encryption/decryption in the symmetric system and the private key used for decryption in the asymmetric system, is not revealed to the adversary,
3) Long terms keys of communicating entities are not compromised, and

The CasperFDR tool evaluated the protocol and did not find any feasible attack(s).

VI. Conclusion & Future Work

In this paper, we briefly described an OPV process that is based on publicly available information. The architecture of the payment network for the OPV and subsequently for the online transaction authorisation was explained. This payment network and its associated deployment open up a potential route for an adversary to compromise it to his benefit. We detailed our assumptions regarding the payment network's operating environment, the capabilities of an adversary and potential attack scenarios. Subsequently, we proposed three potential ways to enhance the OPV process and a proposal of how to bind it to the online transaction authorisation. Proposed solutions were then analysed with a discussion on their security in the context of the adversary's capabilities. We also provided the execution measurements for our proposed modifications; this showed the potential performance penalty incurred by our proposals. Furthermore, proposed modifications were then subjected to the mechnical formal anlysis using the CasperFDR tool. The concerns raised by this paper are considered to be valid as the OPV and online transaction authorisation is considered the highest level of trust in the card-based payment mechanism. It can differ based on laws/regulations or the relationship between the cardholder and CIB, but if the correct PIN is used in an OPV and online transaction authorisation then the liability of the payment is either with the cardholder or the CIB. If attacks can successfully occur at this level they could potentially cause substantial reputation damage to the overall card-based payment scheme, along with causing financial loss to the cardholder/CIB. Furthermore, such attacks could make it difficult to detect whether an OPV-based transaction was actually made by the cardholder or the adversary, as the compromise of the intermediary nodes might not be detected in time. Therefore, we consider this to be a concern and suggest that a mandating rollout of an OPV process in a geographical region should take into consideration these concerns and our potential solutions.

As part of our future research directions, we aim to explore the payment architecture for contactless payment systems and evaluate it with respect to issues similar to those presented in this study. The significance of PIN capture for contactless cards would be higher as the potential for successful transactions using relay attacks will increase. We will also evaluate key sharing schemes between payment terminal providers and their payment terminals (at merchants' sites).

References

[1] S. J. Murdoch, S. Drimer, R. Anderson, and M. Bond, "Chip and pin is broken," in *Security and Privacy (SP), 2010 IEEE Symposium on*. IEEE, 2010, pp. 433–446.

[2] R. J. Sullivan, "Can smart cards reduce payments fraud and identity theft?" *Federal Reserve Bank of Kansas City, Economic Review*, vol. 93, no. 3, pp. 35–62, 2008.

[3] W. Rankl and W. Effing, *Smart Card Handbook*, 3rd ed. New York, NY, USA: John Wiley & Sons, Inc., 2003.

[4] K. Mayes and K. Markantonakis, Eds., *Smart Cards, Tokens, Security and Applications*. Springer, 2008.

[5] D. King, "Chip-and-pin: Success and challenges in reducing fraud," in *Retail Payments Risk Forum, January*, 2012.

[6] *EMV Integrated Circuit Card Specifications for Payment Systems, Book 4: Cardholder, Attendant, and Acquirer Interface Requirements, Version 4.3, EMVCo, LLC*, Std. Version 4.3, November 2011.

[7] *Issuer PIN Security Guidelines*, Visa Public , November 2010.

[8] O. Berkman and O. Ostrovsky, "The unbearable lightness of pin cracking," in *Financial Cryptography and Data Security*, ser. Lecture Notes in Computer Science, S. Dietrich and R. Dhamija, Eds. Springer Berlin Heidelberg, 2007, vol. 4886, pp. 224–238. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-77366-5_20

[9] M. Mannan and P. C. van Oorschot, "Weighing down "the unbearable lightness of pin cracking"," in *Financial Cryptography and Data Security*. Springer, 2008, pp. 176–181.

[10] M. Centenaro, R. Focardi, F. L. Luccio, and G. Steel, "Type-based analysis of PIN processing APIs," in *Computer Security–ESORICS 2009*. Springer, 2009, pp. 53–68.

[11] *EMV Integrated Circuit Card Specifications for Payment Systems, Book 3: Application Specification, Version 4.3, EMVCo, LLC*, Std. Version 4.3, November 2011.

[12] BBC-News, "US and UK accused of hacking Sim card firm to steal codes," 20th February 2015. [Online]. Available: http://www.bbc.co.uk/news/technology-31545050

[13] Kaspersky-Lab, "Equation group: The crown creator of cyber-espionage," 16th February 2015. [Online]. Available: http://www.kaspersky.com/about/news/virus/2015/equation-group-the-crown-creator-of-cyber-espionage

[14] Kaspersky-Lab., "The great bank robbery: Carbanak cybergang steals $1bn from 100 financial institutions worldwide," 16th February 2015. [Online]. Available: http://www.kaspersky.com/about/news/virus/2015/Carbanak-cybergang-steals-1-bn-USD-from-100-financial-institutions-worldwide

[15] *EMV Integrated Circuit Card Specifications for Payment Systems, Book 1: Application Independent ICC to Terminal Interface Requirements, Version 4.3, EMVCo, LLC*, Std., November 2011.

[16] *EMV Integrated Circuit Card Specifications for Payment Systems, Book 2: Security and Key Management, Version 4.3, EMVCo, LLC*, Std., November 2011.

[17] Joan Daemen and Vincent Rijmen, *The Design of Rijndael: AES - The Advanced Encryption Standard*. Berlin, Heidelberg, New York: Springer Verlag, 2002.

[18] D. McGrew and J. Viega, "The galois/counter mode of operation (gcm)," *Submission to NIST. http://csrc. nist. gov/CryptoToolkit/modes/proposedmodes/gcm/gcm-spec. pdf*, 2004.

[19] M. Dworkin, *Recommendation for block cipher modes of operation: Galois/Counter Mode (GCM) and GMAC*. US Department of Commerce, National Institute of Standards and Technology, 2007.

[20] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone, *Handbook of Applied Cryptography*. CRC, October 1996.

[21] *Java Card Platform Specification: Classic Edition; Application Programming Interface, Runtime Environment Specification, Virtual Machine Specification, Connected Edition; Runtime Environment Specification, Java Servlet Specification, Application Programming Interface, Virtual Machine Specification, Sample Structure of Application Modules*, Oracle Std. Version 3.0.1, May 2009. [Online]. Available: http://java.sun.com/javacard/3.0.1/specs.jsp

[22] R. N. Akram, K. Markantonakis, and K. Mayes, "Pseudorandom Number Generation in Smart Cards: An Implementation, Performance

and Randomness Analysis," in *5th International Conference on New Technologies, Mobility and Security (NTMS)*, Antonio Mana and M. Klonowski, Eds. Istanbul, Turkey: IEEE Computer Society, May 2012.

[23] B. Schneier, *Applied cryptography (2nd ed.): protocols, algorithms, and source code in C*. New York, NY, USA: John Wiley & Sons, Inc., 1995.

[24] G. Lowe, "Casper: a compiler for the analysis of security protocols," *J. Comput. Secur.*, vol. 6, pp. 53–84, January 1998. [Online]. Available: http://dl.acm.org/citation.cfm?id=353677.353680

[25] C. A. R. Hoare, *Communicating sequential processes*. New York, NY, USA: ACM, 1978, vol. 21, no. 8.

[26] P. Ryan and S. Schneider, *The Modelling and Analysis of Security Protocols: the CSP Approach*. Addison-Wesley Professional, 2000.

APPENDIX A
CASPERFDR SCRIPT - SYMMETRIC SYSTEM

```
#Free variables
SC, CI: Agent
Dh, Pb, Rp, Cvr; Num
Cun, Aun: Nonce
InverseKeys = (EnMaKey, EnMaKey),
(SessionEnMaKey, SessionEnMaKey)

#Protocol description
0.    -> SC : SC [CI!=SC] <iMsg :={Dh, Pb,
Cun,SessionEnMaKey,Rp}{EnMaKey}>
1. SC -> CI : SC, iMsg
2. CI -> SC : {DH,Cvr, Aun,
Cun}{SessionEnMaKey}

#Actual variables
SCard, CIssuer, ME: Agent
DH, PB, RP, CVR: Num
CUN, AUN, NMalicious: Nonce

#Processes
INITIATOR(SC,CI, Cun) knows EnMaKey
RESPONDER(CI,SC, Aun) knows EnMaKey

#System
INITIATOR(SCard, SIssuer, CUN)
RESPONDER(SIssuer, SCard, AUN)

#Intruder Information
Intruder = ME
IntruderKnowledge = {SIssuer, SCard, ME,
GMalicious, NMalicious}

#Specification
Aliveness(SI, SC)
Aliveness(SC, SI)
Secret(SC, EnMaKey, [CI])
Secret(Sc, SessionEnMaKey, [CI])
```

APPENDIX B
CASPERFDR SCRIPT - ASYMMETRIC SYSTEM

```
#Free variables
SC, CI: Agent
Dh, Pb, Rp, Cvr; Num
Cun, Aun: Nonce
EKey: Agent->PublicKey
DKey: Agent->SecretKey
InverseKeys = (VKey, SKey), (SessionEnMaKey,
SessionEnMaKey)

#Protocol description
0.    -> SC : SC [CI!=SC] <iMsg := {Dh, Pb,
Cun, SessionEnMaKey, Rp}{EKey(CI)}>
1. SC -> CI : SC, iMsg
2. SC -> CI : {Dh, Cvr, Aun,
Cun}{SessionEnMaKey}

#Actual variables
SCard, CIssuer, ME: Agent
DH, PB, RP, CVR: Num
CUN, AUN, NMalicious: Nonce

#Processes
INITIATOR(SC,CI, Cun)knows EKey
RESPONDER(SC,SI, Aun) knows DKey(CI), EKey

#System
INITIATOR(CIssuer, SCard, CUN)
RESPONDER(SCard, CIssuer, AUN)

#Functions
symbolic EKey, DKey

#Intruder Information
Intruder = ME
IntruderKnowledge = {CIssuer, SCard, ME,
NMalicious, DKey(ME), EKey}

#Specification
Aliveness(CI, SC)
Aliveness(SC, CI)
Secret(SC, SessionEnMaKey, [CI])
```