Cryptanalysis of GlobalPlatform Secure Channel Protocols

Mohamed Sabt[1,2], Jacques Traoré[1]

[1] Orange Labs, 42 rue des coutures, 14066 Caen, France
{mohamed.sabt, jacques.traore}@orange.com
[2] Sorbonne universités, Université de technologie de Compiègne,
Heudiasyc, Centre de recherche Royallieu, 60203 Compiègne, France

Abstract. GlobalPlatform (GP) card specifications are the de facto standards for the industry of smart cards. Being highly sensitive, GP specifications were defined regarding stringent security requirements. In this paper, we analyze the cryptographic core of these requirements; i.e. the family of Secure Channel Protocols (SCP). Our main results are twofold. First, we demonstrate a theoretical attack against SCP02, which is the most popular protocol in the SCP family. We discuss the scope of our attack by presenting an actual scenario in which a malicious entity can exploit it in order to recover encrypted messages. Second, we investigate the security of SCP03 that was introduced as an amendment in 2009. We find that it provably satisfies strong notions of security. Of particular interest, we prove that SCP03 withstands algorithm substitution attacks (ASAs) defined by Bellare et al. that may lead to secret mass surveillance. Our findings highlight the great value of the paradigm of provable security for standards and certification, since unlike extensive evaluation, it formally guarantees the absence of security flaws.

Keywords: GlobalPlatform, secure channel protocol, provable security, plaintext recovery, stateful encryption

1 Introduction

Nowadays, smart cards are already playing an important role in the area of information technology. Considered to be tamper resistant, they are increasingly used to provide security services [38]. Smart cards do not only owe their tamper resistance for their success; programmability is a key issue for the wide adoption of this technology. Indeed, programmability made it possible to load new applications or remotely personalize existing ones during the cards life cycle [33]. However, this dynamicity did not come without price, as it has brought up security concerns about the novel system of content management. The absence of standards has motivated the creation of GlobalPlatform.

GlobalPlatform (GP) [21] is a cross-industry consortium that publishes specifications on how post-issuance management shall be carried out for smart cards. This includes the functionality to remotely manage cards content in a secure way.

The cryptographic heart of these mechanisms is the family of *Secure Channel Protocols* (SCPs) which protect the exchanged messages. Optimized for cards, the used encryption schemes in these protocols do not follow any standardized or provably secure construction.

Since its first publication, the GP card specifications have been the subject of diverse verifications. For instance, authors in [3] examine some aspects of these specifications and prove their soundness via the B method. Nevertheless, to the best of our knowledge, no rigorous analysis of the SCP encryption schemes has been provided before. Our goal is thus to study them through provable security, and hence to validate (or invalidate) the security guaranteed by GP.

1.1 Our Contribution

In this paper, we apply the methods of provable security on GP specifications. We start by analyzing the most popular GP SCP (i.e. SCP02). Much to our surprise, we find that SCP02 is vulnerable to a well-known security flaw caused by encrypting data using CBC mode with no random initialization vector (IV).

We illustrate this theoretical flaw by presenting a plaintext recovery attack where the adversary succeeds in getting some information about encrypted messages. To this end, we define an attack scenario in which several entities (e.g. service providers) communicate with the smart card via a trusted third party. Our attack allows a malicious entity to recover some encrypted messages belonging to another entity. In particular, messages including data with limited values and thus of low entropy, such as PINs, are the most exposed.

Then, we shift our analysis to the youngest member of the SCP family (i.e. SCP03). SCP03 encrypts messages using the "Encrypt-then-MAC" (EtM) method that is proved secure in [8]. In this paper, we provide a stronger result: SCP03 satisfies the security model defined by Bellare et al. in [6] which better models the particularity of SCP03. Indeed, SCP03 maintains a counter (i.e. state) for its decryption. One main advantage of this model is that, in addition to satisfying the existing notions of confidentiality [4] and integrity [7], it protects against replay and out-of-delivery attacks. More importantly, we prove that SCP03 defends against the recent threat of mass surveillance by algorithm-substitution attacks (ASAs) [9]. Typically closed-source, the industry of smart cards is concerned about ASAs, since no code scrutiny is possible to assert the absence of backdoors in the implemented protocols. This could damage the confidence in smart cards. Our proof guarantees that SCP03 cannot undetectably contain hidden backdoors allowing mass surveillance as it is outlined in [9].

Our work brings to light an interesting fact: security in well-established standards still does not withstand a simple cryptanalysis. We show, once again, that security by extensive verification or eminent authority is highly misleading. Indeed, being involved in sensitive services (e.g. payment), GP specifications have undergone rigorous verification and validation. This is why they are used in several systems achieving high assurance level in common criteria (CC) [36]. We emphasize that the presented vulnerability of SCP02 is well-known in the domain of cryptography [30]. Thus, our result raises serious concerns about CC certification. We encourage therefore further integration of *provable security* inside the enterprise of certification to improve the security of the certified protocols.

1.2 Related Work

Blockwise-Adaptive Attack. Formalized by Fouque et al. in [20], blockwise security has been firstly introduced in [29]. Its idea is simple: messages are not processed atomically in practice, so an adversary is able to get the ciphertext of a part of the message. Authors motivate this notion by attacking three encryption schemes proved to be secure against chosen plaintext attacks. We will focus solely on the case of CBC mode. The security proof of CBC in [4] holds only if all the calls to the underlying block cipher are independent from each other. This means that the $(i-1)$th block of ciphertext must not be known before choosing the ith block of plaintext, otherwise independence is lost and the proof fails. We note that using a predictable IV could be seen as a special case, since the first block of ciphertext, which is the IV, is known in advance before choosing the message.

Despite its popularity, Mitchell in [34] (and more recently Rogaway in [40]) concludes that the CBC mode involves so many security constraints that it would be better to abandon it for future designs. Indeed, a great number research effort has been dedicated to extend the weakness of CBC beyond theory. Some have adapted the vulnerability mentioned above in order to undermine the security of SSL3.0/TLS1.0 [2, 14]. Both attacks were motivated by the fact that the SSL3.0/TLS1.0 standard mandates the use of CBC encryption with chained (IVs); i.e. subsequent IVs are the last block of the previous ciphertexts. The attack of [14], called BEAST, is so efficient that migration to TLS1.1 has been recommended by IETF. Independently from blockwise security, authors in [6], inspired from [12], outline a vulnerability in the secure shell protocol (SSH) caused by the same reason: CBC encryption with chained IVs. It is worth noting that all the attacks of [2, 6, 12, 14] follow the same principle: some kind of plaintext recovery is possible when the attacker can both predict the next IV and control the first block of the message.

The presented attack against SCP02 follows a similar principle because constant IVs are always predictable. Despite similarity, we argue that the vulnerability presented in this paper is caused by the design of SCP02 that is quite delicate to secure. Indeed, it instructs the use of CBC encryption together with CBC-MAC for computing integrity tag. Such combination creates two contradictory requirements for security. Indeed, it is proved that a random IV is necessary for CBC encryption [4], while CBC-MAC must use constant IV [5].

Authenticated Encryption. Authenticated encryption (AE) is a symmetric encryption scheme that protects both data confidentiality and integrity (authenticity). The security notions of AE were formalized in the early 2000s in [7, 31]. Generic composition [8] is the most popular approach for numerous security protocols, such as SSH, TLS and IPsec. This approach is about combining confidentiality-providing encryption together with a message authentication code (MAC). Generally, three composition methods are considered: Encrypt-and-MAC, MAC-then-Encrypt and Encrypt-then-MAC.

The family of SCP protocols follows the paradigm of generic composition. On the one hand, SCP02 relies on the "Encrypt-and-MAC" (EaM) method. As pointed out in [8], this composition method is not generically secure, but we do not consider this result in our analysis for two reasons. First, chosen ciphertext attacks are not included in our threat model, since they are hardly applicable. Indeed, the decryption operation in SCP02 is only performed by smart cards that are unlikely to misbehave (due to their tamper-resistance and their controlled content management). Smart cards keep the decrypted messages and never output them outside, otherwise of course encryption would be of no use. Thus, attackers cannot obtain the result of the decryption operation. Second, the used cryptographic schemes for both encryption and authentication are not secure, hence simpler attacks exist. We provide further details in Section 3.

On the other hand, SCP03 utilizes the "Encrypt-then-MAC" (EtM) method which is proved to satisfy standard security notions: confidentiality (IND-CPA) and integrity of messages. We note that EtM is ill-suited to formalize all the power of SCP03. In this paper, we prove that SCP03 protects against a wider range of attacks, thanks to its stateful decryption. Of particular value, we prove that it withstands replay attacks and that any secret subversion of SCP03 for malicious goals can be detected. The latter is an important feature, since the absence of source code might cast doubts on the trustworthiness of smart cards.

1.3 Paper Outline

The rest of the paper is structured as follows: Section 2 gives background information on GlobalPlatform and reviews some classical definitions. In Section 3, we introduce the attack against SCP02 and demonstrate how it could be exploited in practice. Section 4 and 5 present our provable security results of SCP03. We end this paper by providing some discussion and specific recommendations related to the identified vulnerability.

2 Preliminaries

2.1 GlobalPlatform

GP card specification [23] refers to a number of technical standards that aim to develop flexible framework for smart cards. Within the GP architecture, the security domain (SD) controls applications on smart cards by supporting various cryptographic functions. For the purpose of this paper, we will be uniquely interested in secure communication, and we will be using the notion of SD as the component containing its private key that it uses to establish secure sessions.

Secure Channel Protocol. The GP card specification defines secure channel protocols (SCPs) to provide secure communication. Mainly designed for content management, they are also used by applications for their sensitive operations.

Whenever a secure session is needed, the SCP executes three steps: (1) *initialization* that includes entities authentication and derivation of session keys; (2) *operation* in which exchanged data are protected; and (3) *termination* ending the session. Our target in this paper is the encryption schemes employed during the second step. In follows, we provide more details about the *operation* of SCP02 and SCP03. SCP01 is not discussed because of its deprecation. It is worth mentioning that all given details on SCPs come from the GP card specification version 2.3, which is the latest version at the time of writing this paper.

2.2 Definitions

Notation. A message is a string. A string is an element of $\{0,1\}^*$. The concatenation of strings X and Y is denoted $X||Y$ or simply XY. For a string X, its length is represented by $|X|$. For an integer $N \in \mathbb{N}$, N`++` denotes the C-like `++` operator that returns the value N and then increases its value by 1. A *block cipher* is a function $E : \mathtt{Key} \times \{0,1\}^n \longrightarrow \{0,1\}^n$, where `Key` is a finite nonempty set and $E_{\mathsf{k}}(.) = E(\mathsf{k}, .)$ is a permutation, hence invertible, on $\{0,1\}^n$. The number n is called the *block length.* A *tweakable block cipher* (TBC) [32] extends the notion of block ciphers. A TBC $\widetilde{E} : \mathtt{Key} \times \mathtt{Tweak} \times \{0,1\}^n \longrightarrow \{0,1\}^n$ is a family of permutations parameterized by a pair (K, T), where K is a secret key and T is a public tweak. We define five finite nonempty sets of strings: `Key`, `TWEAK`, `NONCE`, `MSG` and `CTXT`. Let K be a key, T be a tweak, N be a nonce, M be a message and C be a ciphertext. Henceforth, unless stated otherwise, for all K, T, N, M and C, we have $\mathsf{K} \in \mathtt{Key}$, $T \in \mathtt{TWEAK}$, $N \in \mathtt{NONCE}$, $M \in \mathtt{MSG}$ and $C \in \mathtt{CTXT}$. We use the notation $\mathbf{A}^{\mathcal{O}}$ to denote the fact that the algorithm $\mathbf{A}$ can make queries to the function $\mathcal{O}$. Hereafter, we say that the *adversary* $\mathbf{A}$ has access to the *oracle* $\mathcal{O}$. If f is a probabilistic (resp., deterministic) algorithm, then $y \overset{R}{\leftarrow} f(x)$ (resp., $y \leftarrow f(x)$) denotes the process of running f on input x and assigning the result to y. The notation $A \Rightarrow x$ means that the adversary $\mathbf{A}$ outputs the value x.

Symmetric Encryption Schemes. A *symmetric encryption scheme* $\mathcal{SE}$ is defined by three algorithms $(\mathcal{K}, \mathcal{E}, \mathcal{D})$, where (1) the *key generation algorithm*, $\mathcal{K}$, takes a security parameter $k \in \mathbb{N}$ and returns a key K. We write $\mathsf{K} \overset{R}{\longleftarrow} \mathcal{K}(k)$; (2) the *encryption algorithm*, $\mathcal{E}$, takes a key K and a plaintext M to produce a ciphertext C. We write $C \longleftarrow \mathcal{E}_{\mathsf{k}}(M)$; and (3) the *decryption algorithm*, $\mathcal{D}$, takes a key K and a ciphertext C to return either the corresponding plaintext M or a special symbol $\perp$ to indicate that the ciphertext is invalid. We require that $\mathcal{D}_{\mathsf{k}}(\mathcal{E}_{\mathsf{k}}(M)) = M$ for all M and K.

The Cipher Block Chaining (CBC) Mode. Both SCP02 and SCP03 use symmetric encryption with the CBC mode. In CBC, each block of the plaintext is XORed with the previous ciphertext block before being encrypted. The first block of the plaintext is XORed with an initial value (IV). Here, we only consider the variant where the IV is explicitly given as input. We write $C \longleftarrow \mathcal{E}_{\mathsf{k}}\text{-CBC}(iv, M)$ and $M \longleftarrow \mathcal{D}_{\mathsf{k}}\text{-CBC}(iv, C)$.

Nonce-based Symmetric Encryption. As defined by Rogaway in [39], a *nonce-based encryption* n$\mathcal{SE} = (n\mathcal{K}, n\mathcal{E}, n\mathcal{D})$ is a symmetric scheme where both the encryption and the decryption algorithms are deterministic and stateless. They take an extra input called the *nonce* N, which is a variable that takes a new value with every encryption. We write $C \longleftarrow n\mathcal{E}_{\mathsf{k}}(N, M)$ and $M \longleftarrow n\mathcal{D}_{\mathsf{k}}(N, C)$.

Message Authentication Schemes. Conventionally, a *message authentication scheme* (MAC) $\mathcal{MA} = (\mathcal{K}, \mathcal{T}, \mathcal{V})$ consists of three algorithms. $\mathcal{K}$ is the probabilistic algorithm for key generation. The tagging algorithm, $\mathcal{T}_{\mathsf{k}}$, takes a key K and a message M to return a tag τ. The verification algorithm, $\mathcal{V}_{\mathsf{k}}$, takes a key K, a message M and a candidate tag τ' to return a bit. We require that $\mathcal{V}_{\mathsf{k}}(M, \mathcal{T}_{\mathsf{k}}(M)) = 1$ for all M and K.

Tweakable Functions. A *tweakable* function $\widetilde{F}(.,.)$ is a function where a 'tweak' is required for its computation. We write $y \longleftarrow \widetilde{F}(T, M)$.

Standard Security Notions. We associate to any adversary a number called its "advantage" that measures her success in breaking a given scheme. A scheme is said *secure* with respect to a given security notion if all related polynomial-time adversaries have a negligible advantage. We write $\mathbf{Adv}^{\mathcal{SN}}_{\mathcal{SC}}(A)$, where $\mathbf{A}$ is an adversary attacking the scheme $\mathcal{SC}$ regarding the security notion $\mathcal{SN}$.

Definition 1. *(Indistinguishability of a Symmetric Encryption Scheme (IND)).* Given a symmetric encryption $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ and a ciphertext of one of two plaintexts, no adversary can distinguish which one was encrypted. IND can be expressed as an experiment [4]. Let $\mathcal{E}_{\mathsf{k}}(\mathcal{LR}(.,.,b))$ be a *left-or-right* oracle where $b \in \{0, 1\}$: the oracle takes two messages as input, m_0 and m_1, where $|m_0| = |m_1|$, and returns $C \leftarrow \mathcal{E}_{\mathsf{k}}(m_b)$. The adversary submits queries of the form (m_0, m_1) to the oracle, and must guess the bit b, i.e. which message was encrypted. This security notion is often called *IND-CPA*, where *CPA* represents chosen-plaintext attacks. For an adversary $\mathbf{A}^{\mathcal{E}}$, the advantage is defined as $\mathbf{Adv}^{\textit{ind-cpa}}_{\mathcal{SE}}(A) = \big|\mathsf{Pr}[A^{\mathcal{E}} \Rightarrow 1 \,|\, b = 1] - \mathsf{Pr}[A^{\mathcal{E}} \Rightarrow 1 \,|\, b = 0]\big|$. There is a stronger security notion associated to IND that is called IND-CCA (CCA stands for chosen-ciphertext attacks). In the IND-CCA experiment, besides the encryption oracle, the adversary has access to a decryption oracle $\mathsf{D}_{\mathsf{k}}(.)$, so that she can choose any ciphertext and obtain its plaintext. There is one restriction for using $\mathsf{D}_{\mathsf{k}}(.)$: the adversary cannot ask to decrypt ciphertexts that were previously generated by the encryption oracle, otherwise a trivial attack is possible.

Definition 2. *(Strong Unforgeability (SUF-CMA)).* This notion was adapted by Bellare et al. [5] from the definition of security of digital signatures. Given a message authentication scheme $\mathcal{MA} = (\mathcal{K}, \mathcal{T}, \mathcal{V})$, we consider a game in which the adversary makes arbitrary queries to a tagging oracle $\mathcal{T}_{\mathsf{k}}$ as well as a verification oracle $\mathcal{V}_{\mathsf{k}}$. The adversary $\mathbf{A}^{\mathcal{T},\mathcal{V}}$ wins (outputs 1) if she can find a pair (M, τ) such that $\mathcal{V}_{\mathsf{k}}(M, \tau) = 1$, but τ was never returned by $\mathcal{T}_{\mathsf{k}}$ as tag of M. The advantage of $\mathbf{A}$ is defined as $\mathbf{Adv}^{\textit{suf-cma}}_{\mathcal{MA}}(A) = \mathsf{Pr}[A^{\mathcal{T},\mathcal{V}} \Rightarrow 1]$.

Definition 3. *(Integrity of Ciphertext (INT-CTXT)).* Defined in [8], this notion requires that no adversary be able to produce a valid ciphertext which the encryption oracle had never produced before. Given an encryption scheme $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$, we consider a game in which the adversary has access to an encryption oracle $\mathcal{E}_{\mathsf{k}}(.)$ and to a decryption one $\mathcal{D}_{\mathsf{k}}(.)$. The adversary $\mathbf{A}^{\mathcal{E},\mathcal{D}}$ wins (outputs 1) if she can find a ciphertext C, such that (1) it was not produced by $\mathcal{E}_{\mathsf{k}}(.)$ and (2) it does not decrypt to $\perp$. The advantage of $\mathbf{A}$ is defined as $\mathbf{Adv}^{\textit{int-ctxt}}_{\mathcal{SE}}(A) = \Pr[A^{\mathcal{E},\mathcal{D}} \Rightarrow 1]$.

Definition 4. *(Stateful Pseudorandom Function (sPRF)).* Let $F = \{F_{\mathsf{k}} : \mathsf{K} \in \mathcal{K}(k)\}$ where F_{k} is a deterministic stateful function mapping l-bit strings to l'-bit strings for each $\mathsf{K} \in \mathcal{K}(k)$. Let $R_{\mathcal{S}}$ be a stateful random-bit oracle. This means that the output of $\mathcal{R}_{\mathcal{S}}$ depends on its state. Indeed, given a message $M \in \{0,1\}^l$, $\mathcal{R}_{\mathcal{S}}(M)$ returns two different l'-bit random strings for two subsequent calls. The goal is that no adversary $\mathbf{A}$ can distinguish whether she is interacting with a random instance of F or with its oracle $R_{\mathcal{S}}$. $\mathbf{A}$'s advantage is $\mathbf{Adv}^{\textit{sprf}}_{F}(A) = \left|\Pr[A^{F} \Rightarrow 1] - \Pr[A^{\mathcal{R}_{\mathcal{S}}} \Rightarrow 1]\right|$.

Definition 5. *(Indistinguishability from Tweakable Random Bits under CPA ($\widetilde{\text{IND}}$-CPA)).* Here, we present a variant of the distinguishing concept defined for tweakable functions and presented in [10]. We define $\widetilde{\text{IND}}$-CPA as follows. Let $\widetilde{F}$ be a tweakable function mapping pairs of (t, l)-bit strings for each $\mathsf{K} \in \mathcal{K}(k)$. Let $\widetilde{\mathcal{R}}$ be a tweakable random-bit oracle from $\{0,1\}^t \times \{0,1\}^l$ to $\{0,1\}^{l'}$. The goal is that no adversary $\mathbf{A}$ can distinguish whether she is interacting with a random instance of $\widetilde{F}$ or with its oracle $\widetilde{\mathcal{R}}$. The advantage of $\mathbf{A}$ is defined as $\mathbf{Adv}^{\widetilde{\text{ind}}\text{-cpa}}_{\widetilde{F}}(A) = \left|\Pr[A^{\widetilde{F}} \Rightarrow 1] - \Pr[A^{\widetilde{\mathcal{R}}} \Rightarrow 1]\right|$.

3 Secure Channel Protocol '2'

3.1 Description

SCP02 is the recommended protocol in the GP specifications. It is built upon symmetric encryption based on block ciphers, hence the need of secret keys and padding data. Informally, it uses "Encrypt-and-MAC" construction, wherein the message is both encrypted and integrity protected (by using a MAC algorithm). The MAC value is appended to the encrypted message to produce the ciphertext.

In more detail, padding is first added to the message and a MAC tag is computed over the resulted data. Then, the payload is encrypted after stripping off the MAC padding to replace it by a payload one. Padding is done with binary zeroes started by $0x80$. Figure 1 schematically shows the ciphertext format.

Concerning the schemes in use, SCP02 mandates to encrypt data using *triple DES* in CBC mode [26] with no IV, namely IV of binary zeroes (refer to Section E.4.6 in [23]). As for MAC computing, it uses a chained version of ISO9797-1 MAC algorithm 3, which includes a CBC-MAC processing with a simple DES and a Triple DES computation for the last block of the message. As a security

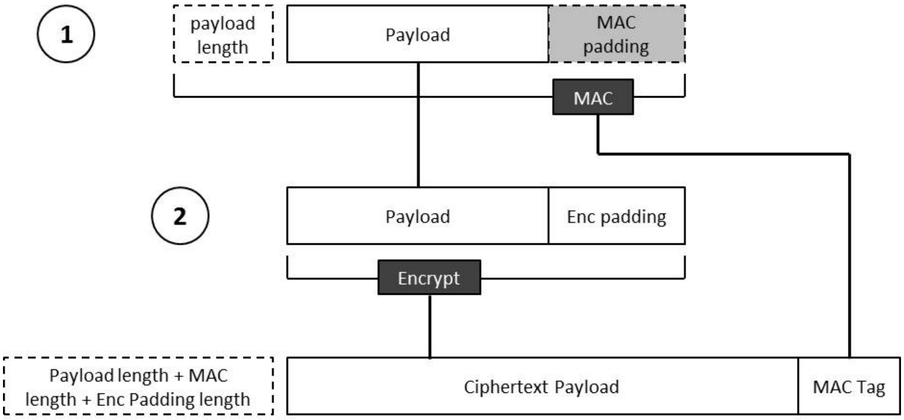

Fig. 1. Ciphertext Generation by SCP02. Grey boxes, i.e. 'MAC padding', are not included in the 'encrypt' operation.

enhancement, the last valid MAC tag is DES-encrypted before being applied to the calculation of the next MAC.

We note that both schemes are vulnerable. Indeed, authors in [19] perform a side-channel attack to defeat the ISO9797-1 MAC algorithm 3. Their attack allows one to recover the secret key used for the MAC computation. The consequences of such attack are limited for the reason that SCP02, like any SCP, generates the MAC tag using a temporary session key. Therefore, we do not consider this attack in the rest of the paper. In the sequel, we describe how an attacker might exploit the absence of random IVs to recover encrypted messages.

3.2 Try-and-Guess Attack

It is easy to see that for a fixed iv the CBC encryption $\mathcal{E}_{\mathsf{k}}$-CBC$(iv, .)$ is a stateless deterministic function of the key K. Indeed, it always yields the same ciphertext when encrypting the same message multiple times (using the same key). This has both theoretical and practical consequences.

Theoretically, it violates the security goal IND-CPA. An adversary can tell which message was encrypted after only two queries: the first query contains the same plaintext M twice, while the second one includes M together with another plaintext. The adversary succeeds with a probability 1, since if M was encrypted, the encryption oracle would output the same result as for the first query. We note that the adversary succeeds due to the fact that the IND-CPA experiment does not constrain the adversary from submitting queries of the form (M, M) to the encryption oracle $\mathcal{E}_{\mathsf{k}}(\mathcal{LR}(., ., b))$.

In practice, an eavesdropper observing the stream of ciphertexts is able to determine whether two ciphertexts come from the same message. Better yet, the eavesdropper can detect whether two messages share the same prefix. This could be useful to study the structure of the encrypted stream by recognizing the presence of the same data multiple times. Now, we turn the above scenario into a more serious attack.

Consider an adversary **A** who can mount a chosen-plaintext attack. **A** starts by observing a ciphertext C ($= \mathcal{E}_\mathsf{k}$-SCP02(M)). Recall that the goal of **A** is to find M. **A** achieves her goal by repeatedly trying all possible values for M until the correct one is identified. For instance, if the adversary knows that M is one of N possible values, then she can determine the actual value of M after $N/2$ (on average) guesses. We describe the algorithm of **A** as follows.

Algorithm $\mathbf{A}^{\mathcal{E}_\mathsf{k}\text{-SCP02}(.)}$

```
Get $C$ from eavesdropping
found ⟵ false
repeat
    $M' \longleftarrow$ guess($C$)
    $C' \longleftarrow \mathcal{E}_\mathsf{k}$-SCP02($M'$)
    if C = C' then
        found ⟵ true
    end if
until found = true
return $M'$
```

where `guess` is a function that takes a ciphertext C as input and returns one possible decryption of C for each call. We notice that the adversary keeps on making guesses until finding the message that encrypts to the eavesdropped ciphertext. Therefore, this theoretical attack is efficient against data with limited values and thus of low entropy, but it is worthless in case the exchanged data takes random values or their format is not known in advance.

We acknowledge that Try-and-Guess attack (TaGa) as outlined above has been previously suggested in other contexts (see [2, 12, 14]). Nevertheless, we believe that there is value in reiterating the discussion about this security flaw. The fact that the *de facto* standard of the sensitive industry of smart cards is still vulnerable to such attacks is of great interest. It indicates how the security community is divided between those designing theoretical cryptosystems and those implementing them in the real-world. We hope that our work would constitute a step towards bridging this gap.

In view of the ongoing popularity of SCP02, we believe that this vulnerability has not been identified yet. To the best of our knowledge, our work is the first one to apply TaGa in the context of GP specification for smart cards.

3.3 Plaintext Recovery Against Smart Cards

Here, we illustrate the fallout of TaGa by a theoretical, yet real-world, attack scenario. Our attack applies to smart cards following the GP model for content management.

Actors. We define four actors to describe the plot of the attack: (1) a *trusted service manager* (TSM) who owns a security domain on a smart card; (2) a *victim* who uses the said smart card to execute some critical services; (3) an *honest service provider* offering a sensitive service to the victim (e.g. payment); and (4) a *malicious service provider* that offers some service to the victim, but mainly aims to compromise the other services.

Threat Model. The intent of the attacker is to recover some sensitive data related to applications installed on smart cards.

For this purpose, we assume that the adversary is capable of installing an application on the targeted smart card. Any service provider does have this ability via a TSM. In addition, the application which the adversary is supposed to install includes no harmful behaviors. In particular, it does not attempt to attack the card system. Moreover, we assume that the adversary partially controls all communications with the card: she can drop and eavesdrop any exchanged message. Finally, we suppose that the adversary is targeting a well-protected card, and thus no direct attack is possible. This implies several assumptions. First, the card system shall contain no logical security flaw. Second, the card shall implement the appropriate countermeasures to withstand hardware attacks. Third, its security domains shall have been created and personalized with random keys. We emphasize that these assumptions are highly plausible for the smart card industry where products undergo extensive verification tests [36].

To sum up, in order to succeed her attack, the adversary should succeed in recovering the data while being transferred between the card and the TSM. This implies to break the encryption scheme implemented by the security domain. Being remote and software-only, our model represents a new kind of threat, since most related work involve some sophisticated hardware attacks [1, 25]. Our model provides several advantages over those defined in the literature, since it concerns a large number of smart cards regardless of their manufacturers. Indeed, our attack solely involves details defined in the GP card specifications which are common to all GP compliant cards.

Attack Workflow. We suppose that the attacker has already convinced the TSM to install her application. The attack is structured into two phases.

During the first phase, the honest service provider needs to personalize her application with some secret data. She sends her query to the TSM that is responsible to carry out the secure communication to the smart card. The malicious provider detects this event and reacts accordingly. She starts by asking the TSM to send some dummy query to the smart card. Thus, the TSM shares the established secure session with the two service providers. Afterward, the attacker intercepts the encrypted messages, grabs that of the honest service provider, and drops hers (easily recognizable by, for instance, its header).

As for the second phase, the attacker makes some guess, asks the TSM to encrypt it, and then intercepts the produced ciphertext which she discards whenever the guess was wrong. The attacker repeats this until she succeeds.

One technical issue might rise by this scenario: SCP02 instructs to double-encrypt sensitive data by the TSM. Data are firstly encrypted by ECB (Electronic Code Book) mode before applying the encryption of the secure channel. We argue that this has no impact on our attack, since the overall encryption remains stateless and deterministic. Indeed, ECB is deterministic, and the composition of two deterministic functions is clearly deterministic.

3.4 Discussion about Theoretical Feasibility

Several conditions must be met before the attacker can successfully recover some sensitive data. Below, we present these conditions and discuss their relevance.

Using One Common Session. First and foremost, the attacker must encrypt her test cases with the same key that encrypts the data to be recovered. This supposes that the TSM shares a secure session between different service providers. Some might argue that this is not a trivial requirement, and therefore our attack scenario cannot be mounted in practice. However, we argue that session sharing is not uncommon for three reasons. First, there is no mention in the specifications that could be understood as it is *bad* practice to share sessions or even SD. Second, SCP02 generates its session keys by encrypting some constants concatenating to a 2-byte counter. Thus, the TSM must change its master key after only 2^{16} sessions, which makes the TSM very eager to optimize the opening of secure sessions. Third, being expensive, the TSM is also eager to reduce the number of its leased SDs. Thus, it might install several applications into the same SD for the service providers that are not willing to pay the cost of having their own SD.

Synchronization. The TSM accepts to continue sending the attacker queries without receiving any acknowledgment. As a matter of fact, this mode of asynchronous communication is often employed for optimization. Indeed, the transmission rate of smart cards is slow [38]. Therefore, the TSM usually pushes all the commands to the terminal. The terminal forwards them to the associated card, and then collects all the returned values to send them back to the TSM. Such method of communication helps improve not only the communication time, bu also the undetectability of the attack, since the attacker application needs not to secretly include a special mode to manage all the sent commands during a session of TaGa. The attacker just intercepts and drops them.

Low-Entropy Data. This requirement is essential to succeed the Try-and-Guess attack. Low-entropy data are not rare in the context of smart cards. First, applications on smart cards often process enumerated variables with limited choices. This includes variables representing numerical values (e.g. amount of money), since integers are generally coded by two bytes in smart cards (JavaCard v2 [11] only supports signed `short` as numerical type). In practice, 4-byte PIN codes ($\leq 10,000$ choices) are also considered as low-entropy data. Second, despite the length of plaintext, the format used for numerous card applications is quite predictable. Data, like those of GP commands [23] and the EMV standard [17], are often structured with ASN.1 BER-TLV [27]. Such a format contains at least two public bytes: the tag value and the data length which the adversary already knows. In addition, the padding in SCP02 is constant and public. We illustrate by an example. The attacker wants to know how much money the user has provisioned her payment application. If the payment application is GP compliant, the provisioning command will be the GP command `Store Data`. Thus, the plaintext to be recovered is of the form:

`Tag`(1 byte) || `Length`(1 byte) || `Value`(2 bytes) || `Padding`(4 bytes)

Within these eight bytes, the only bytes to be guessed are those of the `Value`. Therefore, there are no more than $2^{15} = 32,768$ choices, due to the fact that money should always remain positive. In practice, much fewer queries are required, since specific amounts of money are often suggested for account recharge.

4 Secure Channel Protocol '3'

4.1 Description

SCP03, published as an amendment to card specification 2.2 [22], defines a new set of cryptographic methods based on AES. Similar to SCP02, it requires secret keys and padding, since it relies on block ciphers. SCP03 uses the "Encrypt-then-MAC" (EtM) method in which the ciphertext is produced by encrypting the message and then appending the result of applying the MAC scheme to the encrypted message. Refer to Construction 1 for more details about SCP03.

Construction 1 *(SCP03 Algorithms for Encryption and Decryption)*
Let E_{k} be an l-block-cipher and let $\mathrm{CBC}[E_{\mathsf{k}}] = (\mathcal{K}\text{-CBC}, \mathcal{E}\text{-CBC}, \mathcal{D}\text{-CBC})$ be a CBC encryption scheme that explicitly takes the iv vector as input. Let $\mathcal{MA} = (\mathcal{K}', \mathcal{T}, \mathcal{V})$ be a message authentication scheme. Let `padding` be a stateless deterministic encoding scheme and let `Len` be a function returning the length of its input. For the sake of clarity, we do not include `padding` in the described algorithms. For $M \in \{0,1\}^{ln}$ with the variables `counter` and `chained` properly initialized, the scheme SCP03-EtM $= (\mathcal{K}, \mathcal{E}, \mathcal{D})$ is defined as follows:

```
Encryption $\mathcal{E}_{\mathsf{k}}(M)$
$iv \longleftarrow E_{\mathsf{k1}}(\texttt{counter++})$
$C \longleftarrow \mathcal{E}_{\mathsf{k1}}\text{-CBC}(iv, M)$
$C' \longleftarrow \texttt{Len}(C) \,||\, C$
$\tau_1 || \tau_2 \longleftarrow \mathcal{T}_{\mathsf{k2}}(\texttt{chained} \,||\, C')$
$\texttt{chained} \longleftarrow \tau_1 || \tau_2$
return $C' \,||\, \tau_1$
```

```
Decryption $\mathcal{D}_{\mathsf{k}}(C)$
Parse $C$ as $\texttt{Len}(C') \,||\, C' \,||\, \tau$
if cannot parse then return $\perp$
$C'' \longleftarrow \texttt{chained} \,||\, \texttt{Len}(C') \,||\, C'$
$\tau_1 || \tau_2 \longleftarrow \mathcal{T}_{\mathsf{k2}}(C'')$
if $\tau_1 \neq \tau$ then
    return $\perp$ and halt
end if
$\texttt{chained} \longleftarrow \tau_1 || \tau_2$
$iv \longleftarrow E_{\mathsf{k1}}(\texttt{counter++})$
return $\mathcal{D}_{\mathsf{k1}}\text{-CBC}(iv, C')$
```

We highlight four points in the construction above. First, SCP03 ensures that all the message inputs to $\mathcal{T}_\mathsf{k}$ and $\mathcal{V}_\mathsf{k}$ are encoded. The encoding consists of appending the length of the input (i.e. C). Such encoding makes the set of inputs 'prefix-free', which means that no input can be the prefix of another one. This is an important requirement, since many MAC schemes, like CBC-MAC [5], are secure only for prefix-free set of inputs. Second, we notice that SCP03 ends the opened secure session when a decryption fails. This approach of "halting state" makes SCP03 vulnerable to denial-of-service attacks. However, it is effective against chosen-ciphertext attacks, since all the ensuing ciphertexts will not be decrypted, and therefore a new session with new keys has to be re-negotiated. This makes such attacks more detectable and less likely to succeed. Third, we do not include the padding method of SCP03 (recommended in ISO/IEC 10116:2006 [26]), since Paterson et al. prove that padding has no negative impact on security when it is used in encryption schemes following the EtM construction (like SCP03) [37]. Fourth, the MAC construction is quite peculiar: only half of the MAC (i.e. 8 bytes) is included with the ciphertext, and the remainder is reconstructed during MAC verification. The other half is somehow used as a 'state' between the sender and the receiver. To the best of our knowledge, GlobalPlatform has never provided the rationale behind this unusual construction that complicates the analysis of SCP03. However, we can plausibly assume that this choice was made to reduce the communication overhead incurred by SCP03. Indeed, the transmission rate with the card is low and it greatly increases with respect to the number of the communicated packets (as a matter of fact, the packet length is limited to 255 bytes) [38]. Therefore, despite being so small in other contexts, the overhead of transferring some extra 8 bytes might not be negligible in the case of smart cards.

4.2 Security Models

At first glance, SCP03 seems to fall into the EtM paradigm. Naturally, this raises no question regarding its security, since its generic security is proved in [8]. Here, we prove that SCP03 offers more than the standard security notions.

The construction of SCP03 described in 4.1 brings out three points that should be underlined. First, both the encryption and decryption algorithms involve the use of two variables that maintain their values and get updated after each call. These two variables must be '*in-sync*' between the sender and the receiver, otherwise $\mathcal{D}_\mathsf{k}(.)$ returns $\perp$. Second, the encryption of messages could be seen as a stateful nonce-based CBC encryption scheme. Third, the `chained` variable serves much the same purpose that a tweak does. Taking into consideration these three notes, we can turn the EtM construction of SCP03 into another composite. We start by introducing the two underlying blocks that will compose our new equivalent construction of SCP03.

Analyzing SCP03 via a New Construction.

Definition 6. *(Stateful Nonce-based Symmetric Scheme (Sf-n$\mathcal{SE}$)).* Let $n\mathcal{SE} = (n\mathcal{K}, n\mathcal{E}, n\mathcal{D})$ be a nonce-based encryption scheme. Let `counter` be a static variable initialized by 0 and which maintains its value between calls. For a message M, we define the associated stateful scheme Sf-n$\mathcal{SE} = (n\mathcal{K}\text{-Sf}, n\mathcal{E}\text{-Sf}, n\mathcal{D}\text{-Sf})$ as follows: $n\mathcal{E}_{\mathsf{k}}\text{-Sf}(M) = n\mathcal{E}_{\mathsf{k}}(\texttt{counter++}, M)$ and $n\mathcal{D}_{\mathsf{k}}\text{-Sf}(C) = n\mathcal{D}_{\mathsf{k}}(\texttt{counter++}, C)$.

Definition 7. *(Tweak Chaining MAC ($\mathcal{TC}$-$\widetilde{\mathcal{MA}}$)).* Let $\widetilde{F}_{\mathsf{k}} : \texttt{TWEAK} \times \texttt{MSG} \longrightarrow \texttt{TWEAK}$ be a tweakable MAC function for all key $\mathsf{K} \in \texttt{Key}$. Then, we define the associated chaining scheme $\mathcal{TC}$-$\widetilde{\mathcal{MA}} = (\widetilde{\mathcal{K}}, \widetilde{\mathcal{T}}, \widetilde{\mathcal{V}})$:

Tagging $\widetilde{\mathcal{T}}_{\mathsf{k}}(M)$

```
$\tau_1||\tau_2 \longleftarrow \widetilde{F}_{\mathsf{k}}(\texttt{chained}, M)$
$\texttt{chained} \longleftarrow \tau_1||\tau_2$
return $\tau_1$
```

Verification $\widetilde{\mathcal{V}}_{\mathsf{k}}(M, \tau)$

```
$\tau_1||\tau_2 \longleftarrow \widetilde{F}_{\mathsf{k}}(\texttt{chained}, M)$
$b \longleftarrow [\tau_1 = \tau]$
$\texttt{chained} \longleftarrow \tau_1 \,||\, \tau_2$
return $b$
```

Construction 2 *(Stateful Nonce-based Encrypt-then-Tweak (Sf-nEtTw))*
Let Sf-n$\mathcal{SE} = (n\mathcal{K}\text{-Sf}, n\mathcal{E}\text{-Sf}, n\mathcal{D}\text{-Sf})$ be a stateful nonce-based symmetric scheme. Let $(\mathsf{Enc}, \mathsf{Dec})$ be a prefix-free encoding scheme. Let $\mathcal{TC}$-$\widetilde{\mathcal{MA}} = (\widetilde{\mathcal{K}}, \widetilde{\mathcal{T}}, \widetilde{\mathcal{V}})$ be a tweak chaining MAC. Given a message M, we define the composite stateful nonce-based Encrypt-then-Tweak scheme **Sf-nEtTw** $= (\widetilde{\mathcal{K}}\text{-Sf}, \widetilde{\mathcal{E}}\text{-Sf}, \widetilde{\mathcal{D}}\text{-Sf})$:

Encryption $\widetilde{\mathcal{E}}_{\mathsf{k}}\text{-Sf}(M)$

```
$C \longleftarrow n\mathcal{E}_{\mathsf{k1}}\text{-Sf}(M)$
$C' \longleftarrow \mathsf{Enc}(C)$
$\tau \longleftarrow \widetilde{\mathcal{T}}_{\mathsf{k2}}(C')$
return $C' \,||\, \tau$
```

Decryption $\widetilde{\mathcal{D}}_{\mathsf{k}}\text{-Sf}(C)$

```
Parse $C$ as $C' \,||\, \tau$
$C'' \longleftarrow \mathsf{Dec}(C')$ or return $\perp$
if $\widetilde{\mathcal{V}}_{\mathsf{k2}}(C', \tau) \neq 1$ then
    return $\perp$ and halt
end if
return $n\mathcal{D}_{\mathsf{k1}}\text{-Sf}(C'')$
```

Now, let's see if the Construction 1 actually implies the definition of Construction 2. We start by examining whether the SCP03 operation $\mathcal{T}_{\mathsf{k}}(\texttt{chained} \,||\, C)$ is indeed a secure tweakable MAC function. We notice that the MAC computation in SCP03 is based on CMAC as specified in [16]. As mentioned by the author, CMAC is equivalent to OMAC that is defined in [28]. We rely on the result of [10] in which authors prove that $\text{OMAC}(T \,||\, M)$ is an $\widetilde{\text{IND}}$-CPA tweakable extension of OMAC. Hence, $\mathcal{T}_{\mathsf{k}}(\texttt{chained} \,||\, C) = \widetilde{F}_{\mathsf{k}}(\texttt{chained}, C)$, where $\widetilde{F}_{\mathsf{k}}$ is a tweakable function. Then, we investigate the security of the SCP03 encryption scheme that could be seen as a stateful variant of CBC1 recommended by the NIST in [15] and broken in [39]. CBC1 is a nonce-based scheme that encrypts the nonce to use it as IV. Unlike the insecure CBC1, the stateful CBC1 is IND-CPA secure. The intuition behind this is that attacks against CBC1 generally involve a craftily chosen nonce, and therefore they are not applicable against the stateful CBC1 where nonces are taken as a counter. A full proof is given in Theorem 17 in [4].

Security Notions. A new concrete security treatment is required in order to capture the power of Sf-nEtTw. Here, we outline the security concepts that we will use to study SCP03 and that are formalized by Bellare et al. in [6] and [9].

Definition 8. *(Indistinguishability under stateful CCA (IND-SFCCA)).* Conventionally, we consider an experiment in which the adversary $\mathbf{A}$ has access to a *left-or-right* encryption oracle $\mathcal{E}_{\mathsf{k}}(\mathcal{LR}(.,.,b))$ and a decryption oracle $\mathcal{D}_{\mathsf{k}}(.)$. $\mathcal{D}_{\mathsf{k}}$ returns the result of the decryption when $\mathbf{A}$ makes an *out-of-sync* query. A query is out-of-sync if it satisfies one of these conditions: (1) there are more queries to the decryption oracle than to the encryption one; (2) the ciphertext inside the decryption query is different from the last one computed by $\mathcal{E}_{\mathsf{k}}(\mathcal{LR}(.,.,b))$. As long as $\mathbf{A}$ does not make out-of-sync queries, $\mathcal{D}_{\mathsf{k}}$ updates its internal state, but returns nothing.

Definition 9. *(Integrity of stateful ciphertext (INT-SFCTXT)).* Here, we consider an experiment in which the adversary $\mathbf{A}$ has access to an encryption oracle $\mathcal{E}_{\mathsf{k}}(.)$ as well as a decryption oracle $\mathcal{D}_{\mathsf{k}}(.)$. The scheme is INT-SFCTXT secure if for all polynomial-time adversaries, it is hard to find an out-of-sync C, such that $\mathcal{D}_{\mathsf{k}}(C) \neq \perp$ and C was not produced by $\mathcal{E}_{\mathsf{k}}$. Similarly to IND-SFCCA, $\mathcal{D}_{\mathsf{k}}$ updates its internal state and returns nothing if no out-of-sync query is sent.

Definition 10. *(Algorithm-Substitution Attacks (ASA)).* Motivated by the potential threat of subverting implementations of cryptographic algorithms, Bellare, Paterson and Rogaway in [9] have recently defined ASA security by identifying two adversarial goals – *conducting surveillance* and *avoid detection*. In the ASA experiment, given user's key K and a subversion key $\widetilde{\mathsf{K}}$, the adversary $\mathcal{B}$ (also called big brother) wants to subvert the encryption algorithm $\mathcal{E}_{\mathsf{k}}$ by another one $\widetilde{\mathcal{E}_{\widetilde{\mathsf{k}}}}$. $\mathcal{B}$ requires that the subversion be both successful and undetectable. Here, we focus solely on the surveillance goal (SURV). SURV means that from observing ciphertexts, $\mathcal{B}$ can compromise confidentiality. Stated formally, SURV is defined as a classical distinguishing experiment when given oracle access to one of these two algorithms (i.e. $\mathcal{E}_{\mathsf{k}}$ and $\widetilde{\mathcal{E}_{\widetilde{\mathsf{k}}}}$). Indeed, $\mathcal{B}$, who has access to K but not to $\widetilde{\mathsf{K}}$, is required to distinguish $\mathcal{E}_{\mathsf{k}}$ from $\widetilde{\mathcal{E}_{\widetilde{\mathsf{k}}}}$. We say that an encryption scheme is ASA secure if no adversary $\mathcal{B}$ can succeed the SURV distinguishing game.

Definition 11. *(Unique Ciphertexts (UQ-CTXT)).* Following their work to defeat ASA, Bellare et al. define the notion of 'Unique Ciphertexts' as follows. Let $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be a symmetric encryption scheme. Given a secret key K, a message M, and a state τ, let $\mathcal{C}_{\mathcal{SE}}(\mathsf{K}, M, \tau)$ be the set of all ciphertexts such that $\mathcal{D}^{\tau}_{\mathsf{k}}(C)$ (also denoted $\mathcal{D}_{\mathsf{k}}(C_{\tau})$) returns M. We say that $\mathcal{SE}$ has unique ciphertexts (i.e. UQ-CTXT secure) if the set $\mathcal{C}_{\mathcal{SE}}(\mathsf{K}, M, \tau)$ has size at most one for all K, M, τ. Stated differently, for any given key, message and state, there exists at most one ciphertext that decrypts to the message in question.

Result 4.1 [*Unique Ciphertexts $\Longrightarrow$ ASA resilience*] *[9]*
In other words, let $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be a unique ciphertext symmetric encryption

scheme, and let $\mathcal{B}$ be a SURV adversary. Then, $\mathcal{B}$ cannot succeed the SURV experiment; which means that $\mathcal{SE}$ is resilient to ASA.

5 SCP03 Security Results

We now state our security results regarding SCP03. We provably show that SCP03 protects the integrity and the confidentiality of messages against chosen-plaintext and chosen-ciphertext attacks. In addition, it resists replay, out-of-delivery and algorithm-substitution attacks (ASAs). Indeed, authors of [6,9] prove that cryptographic schemes satisfying IND-SFCCA, INT-SFCTXT and Unique Ciphertexts meet all the security notions mentioned above.

5.1 Sf-nEtTw Security Analysis

In order to prove that SCP03 is IND-SFCCA and INT-SFCTXT secure, we start by analyzing the composite encryption scheme Sf-nEtTw. The following proposition concerns the security properties of $\mathcal{TC}\text{-}\widetilde{\mathcal{MA}}$.

Proposition 1. *(Upper Bound of* $\mathbf{Adv}^{\textit{suf-cma}}_{\mathcal{TC}\text{-}\widetilde{\mathcal{MA}}}(A)$*).* Let $\widetilde{F}_{\mathsf{k}} : \{0,1\}^n \times \mathtt{MSG} \longrightarrow \{0,1\}^n$ be a tweakable function and let $\mathcal{TC}\text{-}\widetilde{\mathcal{MA}}$ be its associated chaining scheme. Let $\mathbf{A}$ be an SUF-CMA adversary against $\mathcal{TC}\text{-}\widetilde{\mathcal{MA}}$ who queries q messages. Then, we can construct a distinguisher $\mathbf{D}$ against $\widetilde{F}$ such that

$$\mathbf{Adv}^{\textit{suf-cma}}_{\mathcal{TC}\text{-}\widetilde{\mathcal{MA}}}(A) \leq \mathbf{Adv}^{\widetilde{\textit{ind}}\textit{-cpa}}_{\widetilde{F}}(D) + \frac{q^2}{2^n} + \frac{1}{2^{n/2}} + \mathsf{Pr}[\mathsf{Col}_\mathsf{q}]$$

Proof. The proof is given in Appendix A.

We now show how schemes following the construction of Sf-nEtTw protect their stateful integrity of ciphertext (i.e. INT-SFCTXT).

Theorem 1. *(Upper Bound of* $\mathbf{Adv}^{\textit{int-sfctxt}}_{\textit{Sf-nEtTw}}(A)$*).* Let Sf-nEtTw be a scheme of stateful nonce-based encryption Sf-n$\mathcal{SE}$ = ($\mathcal{K}$-Sf, $n\mathcal{E}$-Sf, $n\mathcal{D}$-Sf) associated to a tweak chaining MAC $\mathcal{TC}\text{-}\widetilde{\mathcal{MA}} = (\widetilde{\mathcal{K}}, \widetilde{\mathcal{T}}, \widetilde{\mathcal{V}})$ and a prefix-free encoding scheme $(\mathsf{Enc}, \mathsf{Dec})$ as described in Construction 2. Let $\widetilde{F}_{\mathsf{k}} : \{0,1\}^n \times \mathtt{MSG} \longrightarrow \{0,1\}^n$ be the tweakable MAC function related to $\mathcal{TC}\text{-}\widetilde{\mathcal{MA}}$. Consider any INT-SFCTXT adversary $\mathbf{A}$ against Sf-nEtTw who asks to encrypt q messages, we can construct an SUF-CMA adversary $\mathbf{B}$ against $\mathcal{TC}\text{-}\widetilde{\mathcal{MA}}$ such that:

$$\mathbf{Adv}^{\textit{int-sfctxt}}_{\text{Sf-nEtTw}}(A) \leq \mathbf{Adv}^{\textit{suf-cma}}_{\mathcal{TC}\text{-}\widetilde{\mathcal{MA}}}(B) + \mathsf{Pr}[\mathsf{q\text{-}Col}]$$

where, given a message M and a list S containing q outputs of $\widetilde{F}_{\mathsf{k}}$, $\mathsf{Pr}[\mathsf{q\text{-}Col}]$ is the probability that $\widetilde{F}_{\mathsf{k}}(M) \in \mathsf{S}$.

Proof. The proof is given in Appendix B.

5.2 SCP03 Security Analysis

Now, we give our concrete security results for the particular case of SCP03. This requires to compute the different collision probabilities when $\widetilde{\mathrm{OMAC}}_{\mathsf{k}}(T, M) = \mathrm{OMAC}_{\mathsf{k}}(T \,||\, M)$ is used as the tweakable function $\widetilde{F}_{\mathsf{k}}(.,.)$ for all tweak T and message M. Two results about OMAC collisions are stated in Appendix C.1.

SCP03 is both INT-SFCTXT and IND-SFCCA

We show here that SCP03 protects its stateful confidentiality and integrity against powerful adversaries who can perform chosen-ciphertext attacks (CCA).

Theorem 2. *(SCP03 is INT-SFCTXT Secure).* Let E_{k} be a block cipher of size n and let $\mathrm{OMAC}[E_{\mathsf{k}}](.)$ be its associated OMAC scheme. Let $\widetilde{\mathrm{OMAC}}_{\mathsf{k}}(.,.)$ be a tweakable function defined as $\widetilde{\mathrm{OMAC}}_{\mathsf{k}}(T, M) = \mathrm{OMAC}[E_{\mathsf{k}}](T \,||\, M)$ for all tweak T and message M. Given a prefix-free encoding scheme $(\mathsf{Enc}, \mathsf{Dec})$, a stateful nonce-based encryption $\mathrm{Sf}\text{-}n\mathcal{SE} = (\mathcal{K}\text{-Sf}, n\mathcal{E}\text{-Sf}, n\mathcal{D}\text{-Sf})$ and a tweak chaining MAC $\mathcal{TC}\text{-}\widetilde{\mathcal{MA}} = (\widetilde{\mathcal{K}}, \widetilde{\mathcal{T}}, \widetilde{\mathcal{V}})$ whose tweakable MAC function is $\widetilde{\mathrm{OMAC}}_{\mathsf{k}}$, we define SCP03 to be the composite scheme formed by following the Construction 2. Consider any INT-SFCTXT adversary $\mathbf{A}$ attacking SCP03 and asking to encrypt q messages, we can construct a distinguisher $\mathbf{D}$ against $\widetilde{\mathrm{OMAC}}$ and a negligible function negl such that:

$$\mathbf{Adv}^{\textit{int-sfctxt}}_{\mathrm{scp03}}(A) \leq \mathbf{Adv}^{\widetilde{\textit{ind-cpa}}}_{\widetilde{\mathrm{OMAC}}}(D) + \mathsf{negl}$$

Proof. Since SCP03 is a composite scheme formed by following the Sf-nEtTw construction, it satisfies the relations given in Section 5.1. By using Proposition 1 and Theorem 1, we can obtain that

$$\mathbf{Adv}^{\textit{int-sfctxt}}_{\mathrm{scp03}}(A) \leq \mathbf{Adv}^{\widetilde{\textit{ind-cpa}}}_{\widetilde{\mathrm{OMAC}}}(D) + \frac{q^2}{2^n} + \frac{1}{2^{n/2}} + \mathsf{Pr}[\mathsf{Col}_q] + \mathsf{Pr}[\mathsf{q\text{-}Col}]$$

where $\mathsf{Pr}[\mathsf{Col}_q]$ is the collision probability of the tweakable function $\widetilde{\mathrm{OMAC}}_{\mathsf{k}}$ after q messages and $\mathsf{Pr}[\mathsf{q\text{-}Col}]$, given a message M and a list S containing q outputs of $\widetilde{\mathrm{OMAC}}_{\mathsf{k}}$, is the probability that $\widetilde{\mathrm{OMAC}}_{\mathsf{k}}(M) \in \mathsf{S}$.

Now, we use the following lemma to conclude our proof.

Lemma 5.1. Given $n \in \mathbb{N}$, there is a negligible function negl such that:

$$\frac{q^2}{2^n} + \frac{1}{2^{n/2}} + \mathsf{Pr}[\mathsf{Col}_q] + \mathsf{Pr}[\mathsf{q\text{-}Col}] \leq \mathsf{negl}$$

where $\mathsf{Pr}[\mathsf{Col}_q]$ and $\mathsf{Pr}[\mathsf{q\text{-}Col}]$ are as defined above.

Proof. The proof is given in Appendix C.2.

Theorem 3. *(SCP03 is IND-SFCCA Secure).* Let E_{k} be a block cipher of size n and let $\mathrm{OMAC}[E_{\mathsf{k}}](.)$ be its associated OMAC scheme. Let $\widetilde{\mathrm{OMAC}}_{\mathsf{k}}(.,.)$ be a tweakable function defined as $\widetilde{\mathrm{OMAC}}_{\mathsf{k}}(T, M) = \mathrm{OMAC}[E_{\mathsf{k}}](T \,||\, M)$ for all tweak T and message M. Given a prefix-free encoding scheme $(\mathsf{Enc}, \mathsf{Dec})$, a stateful nonce-based encryption Sf-$n\mathcal{SE} = (\mathcal{K}\text{-Sf}, n\mathcal{E}\text{-Sf}, n\mathcal{D}\text{-Sf})$ and a tweak chaining MAC $\mathcal{TC}\text{-}\widetilde{\mathcal{MA}} = (\widetilde{\mathcal{K}}, \widetilde{\mathcal{T}}, \widetilde{\mathcal{V}})$ whose tweakable MAC function is $\widetilde{\mathrm{OMAC}}_{\mathsf{k}}$, we define SCP03 to be the composite scheme formed by following the Construction 2. Consider any IND-SFCCA adversary **A** against SCP03, we can construct an IND-CPA adversary **B** against Sf-$n\mathcal{SE}$ and an INT-SFCTXT adversary **F** against $\mathcal{TC}\text{-}\widetilde{\mathcal{MA}}$ such that:

$$\mathbf{Adv}^{\textit{ind-sfcca}}_{\mathrm{scp03}}(A) \leq \mathbf{Adv}^{\textit{ind-cpa}}_{\text{Sf-}n\mathcal{SE}}(B) + \mathbf{Adv}^{\textit{int-sfctxt}}_{\mathcal{TC}\text{-}\widetilde{\mathcal{MA}}}(F)$$

Proof. This theorem follows directly from the implication proved by Bellare et al. [6]: IND-CPA $\wedge$ INT-SFCTXT $\Longrightarrow$ IND-SFCCA. This means that if an encryption scheme is both IND-CPA and INT-SFCTXT secure, then it is also IND-SFCCA secure. Regarding the INT-SFCTXT security of SCP03, we have just proved it in Theorem 2. Now, let us consider the IND-CPA security property of SCP03. Notice that SCP03 is a variant of Encrypt-then-MAC. Therefore, it inherits the IND-CPA property of its encryption scheme [8]. Stated otherwise, if the underlying encryption scheme Sf-$n\mathcal{SE}$ is IND-CPA secure, then SCP03 is also IND-CPA secure, which concludes our proof.

SCP03 is ASA Resilient

Finally, we prove that SCP03 defends against ASA, hence also against mass surveillance.

Theorem 4. *(SCP03 has Unique Ciphertexts).* Let $\widetilde{\mathrm{OMAC}}_{\mathsf{k}}$ be a tweakable function as defined previously. Given a stateful nonce-based encryption Sf-$n\mathcal{SE} = (\mathcal{K}\text{-Sf}, n\mathcal{E}\text{-Sf}, n\mathcal{D}\text{-Sf})$ and a tweak chaining MAC $\mathcal{TC}\text{-}\widetilde{\mathcal{MA}} = (\widetilde{\mathcal{K}}, \widetilde{\mathcal{T}}, \widetilde{\mathcal{V}})$ whose tweakable MAC function is $\widetilde{\mathrm{OMAC}}_{\mathsf{k}}$, we define SCP03 to be the scheme formed by following the Construction 2. Then, SCP03 is UQ-CTXT secure.

Proof. Let C_i denote the ciphertext produced by encrypting the message M on the state i. Considering the SCP03 design (see Construction 2), we have

$$C_i = \sigma_i \,||\, \tau_i = n\mathcal{E}_{\mathsf{k1}}\text{-Sf}(M) \;||\; \widetilde{\mathcal{T}}_{\mathsf{k2}}(\sigma_i)$$

where $\mathsf{K1}$ and $\mathsf{K2}$ are two independent keys. Now, we study the probability of finding a triplet $(\mathsf{K} = \mathsf{K1}||\mathsf{K2}, M, i)$ so that $|\mathcal{C}_{\mathrm{scp03}}(\mathsf{K}, M, i)| > 1$. By definition, this is equal to the probability of finding a ciphertext C'_i such that: (1) $C'_i \neq C_i$ and (2) $\mathcal{D}_{\mathsf{k}}$-SCP03$(C'_i) = M'$, where $M' = M$. We distinguish two cases.

Case 1 ($\boldsymbol{\sigma'_i \neq \sigma_i}$). Here, we prove that this case and the event of finding C' are contradictory, thereby proving that $\mathsf{Pr}[\mathsf{case\ 1}] = 0$. Indeed, recall that $n\mathcal{SE} = (\mathcal{K}, n\mathcal{E}, n\mathcal{D})$ encrypts messages using a deterministic algorithm. Therefore, as a matter of fact, for a fixed nonce N, $n\mathcal{E}_{\mathsf{k1}}(N, M_1) \neq n\mathcal{E}_{\mathsf{k1}}(N, M_2)$ implies that $M_1 \neq M_2$. Also, we notice that the definition of the set $\mathcal{C}_{\text{scp03}}$ involves that the associated encryption scheme Sf-n$\mathcal{SE} = (n\mathcal{K}\text{-Sf}, n\mathcal{E}\text{-Sf}, n\mathcal{D}\text{-Sf})$ has called $n\mathcal{SE}$ algorithms with the same nonce for each state i. Then, the event $\sigma_i' \neq \sigma_i$ entails $\sigma_i' = n\mathcal{E}_{\mathsf{k1}}\text{-Sf}(M') \neq n\mathcal{E}_{\mathsf{k1}}\text{-Sf}(M) = \sigma_i$, which implies $M' \neq M$. This concludes our proof, since the definition of $\mathcal{C}_{\text{scp03}}$ includes that $M' = M$.

Case 2 ($\boldsymbol{\sigma_i' = \sigma_i}$). Since $C_i' \neq C_i$, this case implies that $\tau_i' \neq \tau_i$. Following the same argument of case 1, we prove that this case and the event of finding C' are contradictory, thereby proving that $\mathsf{Pr}[\mathsf{case\ 2}] = 0$. Indeed, recall that the tweakable MAC function $\widetilde{\text{OMAC}}_{\mathsf{k2}}$ generates its tag using a deterministic algorithm. Therefore, as a matter of fact, for a fixed tweak T, $\widetilde{\text{OMAC}}_{\mathsf{k2}}(T, \sigma_1) \neq \widetilde{\text{OMAC}}_{\mathsf{k2}}(T, \sigma_2)$ implies that $\sigma_1 \neq \sigma_2$. Similarly, the definition of $\mathcal{C}_{\text{scp03}}$ involves that the associated chaining MAC scheme $\mathcal{TC}\text{-}\widetilde{\mathcal{MA}} = (\widetilde{\mathcal{K}}, \widetilde{\mathcal{T}}, \widetilde{\mathcal{V}})$ has called $\widetilde{\text{OMAC}}_{\mathsf{k2}}$ with same tweak for each state i. Then, the event $\tau_i' \neq \tau_i$ entails $\tau_i' = \widetilde{\mathcal{T}}_{\mathsf{k2}}(\sigma_i') \neq \widetilde{\mathcal{T}}_{\mathsf{k2}}(\sigma_i) = \tau_i$, which implies $\sigma_i' \neq \sigma_i$. This concludes our proof, since the definition of case 2 includes that $\sigma_i' = \sigma_i$.

6 Discussion

An important aspect of any cryptanalysis is what it implies in practice. Our study reveals interesting facts about the family of SCP. In particular, two protocols are concerned: SCP02 and SCP03. Here, we discuss our findings.

While discussing our results, we are aware that provable security is not a silver bullet for security, as authors of [13] notice that several cryptographic schemes have been proved secure and then broken some years later. We argue that this fact does not nullify the interest of such a powerful security tool. Indeed, despite being imperfect, provable security has greatly helped ruling out a large class of attacks in security protocols. In addition, although its findings should not be taken as absolute, they constitute a general direction that aims at designing better cryptographic schemes.

THE VULNERABLE, YET POPULAR, SCP02. In Section 3, we see that, unlike extensive evaluation, provable security for certified products provides a strong guarantee of security without promoting complexity. Indeed, we demonstrated a theoretical attack against the protocol SCP02. In addition, we showed how some technical details about SCP02 make the attacker likely to succeed in the context of smart cards. Surprisingly, the presented attack arises from a fundamental design flaw in SCP02, which is the use of CBC mode with no IV.

It is not clear that why the SCP02 designers made such a choice. However, we might suspect that the reasons behind this are twofold. First, when the first variant of SCP02 was published in 2000, cryptographic results about using CBC mode with stateful nonce-based IVs were not well-established yet. Second, designers chose not to use random IVs in order to reduce the overhead of SCP02.

Indeed, a random IV must be appended to the sent ciphertext, thereby increasing the communication overhead with the smart card. In addition, the implementations of CBC mode in smart cards have been optimized to pre-generate some objects during the initialization of the cipher object. The problem is that choosing the IV is uniquely done together with the choice of the encryption key during the initialization phase. Therefore, constantly modifying the IV implies constant initialization of the cipher object that can no longer performs its optimization in advance. Thus, we argue that the real challenge of SCP02 was to achieve good performance in a limited environment, like a smart card, and still ensuring security. In the complex GP card specifications, the tiny detail of 'just keep using the same IV' might have passed unnoticed, especially that to the best of our knowledge, no formal analysis of SCP02 has been performed before.

Furthermore, identifying such a well-known vulnerability tells us something: smart cards industry has difficulty in catching up with the advances on cryptography. Finalized in 2003, SCP02 keeps existing, while other protocols have continuously been updated. Ironically, the stringent requirements of smart cards about security are both its strongest and weakest point: they do not make this technology only secure and trustworthy, but also so slow to improve. We illustrate by three examples. First, EMV [17], which is the actual standard of payment, still mandates the use of Triple DES with two independent keys instead of using AES (see Section 5.7 in the EMV Card Personalization Specification [18]). Second, numerous card manufacturers continue relying on SCP02, although SCP03 was published in 2009. For instance, NXP instructs the support of SCP02 and makes it optional for SCP03 for all its JCOP products that are certified EAL5+ [35]. Third, the SCP family (i.e. SCP02 and SCP03) still requires encrypting data using the CBC mode. As a matter of fact, Mitchell in [34] (and more recently Rogaway in [40]) promotes abandoning CBC for future designs.

The powerful SCP03. Introduced as an amendment in 2009, we have analyzed SCP03 in Sections 4 and 5 and have found that it provably satisfies strong security notions. Of a particular interest, we proved that SCP03 resists against the algorithm substitution attacks (ASAs) that could lead to secret mass surveillance [9]. This result is significant, as it increases the trust in the closed industry of smart cards. The advantages offered by SCP03 are clear: it is provably secure and it is being gradually implemented by card manufacturers. It is true that the added security comes with additional cost: maintaining a 2-byte counter (i.e state) per session as well as one more block cipher invocation per message (recall that the counter is encrypted in order to be used as an IV). However, modern smart cards include a dedicated cryptographic co-processor, hence the incurred overhead is very small.

Standards are particularly susceptible to significant modification. Therefore, we feel that the recently created GP 'Crypto Sub-Task Force' [24] may have a hard time justifying to wholly reconsider the design of the SCP family. Therefore, we advocate the deprecation of SCP02 as soon as possible and the switch over to SCP03 that should be included in the main specification instead of being an amendment. Our goal is to provide enough information to the GP community so that the Crypto Sub-Task Force can take an informed decision when deciding how to fix the current problems with SCP02. At this point, a quote from [6] seems appropriate: "*in the modern era of strong cryptography, it would seem counterintuitive to voluntarily use a protocol with low security when it is possible to fix the security (...) at low cost*".

References

1. Aumüller, C., Bier, P., Fischer, W., Hofreiter, P., Seifert, J.P.: Fault attacks on rsa with crt: Concrete results and practical countermeasures. In: Cryptographic Hardware and Embedded Systems - CHES 2002, Lecture Notes in Computer Science, vol. 2523, pp. 260–275. Springer-Verlag (2003)
2. Bard, G.V.: A challenging but feasible blockwise-adaptive chosen-plaintext attack on ssl. In: Proc. of the International Conference on Security and Cryptography. pp. 7–10. SECRYPT '06, INSTICC Press (2006)
3. Béguelin, S.Z.: Formalisation and verification of the globalplatform card specification using the b method. In: Construction and Analysis of Safe, Secure, and Interoperable Smart Devices, Lecture Notes in Computer Science, vol. 3956, pp. 155–173. Springer-Verlag (2005)
4. Bellare, M., Desai, A., Jokipii, E., Rogaway, P.: A concrete security treatment of symmetric encryption. In: Proceedings of the 38th Annual Symposium on Foundations of Computer Science. pp. 394–403. FOCS '97, IEEE (1997)
5. Bellare, M., Kilian, J., Rogaway, P.: The security of the cipher block chaining message authentication code. Journal of Computer and System Sciences 61(3), 362–399 (December 2000)
6. Bellare, M., Kohno, T., Namprempre, C.: Authenticated encryption in ssh: Provably fixing the ssh binary packet protocol. In: Proceedings of the 9th ACM Conference on Computer and Communications Security. pp. 1–11. CCS '02, ACM (2002)
7. Bellare, M., Namprempre, C.: Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. In: Advances in Cryptology – ASIACRYPT 2000, Lecture Notes in Computer Science, vol. 1976, pp. 531–545. Springer-Verlag (2000)
8. Bellare, M., Namprempre, C.: Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. Journal of Cryptology 21(4), 469–491 (September 2008)
9. Bellare, M., Paterson, K.G., Rogaway, P.: Security of symmetric encryption against mass surveillance. In: Advances in Cryptology – CRYPTO 2014, Lecture Notes in Computer Science, vol. 8616, pp. 1–19. Springer-Verlag (2014)
10. Bellare, M., Rogaway, P., Wagner, D.: The eax mode of operation. In: Fast Software Encryption – FSE 2004, Lecture Notes in Computer Science, vol. 3017, pp. 389–407. Springer-Verlag (2004)
11. Chen, Z.: Java Card Technology for Smart Cards: Architecture and Programmer's Guide. Addison-Wesley Longman Publishing Co., Inc. (2000)
12. Dai, W.: An attack against ssh2 protocol (February 2002), email to the SECSH Working Group available from `ftp://ftp.ietf.org/ietf-mail-archive/secsh/2002-02.mail`
13. Degabriele, J.P., Paterson, K., Watson, G.: Provable security in the real world. IEEE Security and Privacy 9(3), 33–41 (May 2011)

14. Duong, T., Rizzo, J.: Here come the xor ninjas. unpublished (2011)
15. Dworkin, M.: Recommendation for block cipher modes of operation: Methods and techniques. National Institute of Standards and Technology (NIST) (December 2001), nIST Special Publication 800-38A
16. Dworkin, M.: Recommendation for block cipher modes of operation: The cmac mode for authentication. National Institute of Standards and Technology (NIST) (November 2001), nIST Special Publication 800-38B
17. EMVCo: EMVCo Specification, `https://www.emvco.com/specifications.aspx`
18. EMVCo: Emv card personalization specification – version 1.1 (July 2007), `https://www.emvco.com/specifications.aspx?id=20`
19. Feix, B., Thiebeauld, H.: Defeating iso9797-1 mac algo 3 by combining side-channel and brute force techniques. Cryptology ePrint Archive, Report 2014/702 (2014)
20. Fouque, P.A., Joux, A., Martinet, G., Valette, F.: Authenticated on-line encryption. In: Selected Areas in Cryptography – SAC 2003, Lecture Notes in Computer Science, vol. 3006, pp. 145–159. Springer-Verlag (2003)
21. GlobalPlatform: The standard for managing applications on secure chip technology, `https://www.globalplatform.org`
22. GlobalPlatform: Secure channel protocol '3' – card specification v2.2 – amendment d v1.1.1 (July 2014), `http://www.globalplatform.org/specificationscard.asp`
23. GlobalPlatform: Globalplatform card specification v2.3 (October 2015), `http://www.globalplatform.org/specificationscard.asp`
24. GlobalPlatform: About globalplatform – security task force activities and achievements – 2016 activities and priorities (2016), `https://www.globalplatform.org/aboutustaskforcesSecurity.asp`
25. Hemme, L.: A differential fault attack against early rounds of (triple-)des. In: Cryptographic Hardware and Embedded Systems - CHES 2004, Lecture Notes in Computer Science, vol. 3156, pp. 254–267. Springer-Verlag (2004)
26. ISO/IEC JTC 1/SC 27: Information technology – security techniques – modes of operation for an n-bit block cipher. Tech. rep., International Organization for Standardization (February 2006)
27. ISO/IEC JTC 1/SC 6: Information technology – asn.1 encoding rules: Specification of basic encoding rules (ber), canonical encoding rules (cer) and distinguished encoding rules (der). Tech. rep., International Organization for Standardization (December 2002)
28. Iwata, T., Kurosawa, K.: Omac: One-key cbc mac. In: Fast Software Encryption – FSE 2003, Lecture Notes in Computer Science, vol. 2887, pp. 129–153. Springer-Verlag (2003)
29. Joux, A., Martinet, G., Valette, F.: Blockwise-adaptive attackers: Revisiting the (in)security of some provably secure encryption models: Cbc, gem, iacbc. In: Advances in Cryptology – CRYPTO 2002, Lecture Notes in Computer Science, vol. 2442, pp. 17–30. Springer-Verlag (2002)
30. Katz, J., Lindell, Y.: Introduction to Modern Cryptography. Chapman & Hall Book (2015), second edition
31. Katz, J., Yung, M.: Unforgeable encryption and chosen ciphertext secure modes of operation. In: Fast Software Encryption – FSE 2001, Lecture Notes in Computer Science, vol. 1978, pp. 284–299. Springer-Verlag (2001)
32. Liskov, M., Rivest, R.L., Wagner, D.: Tweakable block ciphers. In: Advances in Cryptology – CRYPTO 2002, Lecture Notes in Computer Science, vol. 2442, pp. 31–46. Springer-Verlag (2002)

33. Markantonakis, C.: The case for a secure multi-application smart card operating system. In: Proc. of the First International Workshop on Information Security. pp. 188–197. ISW '97, Springer-Verlag (1998)
34. Mitchell, C.J.: Error oracle attacks on cbc mode: Is there a future for cbc mode encryption? In: Information Security – ISC '05, Lecture Notes in Computer Science, vol. 3650, pp. 244–258. Springer-Verlag (2005)
35. NXP Semiconductors Germany Gmbh: Nxp j3e081_m64, j3e081_m66, j2e081_m64, j3e041_m66, j3e016_m66, j3e016_m64, j3e041_m64 secure smart card controller. Common Criteria for Information Technology Security Evaluation (August 2014), certification Report: NSCIB-CC-13-37761-CR2
36. Oracle: Java card protection profile – closed configuration. Common Criteria for Information Technology Security Evaluation (December 2012), certification Report: ANSSI-CC-PP-2010/07
37. Paterson, K.G., Watson, G.J.: Authenticated-encryption with padding: A formal security treatment. In: Cryptography and Security: From Theory to Applications, Lecture Notes in Computer Science, vol. 6805, pp. 83–107. Springer-Verlag (2012)
38. Rankl, W., Effing, W.: Smart Card Handbook. John Wiley & Sons, Inc., 4th edn. (June 2010)
39. Rogaway, P.: Nonce-based symmetric encryption. In: Fast Software Encryption – FSE 2004, Lecture Notes in Computer Science, vol. 3017, pp. 348–358. Springer-Verlag (2004)
40. Rogaway, P.: Evaluation of some blockcipher modes of operation. Tech. rep., Cryptography Research and Evaluation Committees (CRYPTREC) for the Government of Japan (2011)

A Proof of Proposition 1

We start by providing three definitions that we will use throughout our proof.

MAC Function. Here, we just recall how a MAC scheme is related to its MAC function. Let $\mathcal{MA}[F] = (\mathcal{K}, \mathcal{T}, \mathcal{V})$ be a MAC scheme based on the MAC function F. F takes as input a key K and a message M to output a tag τ. The tagging algorithm $\mathcal{T}_{\mathsf{k}}$ and the verification algorithm $\mathcal{V}_{\mathsf{k}}$ are defined as follows:

```
Tagging $\mathcal{T}_{\mathsf{k}}(M)$

$\tau \longleftarrow F_{\mathsf{k}}(M)$
return $\tau$
```

```
Verification $\mathcal{V}_{\mathsf{k}}(M, \tau)$

if $F_{\mathsf{k}}(M) = \tau$ then
    return 1
else
    return 0
end if
```

Truncated MAC. Let $T : \{0,1\}^n \longrightarrow \{0,1\}^{n_T}$ be a transformation function. Let $\mathcal{MA}[F] = (\mathcal{K}, \mathcal{T}, \mathcal{V})$ be a MAC scheme based on the MAC function F. We define the transformed MAC scheme $To\mathcal{MA} = (\mathcal{K}, To\mathcal{T}, To\mathcal{V})$ that uses ToF as its MAC function, where o denotes the composition operator. A *truncated MAC* is a transformed MAC in which $T(.)$ is the $\mathtt{MSB}_l(.)$ function that takes a message as input and returns the l most significant (i.e. left-most) bits.

Tweak Chaining MAC2 ($\mathcal{TC}$-$\widetilde{\mathcal{MA}}2$). Let $\widetilde{F}_{\mathsf{k}} : \{0,1\}^n \times \mathtt{MSG} \longrightarrow \{0,1\}^n$ be a tweakable function and let $\mathcal{TC}$-$\widetilde{\mathcal{MA}}$ be its associated chaining scheme. We define $\mathcal{TC}$-$\widetilde{\mathcal{MA}}2$ as $\mathcal{TC}$-$\widetilde{\mathcal{MA}}$ except that $\mathcal{TC}$-$\widetilde{\mathcal{MA}}2$ operates on the entire tag returned by $\widetilde{F}(.,.)$ and not only on its half as in $\mathcal{TC}$-$\widetilde{\mathcal{MA}}$. Stated differently, $\mathcal{TC}$-$\widetilde{\mathcal{MA}}2$ is a MAC scheme in which the MAC function $F2$ is defined as follows:

```
MAC Function $F2_{\mathsf{k}}(M)$

$\tau \longleftarrow \widetilde{F}_{\mathsf{k}}(\mathtt{chained}, M)$
chained $\longleftarrow \tau$
return $\tau$
```

where `chained` is a static variable (i.e. maintains its value between calls) that was initialized with $\mathbf{0}^n$.

Having thus presented the above definitions, we are now on a position to make our proof. Let $\widetilde{F}_{\mathsf{k}} : \{0,1\}^n \times \mathtt{MSG} \longrightarrow \{0,1\}^n$ be a tweakable function and let $\mathcal{TC}$-$\widetilde{\mathcal{MA}}2[F2]$ be its associated tweak chaining MAC2 scheme. We notice that the $\mathcal{TC}$-$\widetilde{\mathcal{MA}} = (\widetilde{\mathcal{K}}, \widetilde{\mathcal{T}}, \widetilde{\mathcal{V}})$ scheme presented in Definition 7 can be seen as the truncated MAC of $\mathcal{TC}$-$\widetilde{\mathcal{MA}}2[F2]$, where $T(.) = \mathtt{MSB}_{n/2}(.)$. Thus, we denote the MAC function of $\mathcal{TC}$-$\widetilde{\mathcal{MA}}$ as $ToF2$.

Consider any polynomial-time SUF-CMA adversary $\mathbf{A}$ against $\mathcal{TC}$-$\widetilde{\mathcal{MA}}$. Recall that $\mathbf{A}$ can make two types of queries: tagging queries and verification queries. We suppose that $\mathbf{A}$ makes q tagging queries. We associate two adversaries to $\mathbf{A}$: an sPRF adversary $\mathbf{B}$ against the MAC function $F2$ (or equivalently against $\mathcal{TC}\text{-}\widetilde{\mathcal{MA}}2[F2]$), and an $\widetilde{\text{IND}}$-CPA distinguisher $\mathbf{D}$ against the tweakable function $\widetilde{F}_{\mathsf{k}}$. Now, we state the following lemmas in which we define how the adversaries $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{D}$ interact between each other and from which the proposition 1 follows directly.

Lemma A.1. $\mathbf{Adv}^{\mathit{suf\text{-}cma}}_{\mathcal{TC}\text{-}\widetilde{\mathcal{MA}}}(A) = \mathbf{Adv}^{\mathit{sprf}}_{F2}(B) + 1/2^{n/2}$

Lemma A.2. $\mathbf{Adv}^{\mathit{sprf}}_{F2}(B) \leq \mathbf{Adv}^{\widetilde{\mathit{ind}}\text{-}\mathit{cpa}}_{\widetilde{F}}(D) + \mathsf{Pr}[\mathsf{Col}] + \mathsf{Pr}[\mathsf{Col_q}]$

Lemma A.3. $\mathsf{Pr}[\mathsf{Col}] \leq q^2/2^n$

Proof of Lemma A.1: Recall that $\mathbf{B}$ has access to the oracle $\mathcal{O}$ and her goal is to distinguish whether $\mathcal{O}$ is the MAC function $F2$ or the stateful random oracle $\mathcal{R}_{\mathcal{S}}$. Recall also that the MAC function of $\mathcal{TC}\text{-}\widetilde{\mathcal{MA}}$ is $ToF2$. The algorithm $\mathbf{B}$ is described below:

Algorithm $\mathbf{B}^{\mathcal{O}}$

```
repeat
    if A queries $(M)$ then
        $\tau \longleftarrow To\mathcal{O}(M)$
        output $\tau$ to A
    end if
    if A queries $(M, \tau)$ then
        $b \longleftarrow [\tau = To\mathcal{O}(M)]$
        output $b$ to A
    end if
until A ends
if A forges then
    return 1
else
    return 0
end if
```

We can see that $\mathbf{B}$ perfectly simulates the answers to $\mathbf{A}$. In addition, $\mathbf{B}$ returns 1 (i.e. guesses that the oracle $\mathcal{O}$ is the MAC function $F2$) when $\mathbf{A}$ succeeds in forging a tag. Therefore, the following relation holds:

$$\mathsf{Pr}[\mathbf{A}^{To\mathcal{O}} \text{ forges}] = \mathsf{Pr}[\mathbf{B}^{\mathcal{O}} \Rightarrow 1] \tag{1}$$

where we use the equivalent notation in which we note that the adversary $\mathbf{A}$ has access to the MAC function as oracle instead of the tagging/verification oracles.

By definition of strong unforgeability of the MAC scheme $\mathcal{TC}\text{-}\widetilde{\mathcal{MA}}$ (see Definition 2), the advantage of $\mathbf{A}$ is defined by the probability of her success when she has access to the oracle $ToF2$. Therefore, we have:

$$\begin{aligned}
\mathbf{Adv}^{suf\text{-}cma}_{\mathcal{TC}\text{-}\widetilde{\mathcal{MA}}}(\mathbf{A}) &= \mathsf{Pr}[\mathbf{A}^{ToF2} \text{ forges}] \\
&= \mathsf{Pr}[\mathbf{A}^{ToF2} \text{ forges}] + \big(\mathsf{Pr}[\mathbf{A}^{To\mathcal{R}_\mathcal{S}} \text{ forges}] - \mathsf{Pr}[\mathbf{A}^{To\mathcal{R}_\mathcal{S}} \text{ forges}]\big) \\
&= \big(\mathsf{Pr}[\mathbf{A}^{ToF2} \text{ forges}] - \mathsf{Pr}[\mathbf{A}^{To\mathcal{R}_\mathcal{S}} \text{ forges}]\big) + \mathsf{Pr}[\mathbf{A}^{To\mathcal{R}_\mathcal{S}} \text{ forges}] \\
&= \big(\mathsf{Pr}[\mathbf{B}^{F2} \Rightarrow 1] - \mathsf{Pr}[\mathbf{B}^{\mathcal{R}_\mathcal{S}} \Rightarrow 1]\big) + \mathsf{Pr}[\mathbf{A}^{To\mathcal{R}_\mathcal{S}} \text{ forges}] \text{ (from 1)} \\
&= \mathbf{Adv}^{sprf}_{F2}(B) + \mathsf{Pr}[\mathbf{A}^{To\mathcal{R}_\mathcal{S}} \text{ forges}]
\end{aligned}$$

Now, we examine $\mathsf{Pr}[\mathbf{A}^{To\mathcal{R}_\mathcal{S}} \text{ forges}]$, which is equal to the probability that $\mathbf{A}$ forges against a MAC scheme that has $To\mathcal{R}_\mathcal{S}$ as its MAC function. Recall that $\mathcal{R}_\mathcal{S}$ is a random oracle. Let us suppose that (M, τ) is the forging query that $\mathbf{A}$ uses to break the scheme. Therefore, the following relations holds: $\tau = T\left(\mathcal{R}_\mathcal{S}(M)\right)$. Thus, we conclude our proof by showing that we have:

$$\begin{aligned}
\mathsf{Pr}[\mathbf{A}^{To\mathcal{R}_\mathcal{S}} \text{ forges}] &= \mathsf{Pr}[x \xleftarrow{R} \{0,1\}^n,\ T(x) = \tau] \\
&= \frac{1}{2^{n/2}} \qquad \big(\text{since } T(.) = \mathtt{MSB}_{n/2}(.)\big)
\end{aligned}$$

Proof of Lemma A.2: Here, we consider any sPRF adversary $\mathbf{B}$ against $F2$ and we associate it to a particular $\widetilde{\text{IND}}$-CPA distinguisher $\mathbf{D}$ against the tweakable function $\widetilde{F}_\mathsf{k} : \{0,1\}^n \times \mathtt{MSG} \longrightarrow \{0,1\}^n$. Recall that $\mathbf{D}$ has access to the oracle $\mathcal{O}(.,.)$ and her goal is to distinguish whether $\mathcal{O}$ is $\widetilde{F}_\mathsf{k}(.,.)$ or $\widetilde{\mathcal{R}}(.,.)$, where $\widetilde{\mathcal{R}}(.,.)$ is a function that, on input (T, M), returns n-bit random strings. Recall also that $\mathbf{D}$ is a tweak-respecting adversary (i.e. does not repeat tweak). We define the algorithm of $\mathbf{D}$ as follows:

Algorithm $\mathbf{D}^{\mathcal{O}}$

```
$t \longleftarrow \mathbf{0}^n$
$\mathsf{S} \longleftarrow \{t\}$
repeat
    if B queries $(M)$ then
        $t \longleftarrow \mathcal{O}(t, M)$
        $\mathsf{S} \longleftarrow \mathsf{S} \cup \{t\}$
        output $t$ to B
        if $\mathsf{S}$ contains duplicate values then
            return 1
        end if
    end if
until B outputs $b'$
return $b'$
```

where S is a multiset in which values can repeat. We argue that when S does not contain the same value twice, $\mathbf{D}$ is perfectly simulating $\mathbf{B}$'s execution environment. This is true because $\widetilde{F}(.,.)$ is no distinguishable from the random oracle $\widetilde{\mathcal{R}}(.,.)$ only against tweak-respecting adversaries. We illustrate the importance of such a condition by an example. In our example, we take $\widetilde{\mathrm{OMAC}}(T, M)$ $(= \mathrm{OMAC}(T||M))$ as the tweakable function $\widetilde{F}(.,.)$. Now, we show that $\mathbf{B}$ can easily see under the simulation environment that she is not interacting with a random oracle. $\mathbf{B}$ knows that the initial tweak (i.e. state) is $\mathbf{0}^n$ and queries $M_1 = \mathbf{0}^n$ to receive τ_1 from her oracle. Then, let T be a tweak that repeats twice. For the first occurrence of T, $\mathbf{B}$ queries $M_2 = \mathbf{0}^n$ to receive τ_2 and for its second occurrence she queries $M_2 = \mathbf{0}^n \,||\, \tau_2 \,||\, \mathbf{0}^n$ to receive τ_3. It is easy to see that $\tau_1 = \tau_3$ when $\mathcal{O} = \widetilde{F}_{\mathsf{k}}(.,.)$ $(\neq \widetilde{\mathcal{R}}(.,.))$. Indeed, we have

$$\begin{aligned}
\tau_1 &= E_{\mathsf{k}}\left(E_{\mathsf{k}}(\mathbf{0^n})\right)\\
\tau_2 &= E_{\mathsf{k}}\left(E_{\mathsf{k}}(T)\right)\\
\tau_3 &= E_{\mathsf{k}}\left(E_{\mathsf{k}}\left(\cancel{E_{\mathsf{k}}\left(E_{\mathsf{k}}(T)\right)} \oplus \cancel{\tau_2}\right) \oplus \mathbf{0^n}\right)
\end{aligned}$$

Thus, from $\mathbf{D}$'s algorithm, we can see that

$$\begin{aligned}
\mathsf{Pr}[\mathbf{D}^{\widetilde{\mathbf{F}}} \Rightarrow 1] &= \mathsf{Pr}[\mathbf{B}^{F2} \Rightarrow 1] \ + \mathsf{Pr}[\mathsf{S}|\widetilde{F}]\\
\mathsf{Pr}[\mathbf{D}^{\widetilde{\mathcal{R}}} \Rightarrow 1] &= \mathsf{Pr}[\mathbf{B}^{\mathcal{R}_S} \Rightarrow 1] + \mathsf{Pr}[\mathsf{S}|\widetilde{\mathcal{R}}]
\end{aligned}$$

where $\mathsf{Pr}[\mathsf{S}]$ is the probability that the multiset S contains duplicate values.

By using the two above relations, we get

$$\begin{aligned}
\mathbf{Adv}^{sprf}_{F2}(B) &= \mathbf{Adv}^{\widetilde{ind\text{-}cpa}}_{\widetilde{F}}(D) + \overbrace{\mathsf{Pr}[\mathsf{S}|\mathcal{R}]}^{\mathsf{Pr}[\mathsf{Col}]} - \overbrace{\mathsf{Pr}[\mathsf{S}|\widetilde{F}]}^{\mathsf{Pr}[\mathsf{Col_q}]}\\
&\leq \mathbf{Adv}^{\widetilde{ind\text{-}cpa}}_{\widetilde{F}}(D) + \mathsf{Pr}[\mathsf{Col}] + \mathsf{Pr}[\mathsf{Col_q}]
\end{aligned}$$

where $\mathsf{Pr}[\mathsf{Col_q}]$ is the collision probability of the tweakable function $\widetilde{F}$ after q messages. Thus, our proof ends.

Proof of Lemma A.3: Informally speaking, the lemma means that the set $\{x : x_0 = \mathbf{0}^n,\ x_i = \mathcal{R}(x_0, .)\}$ has asymptotically negligible probability to include duplicate values. Recall that q is the number of $\mathbf{B}$'s queries. We start our proof by making induction on q. For all $q \geq 1$, we prove that

$$\mathsf{Pr}[\mathsf{Col}] = \frac{q(q+1)}{2^{n+1}} \tag{2}$$

Then, we conclude our proof by noticing that $q(q+1)/2^{n+1} \leq q^2/2^n$.

Base case. When $q = 1$, the right side of (2) is $1/2^n$. Now, let's look at the left side. After only one call, there are two elements in S: $\{\mathbf{0}^n, y\}$, where $y \longleftarrow \mathcal{R}(\mathbf{0}^n, .)$. Thus, $\mathsf{Pr}[\mathsf{Col}] = \mathsf{Pr}[x \overset{R}{\leftarrow} \{0,1\}^n, x = \mathbf{0}^n]$, which is equal to $1/2^n$.

Induction step. Suppose that the equation 2 is true for $q = m - 1$. Here, x_i denotes the ith element of the multiset S. After $q = m$ calls, we have

$$\begin{aligned}\mathsf{Pr}[\mathsf{Col}] &= \overbrace{\mathsf{Pr}[\mathsf{Col\ after\ m-1\ calls}]}^{\text{induction hypothesis}} + \mathsf{Pr}[x_m \in \mathsf{S}] \\ &= \frac{m(m-1)}{2^{n+1}} + \frac{m}{2^n} \\ &= \frac{m(m+1)}{2^{n+1}}\end{aligned}$$

Hence, the equation 2 holds for $q = m$, and the induction step is complete.

B Proof of Theorem 1

Recall that **A** can make two types of queries: encryption queries and decryption queries. We denote **A**'s i-th encryption query as M_i and the returned ciphertext as $C_i = \sigma_i||\tau_i$. We denote **A**'s i-th decryption query as $C'_i = \sigma'_i||\tau'_i$ and the returned message as m_i. We associate to **A** an SUF-CMA forger **F** against $\widetilde{\mathcal{TC}\text{-}\mathcal{MA}}$. This association is similar to the one given in the Case 1 of Theorem 4: **F** generates a key $\mathsf{K1} \in \mathtt{Key}$ that she uses for the encryption/decryption algorithms of Sf-n$\mathcal{SE}$. We recall that the forger **F** has access to two oracles: a tagging oracle $\widetilde{\mathcal{T}}_{\mathsf{k2}}$ and a verification oracle $\widetilde{\mathcal{V}}_{\mathsf{k2}}$, where the key $\mathsf{K2}$ is independent from $\mathsf{K1}$. Below, we describe our trivial association.

1. When **A** makes an encryption query M, **F** outputs $\sigma \longleftarrow \mathsf{Enc}\,(n\mathcal{E}_{\mathsf{k1}}\text{-Sf}(M))$. Then, she queries σ to her tagging oracle $\widetilde{\mathcal{T}}_{\mathsf{k2}}$ and receives τ in response. Finally, she outputs $C = \sigma\,||\,\tau$ to **A**.
2. When **A** makes a decryption query $C = \sigma\,||\,\tau$, the forger **F** queries τ to her verification oracle $\widetilde{\mathcal{V}}_{\mathsf{k2}}$ and receives a binary value b. If b is false, then **F** halts after outputting $\perp$. Otherwise, **F** computes $n\mathcal{D}_{\mathsf{k1}}\text{-Sf}\,(\mathsf{Dec}(\sigma))$ and outputs the result to **A**.
3. When **A** wins in her INT-SFCTXT experiment, namely providing a new valid out-of-sync decryption query $C = \sigma\,||\,\tau$, then **F** stops and attempts to evaluate the pair (σ, τ) in order to see whether she succeeds in her forgery. The different cases are presented below in the proof of Lemma B.1.

Now, suppose **A** has made q encryption queries and d decryption ones. Let j be the index of **A**'s first out-of-sync decryption query. We only consider the first out-of-sync query because if it fails, the decryption algorithm will return $\perp$ and halt for all ensuing queries (see our discussion about the approach of halting state in Section 4.1). We define two events in case the **A**'s j-th decryption query succeeds: (1) Col: $\exists i \leq q$ such that $\tau'_j = \tau_i$ and $i \neq j$; (2) Bad: $q \geq j$, $\tau'_j = \tau_j$ and $m_j = M_j$. We state the following lemmas from which Theorem 1 follows directly (using Proposition 1).

Lemma B.1. $\mathbf{Adv}^{\textit{int-sfctxt}}_{\textit{Sf-nEtTw}}(A) \leq \mathbf{Adv}^{\textit{suf-cma}}_{\mathcal{TC}\text{-}\widetilde{\mathcal{MA}}}(F) + \mathsf{Pr}[\mathsf{q\text{-}Col}] + \mathsf{Pr}[\mathsf{Bad}]$

Lemma B.2. $\mathsf{Pr}[\mathsf{Bad}] = 0$

Proof of Lemma B.1: As said previously, $\mathbf{A}$ made q encryption queries before her first out-of-sync query ($\mathcal{Q}$) which is the j-th decryption query ($C'_j = \sigma'_j \,||\, \tau'_j$). We define the following events.

E : $\mathcal{Q}$ correctly verifies

E_1 : E occurs and $\tau'_j \notin \{\tau_1, ..., \tau_q\}$

E_2 : E occurs and $\tau'_j \in \{\tau_1, ..., \tau_q\}$

$E_{2,1}$: E_2 occurs and either $q < j$ or $\tau'_j \neq \tau_j$

$E_{2,2}$: E_2 occurs and $q \geq j$ and $\tau'_j = \tau_j$

$E_{2,2,1}$: $E_{2,2}$ occurs and $m_j = M_j$

$E_{2,2,2}$: $E_{2,2}$ occurs and $m_j \neq M_j$

If $\mathcal{Q}$ fails, then $\mathbf{A}$ cannot win any more, since the decryption algorithm will return $\perp$ for any subsequent query. Therefore, $\mathbf{Adv}^{\textit{int-sfctxt}}_{\text{Sf-nEtTw}}(A) = \mathsf{Pr}[E]$. Considering the different events, we have $\mathsf{Pr}[E] = \mathsf{Pr}[E_1 \vee E_{2,2,2}] + \mathsf{Pr}[E_{2,1}] + \mathsf{Pr}[E_{2,2,1}]$.

Now, we study the probabilities of these events. We can see that $E_{2,1}$ corresponds to the event Col, since it implies that τ'_j has already been produced before and that was not during the j-th encryption query (this includes the fact that $\mathbf{A}$ might not have made j encryption queries yet). Concerning $E_{2,2,1}$, it is easy to see that it satisfies the definition of the Bad event. Consequently, we have

$$\mathbf{Adv}^{\textit{int-sfctxt}}_{\text{Sf-nEtTw}}(A) = \mathsf{Pr}[E_1 \vee E_{2,2,2}] + \mathsf{Pr}[\mathsf{Col}] + \mathsf{Pr}[\mathsf{Bad}]$$

We conclude the proof by examining $\mathsf{Pr}[E_1 \vee E_{2,2,2}]$ and $\mathsf{Pr}[\mathsf{Col}]$.

$\mathsf{Pr}[\boldsymbol{E_1} \vee \boldsymbol{E_{2,2,2}}]$. We notice that when the event $E_1 \vee E_{2,2,2}$ occurs, (i.e. the j-th decryption oracle $C'_j = \sigma'_j \,||\, \tau'_j$ does not return $\perp$), then the forger $\mathbf{F}$ succeeds in finding an SUF-CMA forgery against $\mathcal{TC}\text{-}\widetilde{\mathcal{MA}}$, since the two events ensure that the pair (m_j, τ'_j) was never produced before by the oracle $\widetilde{\mathcal{T}}_{\mathsf{k2}}$.

Indeed, the event E_1 implies that τ'_j has never been queried to $\widetilde{\mathcal{T}}_{\mathsf{k2}}$, while the event $E_{2,2,2}$ implies that the tag τ'_j has never been obtained from querying the oracle $\widetilde{\mathcal{T}}_{\mathsf{k2}}$ with σ'_j as input. This is because for any state i, the following implication is asymptotically true (i.e. n$\mathcal{E}_{\mathsf{k}}$-Sf(.) is injective):

$$M_i \neq M'_i \Longrightarrow n\mathcal{E}_{\mathsf{k1}}\text{-Sf}(M_i) \neq n\mathcal{E}_{\mathsf{k1}}\text{-Sf}(M'_i)$$

Therefore, τ'_j $(= \tau_j)$ was computed for $\sigma_j = n\mathcal{E}_{\mathsf{k1}}\text{-Sf}(M_j)$ which is different from σ'_j (i.e. $\sigma_j \neq \sigma'_j$), since $\sigma'_j = n\mathcal{E}_{\mathsf{k1}}\text{-Sf}(m_j)$ and $M_j \neq m_j$.

Thus, we have

$$\mathsf{Pr}[E_1 \vee E_{2,2,2}] = \mathsf{Pr}[\mathbf{F} forges] = \mathbf{Adv}^{suf\text{-}cma}_{\mathcal{TC}\text{-}\widetilde{\mathcal{MA}}}(F)$$

$\mathsf{Pr[Col]}$. As previously pointed out, $\mathsf{Pr[Col]} = \mathsf{Pr}[\exists i \neq j$ such that $\tau'_j = \tau_i]$. This means that for two different states the following equality holds:

$$\widetilde{\mathcal{T}}_{\mathsf{k2}}(\mathsf{Enc}(n\mathcal{E}_{\mathsf{k1}}\text{-Sf}(m_j))) = \widetilde{\mathcal{T}}_{\mathsf{k2}}(\mathsf{Enc}(n\mathcal{E}_{\mathsf{k1}}\text{-Sf}(M_i)))$$

The above relation supposes that the adversary should find a collision against $\mathcal{TC}\text{-}\widetilde{\mathcal{MA}}$ after q invocations to the $\widetilde{\mathcal{T}}_{\mathsf{k2}}(.)$ oracle, which corresponds to find a state $i(\neq j)$ such that the related MAC tag is equal to the one computed for the state j. By looking at the construction of $\mathcal{TC}\text{-}\widetilde{\mathcal{MA}}$ in Definition 7, we find that $\mathsf{Pr[Col]}$ is equivalent to the probability of encountering a collision against the underlying tweakable function $\widetilde{F}_{\mathsf{k2}}(.,.)$. Stated differently, we have

$$\mathsf{Pr[Col]} = \mathsf{Pr[q\text{-}Col]}$$

where, we recall that, given a message M and a list S containing q outputs of $\widetilde{F}_{\mathsf{k2}}$, $\mathsf{Pr[q\text{-}Col]}$ is the probability that $\widetilde{F}_{\mathsf{k2}}(M) \in \mathsf{S}$.

Proof of Lemma B.2: The event $E_{2,2,1}$ includes all the following events: (1) $q \geq j$; (2) the decryption query $C'_j = \sigma'_j||\tau'_j$ is out-of-sync, hence $C_j \neq C'_j$; (3) $\sigma'_j \neq \sigma_j$, since $\tau'_j = \tau_j$; and (4) $m_j = M_j$. We notice that the events 3 and 4 are contradictory, and therefore $\mathsf{Pr[Bad]} = 0$. Indeed, recall that $n\mathcal{SE} = (\mathcal{K}, n\mathcal{E}, n\mathcal{D})$ encrypts messages using a deterministic algorithm. Therefore, for a fixed nonce N, $n\mathcal{E}_{\mathsf{k1}}(N, M_1) \neq n\mathcal{E}_{\mathsf{k1}}(N, M_2)$ implies that $M_1 \neq M_2$. Also, we notice that the encryption and the decryption states were in-sync prior to the j-th decryption query. This means that the associated Sf-n$\mathcal{SE}$ = ($\mathcal{K}$-Sf, $n\mathcal{E}$-Sf, $n\mathcal{D}$-Sf) has called $n\mathcal{SE}$ algorithms with the same nonce for each state. Thus, the event 3 entails $\sigma'_j = n\mathcal{E}_{\mathsf{k1}}\text{-Sf}(m_j) \neq n\mathcal{E}_{\mathsf{k1}}\text{-Sf}(M_j) = \sigma_j$, which implies $m_j \neq M_j$. This concludes our proof, since the event 4 is $m_j = M_j$.

C Collision Probabilities

C.1 OMAC Collision Probabilities

Here, we state two Results proved in [28] about collisions in OMAC.

Result C.1 [$\mathsf{Pr[Col]_2}$]
Let E_{k} be a block cipher of size l and let OMAC[E_{k}] be its associated OMAC scheme. For the sake of simplicity, we only consider messages M whose length is a multiple of l (i.e. $|M|/l$ is an integer). Given a message M, we denote by μ the number of its blocks, namely $\mu = |M|/l$. Consider two messages M and M', then the following relation characterizes the probability of the OMAC collision:

$$\Pr[\mathsf{Col}_2] = \Pr[\mathsf{Col}(M, M')] \leq \frac{(\mu + \mu')^2}{2^l}$$

Result C.2 [$\mathsf{Pr}[\mathsf{Col}]_{\mathsf{q}}$]
Let E_{k} be a block cipher of size l and let $\mathrm{OMAC}[E_{\mathsf{k}}]$ be its associated OMAC scheme. For the sake of simplicity, we only consider messages M whose length is a multiple of l (i.e. $|M|/l$ is an integer). Given a message M, we denote by μ the number of its blocks, namely $\mu = |M|/l$. Given a list $\mathcal{Q}$ of q messages, the following relation characterizes the probability of the OMAC collision on $\mathcal{Q}$:

$$\Pr[\mathsf{Col}_q] = \Pr[\mathsf{Col}(\mathcal{Q})] \leq \frac{\left(\sum_{i=1}^{q} \mu_i\right)^2}{2^l}$$

C.2 Proof of Lemma 5.1

We need to compute both $\mathsf{Pr}[\mathsf{Col}_{\mathsf{q}}]$ and $\mathsf{Pr}[\mathsf{q\text{-}Col}]$. The case of $\mathsf{Pr}[\mathsf{Col}_{\mathsf{q}}]$ is easy and it can be immediately obtained from Result C.2. Concerning the case $\mathsf{Pr}[\mathsf{q\text{-}Col}]$, it can be calculated from Result C.1. Indeed, given a message M and a list S, $\mathsf{Pr}[\mathsf{Col}_{\mathsf{q}}]$ can be expressed as the sum of the collision probabilities $\mathsf{Pr}[\mathsf{Col}^{\mathsf{i}}_{2}]$ between M and a message M_i for all $M_i \in \mathsf{S}$. Here, $M_{\max}$ denotes any message of maximum length and $\mu_{\max}$ denotes its number of blocks. Therefore, we have

$$\mathsf{Pr}[\mathsf{q\text{-}Col}] \leq \sum_{i=1}^{q} \mathsf{Pr}[\mathsf{Col}(m_{\max}, m_i)] \leq \sum_{i=1}^{q} \frac{(\mu_{\max} + \mu_i)^2}{2^n}$$

From all the relations above, we have

$$\begin{aligned}
\epsilon &= \frac{q^2}{2^n} + \frac{1}{2^{n/2}} + \mathsf{Pr}[\mathsf{Col}_{\mathsf{q}}] + \mathsf{Pr}[\mathsf{q\text{-}Col}] \\
&\leq \frac{1}{2^{n/2}} + \frac{q^2}{2^n} + \frac{\left(\sum_{i=1}^{q} \mu_i\right)^2}{2^n} + \frac{\sum_{i=1}^{q} (\mu_{\max} + \mu_i)^2}{2^n} \\
&\leq \frac{1}{2^{n/2}} + \frac{q^2}{2^n} + \frac{\left(\sum_{i=1}^{q} \mu_i\right)^2}{2^n} + \frac{3q\mu_{\max}^2 + \sum_{i=1}^{q} \mu_i^2}{2^n} \\
&\leq \frac{1}{2^{n/2}} + \frac{q\left(q + (2\mu_{\max})^2\right) + \left(\sum_{i=1}^{q} \mu_i\right)^2 + \sum_{i=1}^{q} \mu_i^2}{2^n}
\end{aligned}$$

which is asymptotically negligible.