Private-key Symbolic Encryption Schemes

Naveed Ahmed[1] Christian D. Jensen[1] Erik Zenner[2]

[1]DTU-Informatics, [2]DTU-Mathematics
Technical University of Denmark, Copenhagen
[1]{naah,Christian.Jensen}@imm.dtu.dk
[2]E.Zenner@mat.dtu.dk

Abstract. Symbolic encryption, in the style of Dolev-Yao models, is ubiquitous in formal security analysis aiming at the automated verification of network protocols. The naïve use of symbolic encryption, however, may unnecessarily require an expensive construction: an arbitrary-length encryption scheme that is private and non-malleable in an adaptive CCA-CPA setting. Most of the time, such assumptions remain hidden and rather symbolic encryption is instantiated with a seemingly "good" cryptographic encryption, such as AES in the CBC configuration.
As an illustration of this problem, we first report new attacks on ECB and CBC based implementations of the well-known Needham-Schroeder and Denning-Sacco protocols. We then present a few symbolic encryption schemes along with their cryptographic semantics, and prove the hierarchical relations between the proposed schemes from both cryptographic and formal perspectives. These symbolic schemes can be seamlessly used in many existing formal security models.

Keywords: Encryption, Hidden Assumptions, Formal Security Model

Common Terminology

A protocol message is denoted by a list without square brackets, so, $M_1, M_2 \neq M_2, M_1$. Two messages (lists) can be concatenated using a *comma*. Encryption of a message M_1, M_2 using a key K_{AB} shared between A and B is denoted by $\{M_1, M_2\}_{K_{AB}}$ or $\{M_1, M_2\}_{AB}$. A *hat* on the top of a symbol represents its binary encoding, e.g., the symbol M_1 in a formal model is encoded as a bit string $\widehat{M_1}$ of length $|\widehat{M_1}|$ in a cryptographic model. An adversary (or his strategy) is denoted by $\mathcal{I}$ (intruder). Exclusive-or ($\oplus$) is abbreviated as Xor in the text. We distinguish between *symbolic encryption*, which is used in formal security models [16, 9], and *cryptographic encryption* as used in the traditional complexity-theoretic cryptography [29, 30]. A *random string* $v \in \{0,1\}^s$ refers to a value of a random variable U_s^v on the uniform distribution of size s.

1 Introduction

A private-key encryption scheme enables two honest parties that share a key to privately communicate over a network, in such a way that a dishonest man-in-middle, the adversary, is unable to gain any non-trivial information about

the communication. The standard security notions for cryptographic encryption include left-right indistinguishability (IND) and non-malleability (NM), which can be characterized in different attack settings [30].

In cryptography, the primitive objects on which functions operate and cryptographic schemes are built are binary strings. The question whether certain security properties hold for a cryptographic scheme is usually answered by a polynomial-time reduction of the scheme to generally accepted cryptographic assumptions, such as the existence of a pseudorandom-number generator. A (uniform) adversary in this model is assumed to be able to compute any feasible (BPP) function. At the protocol level, the simulation paradigm [33] is more common; in which one constructs an ideal model using an idealized functionality with an aim to capture the non-security requirements of a protocol; then, the security of the protocol is proved by showing the equivalence between the ideal model and the actual protocol model.

Most of the cryptographic analysis is done by hand, and the support of automation [35] is quite limited. Unlike cryptographic schemes, the model of any reasonably sized protocol is often complex. Due to the painstaking work involved in concrete security analysis, only a handful of network protocols have been analyzed, e.g., only a small fraction of roughly 200 protocols listed in 2003 [27] are accompanied by such security analysis. Even so, history shows that many incorrect "proofs of security", although done with honest intentions, find their way into the literature, and it often takes many years to locate the errors [20].

Over the years, many abstractions of cryptography have been proposed to enable automated security analysis. The most popular abstractions are in the forms of Dolev-Yao model [6] and its derivatives [15]. In these formal models, two types of simplifications are introduced. Firstly, binary strings and functions are replaced by symbolic terms and derivation rules. In particular, this results in idealized encryption functions—either an adversary can decrypt a symbolic ciphertext (e.g., if he can derive the key) or the adversary gets absolutely no information about the plaintext (cf. indistinguishabilty goal [29]).

The second simplification is related to the capabilities of an adversary, namely the adversary is modeled as a non-deterministic strategy that is limited to selecting its actions from a small set of (pre-defined) logic rules. This can be compared to a computational adversary which is assumed to be able to compute any polynomial-time function besides being able to access certain oracles, e.g., a decryption oracle in a chosen ciphertext attack. The security models that use these two abstractions are commonly referred to as symbolic/formal security models.

Needless to say, a symbolic model is usually simpler than its cryptographic counterpart and provides a type of handle to a system designer who may not be an expert in computational cryptography. A security proof in a symbolic model is relatively short, because security issues in a symbolic model are decision problems, with a yes/no answer compared to probabilistic results of cryptographic analysis. More importantly, computers can do the tedious job of proving (and similarly verifying) the proofs of security. An impressive number of protocols of

practical relevance have been analyzed in this line of work (Lowe's work [14] is a classic example).

The simplicity of the symbolic models, however, comes at a price: any security assurance in a symbolic model does not automatically translate to the underlying computational cryptography. This is due to a huge gap between cryptographic assumptions and the assumptions behind the symbolic abstractions. Not long ago, many research efforts spurred to address this obvious gap between symbolic and computational cryptography, most notably, Abadi and Rogaway [5], and Backes, et al. [11] independently published interesting initial results.

In our work, we address one particular aspect of this problem, namely the safe relaxation of some of the assumptions behind the abstraction of symbolic encryption. In any implementation of symbolic encryption, one has to make certain security critical decisions, e.g., mode of encryption, block alignment, and message authentication. Many attacks targeting the implementation of symbolic encryption are known [17, 19, 18]. One brute-force approach to address such issues is to rely on the most stringent cryptographic interpretation [30]—use an encryption scheme that is private and non-malleable against an adversary that has adaptive access to encryption and decryption oracles. Another approach is to use symbolic cryptography in such a way that it can be instantiated with a relaxed variant of cryptographic encryption; our work pertains to this second approach.

In this paper, we first present new attacks on the ECB and CBC implementations of the Needham-Schroeder symmetric-key (NSSK) protocol [2]. Although cryptographically such implementations are not considered strong, they are not artificial nevertheless, and in fact ECB and CBC do not violate the assumed properties of encryption function by their authors. The NSSK protocol can be broken in another way if an adversary has access to an old key and the relevant encrypted message [4], however, our attack does not rely on any of these assumptions. These attacks also work with the seven-round version of the NSSK protocol [3], which is an improved version of the original NSSK protocol after the flaw [4] was discovered. Further, we report new attacks on two implementations of the Denning-Sacco symmetric-key (DSSK) protocol [4], which is another improved version of the NSSK protocol, and which does not suffer any attacks to the best of our knowledge.

To prevent the type of vulnerabilities exploited by the reported attacks, we advocate better ways of using symbolic encryption function, by proposing four symbolic encryption schemes. These schemes have natural correspondence to standard cryptographic constructions. Many state of the art protocol analysis tools (e.g., OFMC [16], LYSA [9]) can accept these refined specifications. In practice, the proposed schemes correspond to different resource requirements, and therefore a level of safe optimization can be achieved while still working at the abstract level.

The rest of the paper is arranged as follows. In § 2, we briefly examine the prior art. Next, in § 3 and § 4, we present new attacks, which serve as a motivation for improving the way symbolic encryption is used. In § 5, we present

a few symbolic encryption schemes and prove their cryptographic separation, and then in § 6 we show that these schemes also provide different levels of security in a formal security model. In § 7, we discuss our contribution in a broader perspective, and finally, in § 8, we conclude our work.

2 Related Work

Dolev-Yao style symbolic cryptography [6] is the basis of most formal security models, e.g., BAN logic, process calculi [8, 9] and model-checking [16]; Meadows presents an extensive survey [15]. We here do not discuss symbolic security analysis as such and only focus on the cryptographic perspective of symbolic encryption.

Moore [22] was probably the first to highlight the security problems that may occur in implementing symbolic encryption, in particular, in the use of DES for providing message authentication. Boyd [17] describes a few attacks on the NSSK protocol that exploit the implementation of the encryption function, however, the presentation does not come close to the attacks described in this paper. In fact, the adversary gain, in our case, is higher than the previously known attacks [4, 17]. Mao and Boyd [19] discuss some general vulnerabilities that may occur when using cipher-block-chaining mode for implementing encryption. Bellovin [28] reported vulnerabilities in the earlier versions of IP-sec by exploiting CBC-mode encryption.

Stubblebine et al. [18] investigates modes of encryption for discovering *known pairs* and *chosen texts*, using the NRL protocol analyzer. Our attack makes use of chosen texts, in which a party can be used as an encryption oracle; this is then exploited by an adversary who obtains the ciphertext against a plaintext. In the same line of work, Kremer and Ryan [21] model ECB and CBC mode using Blanchet's protocol verifier. Interestingly, they use the NSSK protocol as a case study but stop after indicating the existence of chosen texts in the protocol. Nevertheless, the existence of chosen texts is quite common in cryptographic protocols and often does not lead to insecure encryption.

One of the most interesting cases is that of encryption-only-mode of IP-sec, for which Paterson and Yau [23] exploited CBC mode of encryption. Their attacks work if an implementation does not follow the standard strictly, however, many implementation, including that of Linux, falls in this category. Later, Degabriele and Paterson [24] published another attack that works only if an implementation strictly follows the standard.

Chevalier et al. [7] extend the Dolev-Yao intruder with the capability to exploit Xor operator, as used in CBC, and they show that the protocol insecurity problem is NP-complete. Küsters and Truderung developed a verification method that can reduce the protocol models that are *Xor-linear* to Xor free models, which then can be analyzed using existing tools [25]; however, the CBC based NSSK protocol is not Xor-linear due to the nested encryption.

In a slightly bigger picture—for establishing a theoretically sound link between symbolic cryptography and complexity-theoretic cryptography—there is

an impressive amount of research during the last decade; we only briefly summarize the most notable work in the following. In our view, currently these approaches are more focused on achieving theoretical soundness. Another recent trend, which we do not consider here, is to develop the tool support in a semi-automated manner for cryptographic proofs, such as in EasyCrypt [35]; the state of the art in this direction is in a very primitive stage.

Abadi and Rogaway [5] show that indistinguishablity between the ensembles of cryptographic encryption can be translated to the symbolic form in Dolev-Yao model, in such a way that verification of a symbolic property corresponds to a security guarantee in the computational world (with a high probability). Independent from the Abadi-Rogaway work, Backes, Pfitzmann and Waidner [11] aim at establishing soundness of symbolic cryptography in a general cryptographic model using a general composition theorem.

Micciancio and Warinschi [10] extended the Abadi-Rogaway approach [5] to interactive protocols and mutual authentication in presence of active adversaries. Herzog et al. [13] put forward a new definition of plain-text aware encryption, which can be used for the secure realization of symbolic public-key encryption; this notion is too strong however. In the line of universal composability, Ran Canetti and Herzog [12] show that the Dolev-Yao model can be layered on top of the traditional universal composability framework. Currently, this approach is limited to so-called simple protocols: the protocols that use only those cryptographic schemes that have some standard symbolic counterparts.

This paper is within the scope of a larger effort for improving the soundness of symbolic cryptography, yet the presented work is very much practice-oriented, namely improving the way encryption is modeled to get the results that are closer to traditional cryptographic encryption schemes. On the one hand, the new attacks on old protocols (more than 30 years) certainly serve as a word of caution for a developer who implements a formally verified protocol. On the other hand, these attacks will motivate a security analyst working in the community of formal methods to use the proposed schemes of encryption.

3 Attacks on Implementations of NSSK protocol

The NSSK protocol [2] is a key establishment protocol, based on symmetric encryption and the notion of a trusted third-party (TTP). The protocol was proposed in 1978, at the very start of the era of modern cryptography. It is considered as one of the benchmarks for protocol analysis tools because it contains a subtle flaw: if an adversary gets hold of an old session key then he can masquerade as another entity [4]. In this paper, the reader may assume that when a session expires then the session key is safely discarded. The protocol narrations are listed in the following.

(1) $A \longrightarrow S : A, B, N_A$
(2) $S \longrightarrow A : \{N_A, B, K_{AB}, \{K_{AB}, A\}_{SB}\}_{SA}$
(3) $A \longrightarrow B : \{K_{AB}, A\}_{SB}$
(4) $B \longrightarrow A : \{N_B\}_{AB}$
(5) $A \longrightarrow B : \{N_B - 1\}_{AB}$

Here A and B represent the initiator and the responder role that parties can take during an execution of the protocol; S is the role of the trusted third-party (TTP). It is assumed that S knows the identities of all legitimate entities (principals), and shares a long-term secret key with each of them, namely, S shares K_{SA} and K_{SB} with A and B respectively. The term K_{AB} denotes a session key.

The first message is a request from A to the TTP that A wishes to establish a key with B, by sending its identity, the identity of the peer entity and a nonce. On receiving the request the TTP generates a random session key K_{AB}. The TTP replies with a message encrypted with A's long-term key. Most importantly, this message includes the session key K_{AB}, and another encrypted message containing the same session key but encrypted with B's long term key, which A sends to B in the next step.

When B receives the message, it decrypts it using K_{SB}, then verifies that it contains B's identity, and if successful, then B considers K_{AB} as a valid session key. To verify the freshness of the session key, B sends a nonce, N_B, to A encrypted using the session key. On receiving the message in Step 4, A decrypts it and sends $N_B - 1$ to B encrypted using the same session key. This completes the protocol. If both parties terminate without generating any error then A and B assume that K_{AB} is a valid session key for the subsequent communication.

In the following we describe two version of the attack against two different implementations of encryption function. These attacks are also applicable on the seven-round version of the NSSK protocol [3], which does not suffer from the old-session-key attack [4].

ECB-version

The electronic code book (ECB) mode of encryption is the most obvious and inexpensive way of applying a block cipher to a plaintext that is longer than its block length. It does not provide the strong version of privacy, namely the semantic security [29], but in many places semantic security is not strictly required for the protocol security (see § 6).

For the simplicity of exposition, we assume that each term of the protocol is encoded in a separate block (e.g., $\{N_1, N_2\}_{AB} = \{N_1\}_{AB}, \{N_2\}_{AB}$), and the attacker is an insider, i.e., $\mathcal{I}$ is a legitimate network entity and shared $K_{S\mathcal{I}}$ with the TTP. An attacker $\mathcal{I}$ in the role of A is denoted by $\mathcal{I}(A)$. If blocks are not encoded with this perfect alignment then a less efficient version of the attack may exist that requires more computation and communication on the part of the adversary[1]. The attack is listed in Fig. 1.

The attack consists of three setup-phases, in which $\mathcal{I}$ obtains certain terms that are used in the actual attack. As shown, $\mathcal{I}$ uses S as an oracle. First, he obtains the term $\{K_1\}_{SB}$ in Setup-(a). Then, he uses this term as a nonce in Setup-(b), to obtain the term $\{\{K_1\}_{SB}\}_{SA}$. In Setup-(c), he uses K_1, whose value he knows from Setup-(a), as a nonce to obtain $\{K_1\}_{SA}$. As shown in the

[1] E.g., see the attack on IP-sec [24], which succeeds after a large amount of queries.

Steps	Messages
Setup-(a)	
(1)	$\mathcal{I} \longrightarrow S : \mathcal{I}, B, N_{\mathcal{I}}$
(2)	$S \longrightarrow \mathcal{I} : \{N_{\mathcal{I}}\}_{S\mathcal{I}}, \{B\}_{S\mathcal{I}}, \{K_1\}_{S\mathcal{I}}, \{\{K_1\}_{SB}\}_{S\mathcal{I}}, \{\{\mathcal{I}\}_{SB}\}_{S\mathcal{I}}$
Setup-(b)	
(1)	$\mathcal{I}(A) \longrightarrow S : A, B, \{K_1\}_{SB}$
(2)	$S \longrightarrow \mathcal{I}(A) : \{\{K_1\}_{SB}\}_{SA}, \{B\}_{SA}, \{K_2\}_{SA}, \{\{K_2\}_{SB}\}_{SA}, \{\{A\}_{SB}\}_{SA}$
Setup-(c)	
(1)	$\mathcal{I}(A) \longrightarrow S : A, B, K_1$
(2)	$S \longrightarrow \mathcal{I}(A) : \{K_1\}_{SA}, \{B\}_{SA}, \{K_3\}_{SA}, \{\{K_3\}_{SB}\}_{SA}, \{\{A\}_{SB}\}_{SA}$
Attack	
(1)	$A \longrightarrow S : A, B, N_A$
(2a)	$S \longrightarrow \mathcal{I}(A) : \{N_A\}_{SA}, \{B\}_{SA}, \{K_4\}_{SA}, \{\{K_4\}_{SB}\}_{SA}, \{\{A\}_{SB}\}_{SA}$
(2b)	$\mathcal{I}(S) \longrightarrow A : \{N_A\}_{SA}, \{B\}_{SA}, \{K_1\}_{SA}, \{\{K_1\}_{SB}\}_{SA}, \{\{A\}_{SB}\}_{SA}$
(3)	$A \longrightarrow B : \{K_1\}_{SB}, \{A\}_{SB}$
(4)	$B \longrightarrow A : \{N_B\}_{K_1}$
(5)	$A \longrightarrow B : \{N_B - 1\}_{K_1}$

Fig. 1. Attack on ECB-version of NSSK protocol

Attack, these two terms, $\{\{K_1\}_{SB}\}_{SA}$ and $\{K_1\}_{SA}$, are enough to deceive A and B, in accepting K_1 as a new session key.

At the end of the attack, the adversary gain is the compromised key K_1 as a new session key. Thus, adversary can simply play the man-in-middle role to listen to all subsequent traffic of the session. Moreover, at any time, he can masquerade as A to B or masquerade as B to A, to send and receive legitimate requests for data. In this attack, the new session key is a server generated key, which can also be generated by the adversary locally.

CBC-version

The same attack can be extended, although not trivially, to the case where encryption is implemented using CBC mode of operation, which provides semantic security against CPA. Note that this privacy guarantee is valid independent of the block alignments in a plaintext. Our CPA limited adversary, therefore, manifests that NSSK protocol requires more than semantic security from its encryption function.

As per the standard cryptographic assumption, the initializing vectors (iv) in CBC mode are public values; the superscripts in iv^o, iv^* and iv^x are labels used to easily distinguish between initialization vectors in Setup-(a), Setup-(b) and Setup-(c) respectively; a subscript, such as '1' in iv_1, is used to distinguish different values of initialization vectors. The notation '=' is used to introduce intermediate terms to simplify the description of the attack.

In Setup-(a), $\mathcal{I}$ obtains the term $\{iv_2^o \oplus K_1\}_{SB}$, which he sends as a nonce in Setup-(b) to obtain c_1^*. In Setup-(c), $\mathcal{I}$ obtains the term c_1^x by sending K_1 as a

nonce. In the main phase of the attack, $\mathcal{I}$ replays these two terms, c_1^* and c_1^x, in step (2b). Later in step (3b), $\mathcal{I}$ replays $\{iv_2^o \oplus K_1\}_{SB}$, so that B believes in K_1 as a new session key shared with a party whose identity is $c_6 \oplus iv_2^o \oplus K_1$. At this stage, $\mathcal{I}$ has successfully deceived both A and B into accepting the session keys that he knows. There are two different session keys, namely, A's session key is $c_2 \oplus iv_1^x \oplus K_1$ and B's session key is K_1. Since both of these keys are known to $\mathcal{I}$, he can play a man-in-middle role in any subsequent communication, in the same style as he plays man-in-middle in the steps (3a)-(5b).

Steps	Messages
Setup-(a)	
(1)	$\mathcal{I} \longrightarrow S : \mathcal{I}, B, N_{\mathcal{I}}$
(2)	$S \longrightarrow \mathcal{I} : iv_1^o, iv_2^o, c_1^o = \{iv_1^o \oplus N_{\mathcal{I}}\}_{S\mathcal{I}}, c_2^o = \{c_1^o \oplus B\}_{S\mathcal{I}}, c_3^o = \{c_2^o \oplus K_1\}_{S\mathcal{I}}, c_4^o = \{c_3^o \oplus \{iv_2^o \oplus K_1\}_{SB}\}_{S\mathcal{I}}, c_5^o = \{c_4^o \oplus \{\{iv_2^o \oplus K_1\}_{SB} \oplus \mathcal{I}\}_{SB}\}_{S\mathcal{I}}$
Setup-(b)	
(1)	$\mathcal{I}(A) \longrightarrow S : A, B, \{iv_2^o \oplus K_1\}_{SB}$
(2)	$S \longrightarrow \mathcal{I}(A) : iv_1^*, iv_2^*, c_1^* = \{iv_1^* \oplus \{iv_2^o \oplus K_1\}_{SB}\}_{SA}, c_2^* = \{c_1^* \oplus B\}_{SA}, c_3^* = \{c_2^* \oplus K_2\}_{SA}, c_4^* = \{c_3^* \oplus \{iv_2^* \oplus K_2\}_{SB}\}_{SA}, c_5^* = \{c_4^* \oplus \{\{iv_2^* \oplus K_2\}_{SB} \oplus A\}_{SB}\}_{SA}$
Setup-(c)	
(1)	$\mathcal{I}(A) \longrightarrow S : A, B, K_1$
(2)	$S \longrightarrow A : iv_1^x, iv_2^x, c_1^x = \{iv_1^x \oplus K_1\}_{SA}, c_2^x = \{c_1^x \oplus B\}_{SA}, c_3^x = \{c_2^x \oplus K_3\}_{SA}, c_4^x = \{c_3^x \oplus \{iv_2^x \oplus K_3\}_{SB}\}_{SA}, c_5^x = \{c_4^x \oplus \{\{iv_2^x \oplus K_3\}_{SB} \oplus A\}_{SB}\}_{SA}$
Attack	
(1)	$A \longrightarrow S : A, B, N_A$
(2a)	$S \longrightarrow \mathcal{I}(A) : iv_1, iv_2, c_1 = \{iv_1 \oplus N_A\}_{SA}, c_2 = \{c_1 \oplus B\}_{SA}, c_3 = \{c_2 \oplus K_4\}_{SA}, c_4 = \{c_3 \oplus \{iv_2 \oplus K_4\}_{SB}\}_{SA}, c_5 = \{c_4 \oplus \{\{iv_2 \oplus K_4\}_{SB} \oplus A\}_{SB}\}_{SA}$
(2b)	$\mathcal{I}(S) \longrightarrow A : iv_1, iv_2, c_1 = \{iv_1 \oplus N_A\}_{SA}, c_2 = \{c_1 \oplus B\}_{SA}, c_1^x = \{iv_1^x \oplus K_1\}_{SA}, c_1^* = \{iv_1^* \oplus \{iv_2^o \oplus K_1\}_{SB}\}_{SA}, c_1^* = \{iv_1^* \oplus \{iv_2^o \oplus K_1\}_{SB}\}_{SA}$
(3a)	$A \longrightarrow \mathcal{I}(B) : c_1^x \oplus iv_1^* \oplus \{iv_2^o \oplus K_1\}_{SB}, c_1^* \oplus iv_1^* \oplus \{iv_2^o \oplus K_1\}_{SB}$
(3b)	$\mathcal{I}(A' = c_6 \oplus iv_2^o \oplus K_1) \longrightarrow B : iv_2^o, c_6 = \{iv_2^o \oplus K_1\}_{SB}, \{iv_2^o \oplus K_1\}_{SB}$
(4a)	$B \longrightarrow \mathcal{I}(A') : \{N_B\}_{K_1}$
(4b)	$\mathcal{I}(B) \longrightarrow A : \{N_B\}_{c_2 \oplus iv_1^x \oplus K_1}$
(5a)	$A \longrightarrow \mathcal{I}(B) : \{N_B - 1\}_{c_2 \oplus iv_1^x \oplus K_1}$
(5b)	$\mathcal{I}(A') \longrightarrow B : \{N_B - 1\}_{K_1}$

Fig. 2. Attack on CBC-version of NSSK Protocol

4 Attacks on Implementations of DSSK protocol

As mentioned earlier, Denning and Sacco [4] spotted the original problem in the NSSK protocol. In the same paper, they improved the protocol using time-stamps; the modified protocol is as follows.

(1) $A \longrightarrow S : A, B$
(2) $S \longrightarrow A : \{B, K_{AB}, T, \{A, K_{AB}, T\}_{SB}\}_{SA}$
(3) $A \longrightarrow B : \{A, K_{AB}, T\}_{SB}$

The protocols works essentially in the same way as the NSSK protocol; the new term T represents a time stamp, and obviously the assumption is that there is a loosely synchronized clock in the network. Chevalier and Vigneron [26] reported a possible type flaw in the Denning-Sacco (DSSK) protocol based on a somewhat dubious assumption: if $\{T\} \equiv \{T, \{B, K_{AB}, T\}_{SA}\}$. Even if this assumption holds then the only gain of the adversary is to convince B, for a moment, that it is sharing a key with A, but, after that, no further communication is possible, because neither A nor the attacker knows the key. Other than that, to the best of our knowledge, there are no known attacks against this protocol.

ECB-version

This attack is very similar to the ECB-version of the NSSK protocol and is listed in Appendix A.

CBC-version

Steps	Messages
Setup	
(1)	$\mathcal{I} \longrightarrow S : \mathcal{I}, A$
(2)	$S \longrightarrow \mathcal{I} : iv_1^o, iv_2^o, c_1^o = \{A \oplus iv_1^o\}_{S\mathcal{I}}, c_2^o = \{K_1 \oplus c_1^o\}_{S\mathcal{I}}, c_3^o = \{T_1 \oplus c_2^o\}_{S\mathcal{I}}, c_4^o = \{(\bar{c}_4^o = \{\mathcal{I} \oplus iv_2^o\}_{SA}) \oplus c_3^o\}_{S\mathcal{I}}, c_5^o = \{(\bar{c}_5^o = \{K_1 \oplus \bar{c}_4^o\}_{SA}) \oplus c_4^o\}_{S\mathcal{I}}, c_6^o = \{(\bar{c}_6^o = \{T_1 \oplus \bar{c}_5^o\}_{SA}) \oplus c_5^o\}_{S\mathcal{I}}$
Attack	
(1)	$A \longrightarrow S : A, B$
(2a)	$S \longrightarrow \mathcal{I}(A) : iv_1, iv_2, c_1 = \{B \oplus iv_1\}_{SA}, c_2 = \{K_2 \oplus c_1\}_{SA}, c_3 = \{T_2 \oplus c_2\}_{SA}, c_4 = \{\{A \oplus iv_2\}_{SB} \oplus c_3\}_{SA}, c_5 = \{\{K_2 \oplus \{A \oplus iv_2\}_{SB}\}_{SB} \oplus c_4\}_{SA}, c_6 = \{\{T_2 \oplus \{K_2 \oplus \{A \oplus iv_2\}_{SB}\}_{SB}\}_{SB} \oplus c_5\}_{SA}$
(2b)	$\mathcal{I}(S) \longrightarrow A : iv_1, iv_2, c_1 = \{B \oplus iv_1\}_{SA}, \bar{c}_5^o, \bar{c}_6^o, c_4, c_5, c_6$
(3)	$A \longrightarrow \mathcal{I}(B)$: random data

Fig. 3. Attack on CBC-version of DSSK Protocol

This attack is listed in Fig 3 and is different from the CBC-version of the NSSK protocol. Here, the adversary succeeds in impersonating B to A, i.e, at the end of the attack I in the role of B has a shared key with A. In the setup phase, $\mathcal{I}$ sends a request to S for establishing a connection with A, and as a result, $\mathcal{I}$ receives $\bar{c}_5^o$ and $\bar{c}_6^o$ that are later replayed in the actual attack.

In the main phase of the attack, $\mathcal{I}$ intercepts the reply from S and replace c_2 and c_3 with $\bar{c}_5^o$ and $\bar{c}_6^o$ respectively. Consequently, the last three messages will

decrypt to some random data when A later sends them to B, however, $\mathcal{I}$ can pretend to be B. The session key for A and $\mathcal{I}(B)$ is $K_1 \oplus \bar{c}_4^o \oplus c_1$. Clearly, this term is computable by $\mathcal{I}$ because K_1, $\bar{c}_4^o$ and c_1 are known to $\mathcal{I}$.

The term $\bar{c}_6^o$ is decrypted to T_1. The Setup phase of the attack needs to be in real-time (in a loose sense) so that the difference between T_1 and T_2 is tolerable. As per the authors of the protocol, the definition of real-time is quite relaxed, namely a delay up to $\triangle t_1 + \triangle t_2$ is tolerable, where $\triangle t_1$ is the interval representing normal time-shift between A's local clock and the server clock, and $\triangle t_2$ is the expected network delay. This value is typically equal to a few seconds for most of the networks, such as the Internet.

5 Symbolic Encryption Schemes

None of the above attacks manifest themselves in a formal security model if symbolic encryption is naïvely used by specifying the encryption as one monolithic ciphertext, with an implicit assumption that the encryption will be implemented with an "appropriate" cryptographic encryption scheme. Historically, this overly strong assumption was introduced to simplify the formal model in order to avoid the state explosion problem. Recent advances in model-checking and static analysis, however, make it practical to weaken this strong assumption; in particular, symbolic encryption deserves to be specified using the abstraction of a block-cipher, *finite pseudo random function* (PRF) [31].
We start with the definition of a minimal symbolic encryption system, to present our claims in a simple but precise manner.

Definition 1 (Symbolic Encryption System). *On the set of all base terms $\mathcal{V}$, with a security parameter $s = log_2(|\mathcal{V}|)$, we define a private-key symbolic encryption system $\mathcal{SE}$ as follows.*

- $\mathtt{M} ::= \mathtt{M}, \mathtt{M} \mid \mathtt{V} \mid \{\mathtt{M}\}_K \mid \{\mathtt{C}\}_K^{-1} \mid \phi$
- $\mathtt{V} ::= x \in \mathcal{V}$
- $\mathtt{K} ::= \mathtt{M}$ *(Syntactic sugar to indicate that the term K is being used as a key)*
- $\mathtt{C} ::= \{\mathtt{M}\}_K$ *(Syntactic sugar to indicate that the term is a ciphertext)*
- *Cancellation Rule :* $M = \{\{M\}_K\}_K^{-1} = \{\{M\}_K^{-1}\}_K$
- *Encryption Rule : If an agent knows K and M then the agent knows* $\{M\}_K$.
- *Decryption Rule : If an agent knows K and C then the agent knows* $\{C\}_K^{-1}$.

Here $\mathtt{M}$, $\mathtt{K}$, $\mathtt{C}$ and $\mathtt{V}$ are the formal expression; while M, K and C are the corresponding meta-variables.

Note that we do not include the Xor operator in $\mathcal{SE}$, which although is required by many standard cryptographic schemes (such as CBC). This exclusion is because properly incorporating the Xor operator in formal security analysis is a long-standing open problem [7, 25], e.g., Xor is not supported by OFMC [16], LySa [9], and Spi-calculus [8]. The essence of this paper is about the "smart" use of symbolic encryption from a cryptographic perspective, while working within the scope of existing theories of formal security models and without proposing any extension in the models themselves.

Definition 2 (Cryptographic Semantics). *The cryptographic semantics of the symbolic encryption system in Def. 1 are as follows.*

- $V \stackrel{def}{=} \widehat{V} \stackrel{rnd}{\in} \{0,1\}^s$ *(Each base term is encoded as a uniformly distributed random bit string of a fixed length* $s = |\mathcal{V}|$*)*
- $M_1, M_2 \stackrel{def}{=} \widehat{M_1}, \widehat{M_2}$ *(Concatenation of two bit strings)*
- $\{M\}_K \stackrel{def}{=}$ *UV-PRF*$_{\widehat{K}}(\widehat{M})$ *(also denoted by* $\{\widehat{M}\}_{\widehat{K}}$ *for brevity)*
  Here UV-PRF$_{\widehat{K}}(\widehat{M})$ *is the length-preserving* $\widehat{K}$*th variadic pseudorandom function in a family of UV-PRF.*
- $\{C\}_K^{-1} \stackrel{def}{=}$ *If* C *is in the output table of UV-PRF*$_{\widehat{K}}(\widehat{M})$ *then output corresponding* M*; otherwise, select a random* M*, s.t.,* $|M| = |C|$ *and* M *is not in the output table, and write* (M, C) *pair in the output table of UV-PRF*$_{\widehat{K}}(\widehat{M})$*.*

The prefix UV in the idealized notion UV-PRF denotes an Unbounded family of Variadic PRFs. The term unbounded implies that the family contains an infinite number of functions, which are addressable by the arbitrary size key K. A variadic PRF [1] can take input $\widehat{M}$ of an arbitrary size, as in our case M may consist of potentially infinite number of base terms. The notion of UV-PRF can be compared to the classic notion of a PRF family [31, 32], in which the input domain is finite and the individual functions are addressable by a fixed size key.

Definition 3 (Security). *Let* $\widehat{K}$ *to be a secret. We define the following three security properties for* $\mathcal{SE}$ *in uniform complexity.*

WP-security (Weak Privacy Against Passive Attack) $\stackrel{def}{=}$ *It is infeasible for an adversary* $\mathcal{I}$ *to compute* $\widehat{C}$ *for a known* $\widehat{M}$*, s.t.,* $C = \{M\}_K$*. Further, it is also infeasible for* $\mathcal{I}$ *to compute* $\widehat{M}$ *for a known* $\widehat{C}$*, s.t.,* $M = \{C\}_K^{-1}$*.*

NM-security (Non-malleability Against Chosen Plaintext/Ciphertext Attack) $\stackrel{def}{=}$ *It is infeasible for* $\mathcal{I}$ *to compute* $\widehat{C'}$ *for a known* $\widehat{C}$*, s.t., a symbolic relation* $\mathcal{R}(M, M')$ *holds, where* $M = \{C\}_K^{-1}$ *and* $M' = \{C'\}_K^{-1}$*.*

IND-security (Indistinguishability Against Adaptive Chosen Plaintext Attack) $\stackrel{def}{=}$ *It is infeasible for* $\mathcal{I}$ *to distinguish the two probability distributions: PDF*$(\{\widehat{M}\}_{\widehat{K}})$ *and PDF*$(\{\widehat{M'}\}_{\widehat{K}})$*, where PDF*$(.)$ *is the discrete probability distribution function. Alternatively,* $\mathcal{I}$ *can only succeeds in the indistinguishability experiment (IND-P2-C0) [30] with a negligible probability.*

Clearly, WP-security is implied by IND-security, because if an adversary can recover the plaintext from a ciphertext then he can always win in the indistinguishability experiment.

Proposition 1 (Soundness of $\mathcal{SE}$). *The symbolic encryption function in* $\mathcal{SE}$ *is WP,NM-secure.*

Proof (sketch). This proposition holds by the definition of UV-PRF, i.e., the mapping between $\widehat{M}$ and $\{\widehat{M}\}_{\widehat{K}}$ is random and $\widehat{K}$ is assigned from a uniform distribution. Encryption and decryption oracles can only help in an exhaustive search and passive adversary can only make a random guess for $\widehat{M}$. The formal proof is trivial (but tedious) and is left out. □

In the following, we introduce four symbolic encryption schemes.

Definition 4 (Symbolic Encryption Schemes). *Let $M_1, ..., M_i, ..., M_N$ be the parsing of a polynomial size plaintext M, such that each $|\widehat{M_i}| = s$. Assume the existence of a block-cipher (e.g., AES) with the block size s (say 128) and the key size $|\widehat{K}| \geq s$ (say 128 and 256). The following symbolic encryption schemes are defined in $\mathcal{SE}$.*

1. $\mathcal{SE}^{ecb}(M)$ *(ECB Symbolic Encryption):*
   $\stackrel{def}{=} \{M_1\}_K, ..., \{M_i\}_K, ..., \{M_N\}_K$
2. $\mathcal{SE}^{bk}(M)$ *(Bulk Symbolic Encryption):*
   $\stackrel{def}{=} \{M_1, ..., M_i, ..., M_N\}_K$
3. *The following three schemes are equivalent (see Lemma. 6).*
   (a) $\mathcal{SE}^{rn}(M)$ *(Randomized Symbolic Encryption):*
   $\stackrel{def}{=} V_1, \{M_1\}_{K,V_1}, ..., V_i, \{M_i\}_{K,V_i}, ..., V_N, \{M_N\}_{K,V_N}$
   (b) *(a)* $\mathcal{SE}^{cbc*}(M)$ *(Equivalent CBC Symbolic Encryption):*
   $\stackrel{def}{=} C_0, \{M_1\}_{K,C_0}, ..., \{M_i\}_{K,C_{i-1}}, ..., \{M_N\}_{K,C_{N-1}}$
   where $C_i = \{M_i\}_{K,C_{i-1}}$*, and* $C_0 = V_1$ *is a base term.*
   (c) $\mathcal{SE}^{cbc}(M)$ *(CBC Symbolic Encryption):*
   $\stackrel{def}{=} C_0, \{C_0 \oplus M_1\}_K, ..., \{C_{i-1} \oplus M_i\}_K, ..., \{C_{N-1} \oplus M_N\}_K$
   where $C_i = \{C_{i-1} \oplus M_i\}_K$*, and* $C_0 = V_1$ *is a base term.*
4. $\mathcal{SE}^{rnb}(M)$ *(Randomized Bulk Symbolic Encryption):*
   $\stackrel{def}{=} V, \{M_1, ..., M_i, ..., M_N\}_{K,V}$

Note that the terms V and V_i appear as free variables, therefore these variables are assumed to be instantiated with unique values in each instance of a protocol, such as done in formal analysis tools [16, 9]. Cryptographically, these free variables represent random variables on the uniform distribution of size s, as per Def. 2.

The main motivation for the above division is that we can directly write all of these specifications in many formal security models, where each of these specifications provides a different type of security guarantee in these formal models (§ 6). As the reader may have noted that $\mathcal{SE}^{cbc}$ is an extra-logical scheme, because it uses Xor operator and cannot be in $\mathcal{SE}$, but we later show that $\mathcal{SE}^{cbc}$ is equivalent to $\mathcal{SE}^{cbc*}$. In the following we present a few results regarding the correspondence between symbolic encryption and cryptographic encryption.

Corollary 1. *In $\mathcal{SE}^{ecb}$, $\mathcal{SE}^{rn}$, $\mathcal{SE}^{cbc*}$, and $\mathcal{SE}^{cbc}$, each symbolic encryption term represents a bounded family (i.e., a fixed number of elements) of finite PRFs [31, 32]. In $\mathcal{SE}^{bk}$ and $\mathcal{SE}^{rnb}$, each symbolic encryption term is a bounded family of variadic PRFs (with polynomial size input) [1].*

Proof. Each symbolic encryption expression ($\{\mathtt{M}\}_{\mathtt{K}}$) in $\mathcal{SE}$ is a UV-PRF. Further, in Def. 4, the key size $|K|$ is fixed, therefore, in all symbolic encryption schemes considered here, each encryption term represents a bounded family of PRFs. In $\mathcal{SE}^{ecb}$, $\mathcal{SE}^{rn}$, $\mathcal{SE}^{cbc*}$, and $\mathcal{SE}^{cbc}$, there is only one term of size s, therefore, these

schemes represent finite PRFs. In $\mathcal{SE}^{bk}$ and $\mathcal{SE}^{rnb}$, the number of terms N is polynomially bounded, therefore the (idealized) PRFs in these schemes can take any polynomial sized input. □

Corollary 2. *The schemes $\mathcal{SE}^{ecb}$ and $\mathcal{SE}^{cbc*}$ can be instantiated with ECB mode and CBC mode of cryptographic encryption respectively.*

Proof. Using Corollary 1 and assuming a block-cipher to be a bounded family of finite PRFs, $\mathcal{SE}^{ecb}$ case is trivial. As per Lemma 6, the security of $\mathcal{SE}^{cbc*}$ implies the security of $\mathcal{SE}^{cbc}$, and $\mathcal{SE}^{cbc}$ represents the CBC mode of encryption. Therefore, $\mathcal{SE}^{cbc*}$ can be instantiated with the CBC mode of encryption. □

Let $\mathcal{SE}^a \overset{\text{sem}}{\Rightarrow} \mathcal{SE}^b$ be an implication with respect to cryptographic semantics, namely, a cryptographic encryption scheme meeting the requirements of $\mathcal{SE}^b$ also meets the security requirements of $\mathcal{SE}^a$. The intuition behind the direction of the arrow is that if a symbolic protocol using $\mathcal{SE}^a$ is secure in a formal model then the protocol will remain secure if we replace $\mathcal{SE}^a$ with $\mathcal{SE}^b$. The equivalence $\overset{\text{sem}}{\equiv}$ is a two-way implication.

Lemma 1. *$\mathcal{SE}^{ecb} \overset{sem}{\Rightarrow} \mathcal{SE}^{rn}$; and $\mathcal{SE}^{rn} \overset{sem}{\not\Rightarrow} \mathcal{SE}^{ecb}$*

Proof. $\underline{\mathcal{SE}^{ecb}}$: First we consider $\mathcal{SE}^{ecb}(M)$: $\{M_1\}_K, ..., \{M_i\}_K, ..., \{M_N\}_K$. Each encrypted term $\{M_i\}_K$ in this scheme is a UV-PRF and is therefore WP,NM-secure (Proposition 1). Our aim is to derive the security properties of $\mathcal{SE}^{ecb}(M)$.

As shown by the following attack, $\mathcal{SE}^{ecb}(M)$ is not NM-secure. In an attack on NM-security, an adversary simply permutes the individual encrypted terms. For example, given $\{M_1\}_K, \{M_2\}_K$, the adversary can produce another valid ciphertext $\{M_2\}_K, \{M_1\}_K$ that has a related plaintext to the plaintext of the first ciphertext.

To show $\mathcal{SE}^{ecb}(M)$ is WP-secure, we construct a compiler that can translate an attack $\mathcal{I}_1$ on $\mathcal{SE}^{ecb}(M)$ to the attack $\mathcal{I}_2$ on UV-PRF. The compiler for WP-security is as follows: as soon as $\mathcal{I}_1$ outputs the plaintext M, $\mathcal{I}_2$ parses this plaintext to compute the plaintext of each individual encrypted term M_i. The compiler also works in the similar manner to derive ciphertexts. Since the encrypted terms are UV-PRF and are assumed to be WP-secure, the scheme $\mathcal{SE}^{ecb}(M)$ is WP-secure.

$\underline{\mathcal{SE}^{rn}}$: Next we consider $\mathcal{SE}^{rn}(M)$:
$V_1, \{M_1\}_{K,V_1}, ..., V_i, \{M_i\}_{K,V_i}, ..., V_N, \{M_N\}_{K,V_N}$. This scheme is clearly not NM-secure because the same permutation attack of $\mathcal{SE}^{ecb}(M)$ is also valid here. The scheme $\mathcal{SE}^{rn}(M)$ is WP-secure using the same type of reduction as we did for proving the WP-security of $\mathcal{SE}^{ecb}(M)$.

The scheme $\mathcal{SE}^{rn}(M)$, however, is IND-secure. For this purpose, we can calculate the probability distribution on the plaintexts corresponding to a given ciphertext. The size of each encrypted term in $\mathcal{SE}^{rn}(M)$ is s, as per Def. 4. Since in our model the sizes of a plaintext and its ciphertext are equal, therefore there are 2^s plausible plaintexts, on which we need to compute the probability distribution. For all V_i used in $\mathcal{SE}^{rn}(M)$, $|V_i| \geq s$ and $\widehat{V_i}$ is assigned from a uniform distribution. This means there are 2^s equally probable plaintexts for a given

ciphertext in a domain of size 2^s. Being V_i a free variable, each call to $\mathcal{SE}^{rn}$ uses fresh coin tosses. Therefore, no strategy can succeeds in the indistinguishability experiment even in a statistical sense. Hence, $\mathcal{SE}^{rn}$ is WP,IND-secure but is not NM-secure. Combining this result with that of $\mathcal{SE}^{ecb}$ completes the proof. □

Lemma 2. *$\mathcal{SE}^{bk} \overset{sem}{\Rightarrow} \mathcal{SE}^{rnb}$; $\mathcal{SE}^{rnb} \overset{sem}{\not\Rightarrow} \mathcal{SE}^{bk}$;*

Proof. $\underline{\mathcal{SE}^{bk}}$: First we consider $\mathcal{SE}^{bk}(M)$: $\{M_1, ..., M_i, ..., M_N\}_K$. Clearly this scheme represents one big UV-PRF and is therefore WP,NM-secure (Proposition 1). Since the scheme is deterministic, therefore it can not be IND-secure [29].

$\underline{\mathcal{SE}^{rnb}}$: Next we consider $\mathcal{SE}^{rnb}(M)$: $V, \{M_1, ..., M_i, ..., M_N\}_{K,V}$. This scheme is also a single UV-PRF and is therefore WP,NM-secure (Proposition 1). Further, the scheme $\mathcal{SE}^{rnb}(M)$ is IND-secure. For this purpose, we can calculate the probability distribution on the plaintexts corresponding to a given ciphertext, similar to Lemma 1. The size of the encrypted term in $\mathcal{SE}^{rnb}(M)$ is $N.s$, therefore there are $2^{N.s}$ plaintexts on which we need to compute the probability distribution.

Since the term $\widehat{V}$ is assigned from the uniform distribution, there are 2^s uniformly distributed valid plaintexts for a given ciphertext in a domain of size $2^{N.s}$. Each call of $\mathcal{SE}^{rnb}$ uses fresh coin tosses for $\widehat{V}$, therefore each of the plaintext distributions (of size 2^s) is independently located within the domain of size $2^{N.s}$. Therefore, a polynomial-time (CPA) adversary that queries the encryption oracle q times cannot distinguish between two plaintext distributions with a probability greater than $q.2^{-s}$. As $q.2^{-s}$ is negligible in s, $\mathcal{SE}^{rnb}$ is IND,NM,WP-secure. This result along with that of $\mathcal{SE}^{bk}$ complete the proof. □

Lemma 3. *$\mathcal{SE}^{ecb} \overset{sem}{\Rightarrow} \mathcal{SE}^{bk}$; $\mathcal{SE}^{bk} \overset{sem}{\not\Rightarrow} \mathcal{SE}^{ecb}$*

Proof. From the proof construction in Lemma 2, we know that $\mathcal{SE}^{bk}$ is WP,NM-secure but is not IND-secure. Similarly, from the proof of Lemma 1, we know that $\mathcal{SE}^{ecb}$ is WP-secure but is not NM,IND-secure. Combining these results proves this lemma. □

Lemma 4. *$\mathcal{SE}^{rn} \overset{sem}{\Rightarrow} \mathcal{SE}^{rnb}$; $\mathcal{SE}^{rnb} \overset{sem}{\not\Rightarrow} \mathcal{SE}^{rn}$;*

Proof. From the proof constructions of Lemma 2 and Lemma 1, we know that $\mathcal{SE}^{rn}$ is WP,IND-secure but not NM-secure, and $\mathcal{SE}^{rnb}$ is WP,NM,IND-secure. Hence the implication and non-implication hold. □

Lemma 5. *$\mathcal{SE}^{rn} \overset{sem}{\not\Rightarrow} \mathcal{SE}^{bk}$; $\mathcal{SE}^{bk} \overset{sem}{\not\Rightarrow} \mathcal{SE}^{rn}$;*

Proof. Trivial from the security properties of $\mathcal{SE}^{rn}$ and $\mathcal{SE}^{bk}$ in Lemma 3-4. □

Lemma 6. *$\widehat{\mathcal{SE}^{rn}} \overset{sem}{\equiv} \widehat{\mathcal{SE}^{cbc*}} \overset{sem}{\equiv} \widehat{\mathcal{SE}^{cbc}}$*

Proof. The three schemes are not NM-secure: reordering the encrypted terms in $\widehat{\mathcal{SE}^{rn}}$ is trivial and the corresponding plaintext will be be a permuted version of the original plaintext ; for the other two encryption schemes, on can simply

delete the last encrypted term and the corresponding plaintext will be a truncated version of the original plaintext.
Since IND-security implies WP-security, we only show the equivalence under IND-security. The equivalence relations hold if the three probability distributions (constructed from the cryptographic semantics and corresponding to the symbolic encryption of a message M in $\widehat{\mathcal{SE}^{rn}}$, $\widehat{\mathcal{SE}^{cbc*}}$, and $\widehat{\mathcal{SE}^{cbc}}$) are computationally indistinguishable. Note that for this proof it is not necessary whether or not a particular scheme is IND-secure, which although is implied from the proof construction of Lemma 1, namely $\widehat{\mathcal{SE}^{rn}}$ is IND-secure.

First, we consider $\widehat{\mathcal{SE}^{rn}}(M)$: $\widehat{V_1}, \{\widehat{M_1}\}_{\widehat{K},\widehat{V_1}}, ..., \widehat{V_i}, \{\widehat{M_i}\}_{\widehat{K},\widehat{V_i}}, ..., \widehat{V_N}, \{\widehat{M_N}\}_{\widehat{K},\widehat{V_N}}$.
As per Def. 2, we can replace N free variables with N uniformly distributed variables: $U_s^1, \{\widehat{M_1}\}_{\widehat{K},U_s^1}, ..., U_s^i, \{\widehat{M_i}\}_{\widehat{K},U_s^i}, ..., U_s^N, \{\widehat{M_N}\}_{\widehat{K},U_s^N}$.
We have $\text{PDF}(\{\widehat{M_1}\}_{\widehat{K},U_s^i}) \equiv PDF(U_s^{i'})$ due to the definition of UV-PRF (see Appendix. B). Therefore, $\text{PDF}(\widehat{\mathcal{SE}^{rn}}(M)) = \text{PDF}(U_s^1, U_s^{1'}, U_s^2, U_s^{2'}, ..., U_s^N, U_s^{N'})$.

Next, we consider $\widehat{\mathcal{SE}^{cbc*}}(M)$: $\widehat{V_1}, \{\widehat{M_1}\}_{\widehat{K},\widehat{V_1}}, ..., \{\widehat{M_i}\}_{\widehat{K},\widehat{C_{i-1}}}, ..., \{\widehat{M_N}\}_{\widehat{K},\widehat{C_{N-1}}}$.
Replacing the free variable V_1 with a uniformly distributed variable, we get the following: $U_s^1, \{\widehat{M_1}\}_{\widehat{K},U_s^1}, ..., \{\widehat{M_i}\}_{\widehat{K},\widehat{C_{i-1}}}, ..., \{\widehat{M_N}\}_{\widehat{K},\widehat{C_{N-1}}}$.
We have $\text{PDF}(\{\widehat{M_1}\}_{\widehat{K},U_s^1}) \equiv \text{PDF}(U_s^{2''})$ due to the definition of UV-PRF (see Appendix B). Recursively applying this relation on each term, we get
$\text{PDF}(\widehat{\mathcal{SE}^{cbc*}}(M)) = \text{PDF}(U_s^1, U_s^{2''}, ..., U_s^{N''})$.

In the third case, we consider $\widehat{\mathcal{SE}^{cbc}}(M)$:
$\widehat{V_1}, \{\widehat{M_1} \oplus \widehat{V_1}\}_{\widehat{K}}, ..., \{\widehat{M_i} \oplus \widehat{C_{i-1}}\}_{\widehat{K}}, ..., \{\widehat{M_N} \oplus \widehat{C_{N-1}}\}_{\widehat{K}}$.
Replacing the free variable V_1 with a uniform distribution, we get the following:
$U_s^1, \{\widehat{M_1} \oplus U_s^1\}_{\widehat{K}}, ..., \{\widehat{M_i} \oplus \widehat{C_{i-1}}\}_{\widehat{K}}, ..., \{\widehat{M_N} \oplus \widehat{C_{N-1}}\}_{\widehat{K}}$.
We have $\text{PDF}(\{\widehat{M_1} \oplus U_s^1\}_{\widehat{K}}) \equiv \text{PDF}(U_s^{2'''})$ due to the definition of Xor operator ($\text{PDF}(U_s) = \text{PDF}(U_s \oplus M)$) and the definition of UV-PRF. Applying this property to all of the terms we get the following relation:
$\text{PDF}(\widehat{\mathcal{SE}^{cbc*}}(M)) = \text{PDF}(U_s^1, U_s^{2'''}, ..., U_s^{N'''})$.

Clearly, in all of the three cases, the distribution of a ciphertext is on a series of random variables on the same uniform distribution of size s. Further, we can always make the lengths of the ciphertexts in the last two cases equal to the first case by inserting dummy random variables. Therefore, these three schemes are cryptographically equivalent. □

The results presented in this section rely on the cryptographic semantics of symbolic encryption. Using these semantics, the assumed cryptographic properties for these schemes are easy to interpret. In reality, however, a formal security model relies on some formal semantics (e.g., reduction semantics [9]), rather than any cryptographic semantics, for its correctness. A formal security model is essentially a logical system that supports the execution of a protocol. In practice, it may be the case that no formal security model is able to capture the differ-

ence between the four symbolic encryption schemes, which are cryptographically different.

In the next section, however, we show that there exists a formal security model in which the four symbolic encryption schemes are non-overlapping, i.e., these schemes do provide different types of security guarantees when used to model protocol executions. The formal analysis tool that we use to demonstrate our results is OFMC [16] (Open Source Fixedpoint Model Checker), a symbolic model checker for security protocols.

6 Separation in Practice

We use a constructive approach to show non-implications between the four notions of symbolic encryption in a formal security model. In particular, we invented four experimental (but not artificial) protocols and analyze their security for unbounded (infinite) number of symbolic sessions in OFMC. The purpose is to demonstrate that a security goal achieved by an experimental protocol depends on what notion of encryption is being used. Clearly, if a protocol achieves its goal using one type of encryption but fails if a different type of encryption is used then the protocol construction serves as a constructive proof of non-implication between the two types of encryption.

The tool OFMC consists of two different modules that run in parallel. For our purpose (to show non-implications), we rely on the *Fixedpoint module*, which uses abstract interpretation and over-approximation to verify protocols, by guaranteeing that no unsafe state is reachable no matter how many symbolic sessions are executed in parallel. Another nice feature of the Fixedpoint module is that it can generate security proofs for the interactive theorem prover *Isabelle* [34].

The four experimental protocols are listed in Fig. 4. The end-results implying the separation are shown in Fig. 5, where non-implications as proved in the previous section are also valid in OFMC. The OFMC specifications of all referenced protocols in this section can be found in Appendix C.

In the first protocol, a party B wants to send a confidential piece of data D to A, using an agreed session key $g(N_a, N_b, K)$. Here, $f(.)$ and $g(.)$ are (public) symbolic functions. This protocol uses $\mathcal{SE}^{ecb}$ and the result of the security analysis (in OFMC) shows that the protocol is secure with respects to its goal, i.e., confidentiality of D. This shows that there exist a secure symbolic protocol that only relies on the weak form of privacy provided by $\mathcal{SE}^{ecb}$.

The goal of the second protocol is the same as that of the first one, but it uses a different construction. We specify this protocol in three different ways, using $\mathcal{SE}^{bk}$, $\mathcal{SE}^{rn}$ and $\mathcal{SE}^{ecb}$ respectively, as listed in Appendix C. The analysis results show that the protocol is secure when $\mathcal{SE}^{bk}$ is used but is insecure (i.e., there exists an attack trace) when the other two forms of encryption are used. Hence, $\mathcal{SE}^{bk}$ implies neither $\mathcal{SE}^{ecb}$ nor $\mathcal{SE}^{rn}$. Therefore $\mathcal{SE}^{bk}$ provides a different type of security in this symbolic model.

In the third experimental protocol, the goal is to transmit D from B to A with both confidentially and authentication guarantees. This protocol is specified

Protocol 1
(1) $A \longrightarrow B : N_A$
(2) $B \longrightarrow A : \{N_A\}_{AB}, \{N_B\}_{AB}$
(3) $A \longrightarrow B : \{K\}_{AB}, \{f(N_A, N_B)\}_{AB}$
(4) $B \longrightarrow A : \{D\}_{g(K,N_A,N_B)}$
Goal: Confidentiality of D

Protocol 2
(1) $A \longrightarrow B : N_A$
(2) $B \longrightarrow A : \{N_A, N_B\}_{AB}$
(3) $A \longrightarrow B : \{f(N_A, N_B)\}_{AB}$
(4) $B \longrightarrow A : \{D\}_{f(N_A,N_B)}$
Goal: Confidentiality of D

Protocol 3
(1) $A \longrightarrow B : V_1, \{A\}_{K_{AB},V_1}, V_2, \{B\}_{K_{AB},V_2}, V_3, \{V_2\}_{K_{AB},V_3}$
(2) $B \longrightarrow A : \{D\}_{K_{AB},V_2}$
Goal: Authenticity and Confidentiality of D

Protocol 4
(1) $A \longrightarrow S : V_1, \{A, B\}_{K_{SA},V_1}$
(2) $S \longrightarrow B : V_1, \{A, B, K, \{K\}_{SA}\}_{K_{SB},V_1}$
(3) $B \longrightarrow A : \{K\}_{SA}, \{D\}_K$
Goal: Authenticity and Confidentiality of D

Fig. 4. Experimental Protocols for the Separation of Symbolic Encryption Schemes

in two different ways: one with $\mathcal{SE}^{rn}$ and the other using the $\mathcal{SE}^{ecb}$. The analysis results shows that the protocol is secure with $\mathcal{SE}^{rn}$ but there is an attack if the $\mathcal{SE}^{ecb}$ is used. This demonstrates the non-implication from $\mathcal{SE}^{rn}$ to $\mathcal{SE}^{ecb}$.

In the fourth experimental protocol, the goal is to transmit D from B to A using a server generated key K, with authentication and confidentiality guarantees. The protocol is specified in three different ways, firstly, using $\mathcal{SE}^{rnb}$ (randomized bulk symbolic encryption), secondly, using $\mathcal{SE}^{rn}$, and thirdly, using $\mathcal{SE}^{bk}$. The protocol is secure in the first case, but it is insecure in both of the later cases. This demonstrates two more non-implication results.

As shown in Fig. 5, separation results implies that the four notions of encryption indeed provide different types of security in a complete formal model. In the next section, we discuss some other interesting aspects of our work.

7 Discussion

In practice, it is nonetheless dangerous to assume that a system developer will actually discover and use the correct cryptographic scheme that meets the security requirements of a particular use of symbolic encryption, especially when the cryptographic semantics are not provided. That is why, the developers often use an implementation instance that seems appropriate, e.g., in this paper, the CBC implementations of encryption in NSSK and DSKK protocols indeed guarantee

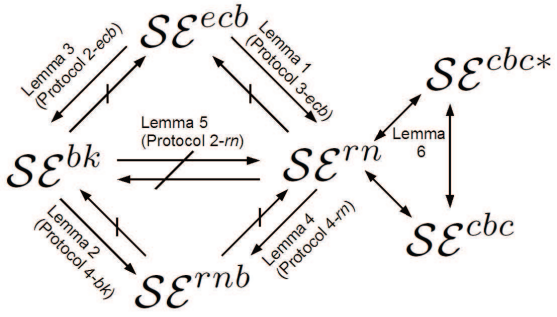

Fig. 5. Separation of Different type of Encryption in Symbolic Models

privacy in a strong sense [29]; however, non-malleability of the ciphertexts, an implicit assumption, is also required for the security of these protocols.

On the other hand, one may always choose to employ a strong encryption scheme meeting the requirements of $\mathcal{SE}^{rnb}$, however, the cost associated with such an overly cautious approach cannot be ignored in practice. For example, if the symbolic model of a protocol that uses $\mathcal{SE}^{ecb}$ is secure then this means that the protocol can be implemented in a relatively efficient manner: a random number generator is not required; the algorithm for message authentication code (MAC, used to guarantee non-malleability) is not required; and communication bandwidth is reduced because we do not need to transmit initialization vectors and MAC codes. Moreover, parallelization of the encryption process is straight forward, which may significantly reduce the execution time on a multi-core architecture. In many applications, such optimizations can make a huge difference, e.g., a hypervisor which has to process millions of requests per second. Our separation results show that many symbolic protocols remain secure when encryption requirements are met by a weaker symbolic encryption scheme, such as $\mathcal{SE}^{ecb}$ or $\mathcal{SE}^{cbc}$; in this way a level of safe optimization can be achieved.

In our observations, most people do not use randomized encryption in formal security models. It is evident from our work that randomness in a symbolic encryption scheme helps to provide stronger security guarantees—a fact that is well known in cryptography for a long time [29].

The attacks listed in § 3 and § 4 were not discovered using any tool, rather they were discovered using an old-fashioned paper-pencil method. After their discovery, we tried to reproduce them in OFMC. Since Xor operation (used in CBC-mode encryption) cannot be modeled in symbolic models, we used the equivalent form of CBC mode, $\mathcal{SE}^{rn}$. The tool reported different types of attacks, which can be found in Appendix A; we were unable to reproduce the reported attacks because the tool stops the exhaustive search as soon as it discovers an attack. Nevertheless, these observations indicate that the insecurity of protocols

can be detected more effectively by modeling cryptographic encryption using the proposed encryption schemes.

It is important to remember that safely instantiating a symbolic encryption scheme with a cryptographic encryption scheme does not mean that the resultant protocol will be secure, because there are many attacks that do not rely on encryption, e.g., Lowe's attack [14] on public-key version of Needham-Schroeder protocol relies on the assumption of a corrupt insider, Denning-Sacco's attack [4] relies on the availability of a compromised old session key. Moreover, there are many security vulnerabilities that are outside the realm of (mathematical) cryptography, e.g., buffer-overflow.

8 Conclusion

In this paper, we reported new attacks on reasonable implementations of well-known protocols. It appears that there is no inherent limitation in symbolic models which may have prevented detecting these attacks in an automated manner. We notice that encryption on multiple terms is traditionally specified as one big monolithic encrypted block, which, however, is not a good way of specifying it for practice-oriented security analysis. We presented four refined ways in which encryption can be specified in a symbolic model, and we show that each of these specifications implies a different set of cryptographic requirements. The proposed specifications not only help to avoid many implementation vulnerabilities similar to the reported attacks, but they also provide a degree of safe optimization. We hope that our work will bring symbolic encryption closer to the secure implementation of encryption.

References

1. Bellare, M. and Rogaway, P.: *On the construction of variable-input-length ciphers*, In Proc.: Fast Software Encryption, pub. Springer, pp. 231–244, 1999
2. Needham, R.M., Schroeder, M.D.: *Using encryption for authentication in large networks of computers*, Communications of the ACM 21(12), pp. 993–999, 1978
3. Needham, R.M., Schroeder, M.D.: *Authentication revisited*, ACM SIGOPS Operating Systems Review 21(1), pub. ACM, pp. 7–7, 1987
4. Denning, D.E., Sacco, G.M.: *Timestamps in key distribution protocols*, Communications of the ACM 24(8), pub. ACM, pp. 533–536, 1981
5. Abadi, M., Rogaway, P.: *Reconciling Two Views of Cryptography*, In TCS: Exploring New Frontiers of Theoretical Informatics, pub. Springer, pp. 3–22, 2000
6. Dolev, D., Yao, A.C.: *On the security of public key protocols*, IEEE Transactions on Information Theory, IT-29(12), pp. 198–208, 1983
7. Chevalier, Y., Kusters, R., Rusinowitch, M., Turuani, M.: *An NP decision procedure for protocol insecurity with XOR*, In Proc.: IEEE Symp. on Logic in CS, 2003
8. Abadi, M., Gordon, A.D.: *Reasoning about cryptographic protocols in the spi calculus*, CONCUR'97: Concurrency Theory, Springer-Verlag LNCS, 1997
9. Bodei, C., Buchholtz, M., Degano, P., Nielson, F., Nielson, H.R.: *Static validation of security protocols*, Journal of Computer Security 13(3), pp. 347–390, 2005

10. Micciancio, D., Warinschi, B.: *Soundness of formal encryption in the presence of active adversaries*, Theory of Cryptography, pub. Springer, pp. 133–151, 2004
11. Backes, M., Pfitzmann, B., Waidner, M.: *A composable cryptographic library with nested operations*, In Proc.: ACM Conf. on Comp. and Comm. Sec., 2003
12. Canetti, R., Herzog, J.: *Universally Composable Symbolic Security Analysis*, Journal of cryptology 24(1), pp. 83–147, 2011
13. Herzog, J., Liskov, M., Micali, S.: *Plaintext awareness via key registration*, Advances in Cryptology-CRYPTO 2003, pp. 548–564, 2003
14. Lowe, G.: *Breaking and fixing the Needham-Schroeder public-key protocol using FDR*, In J.: Tools and Algo. for Constr. and Analysis of Sys., Springer, 1996
15. Meadows, C.: *Formal methods for cryptographic protocol analysis: Emerging issues and trends*, In J.: Selected Areas in Communications 21(1), pub. IEEE, 2003
16. Basin, D., Mödersheim, S., Vigano, L.: *OFMC: A symbolic model checker for security protocols*, In J.: Int. J. of Info. Security 4(3), Springer, pp. 181–208, 2005
17. Boyd, C.: *Hidden assumptions in cryptographic protocols*, IEE Proceedings: Computers and Digital Techniques 137(6), pp. 433–436, 1990
18. Stubblebine, S.G., Meadows, C.A.: *Formal characterization & automated analysis of known-pair & chosen-text attacks*, In J.: Selected Areas in Comm., IEEE, 2000
19. W. Mao and C. Boyd, *On the use of encryption in cryptographic protocols*, Codes and Cyphers, available via Citeseer, 1995
20. Shoup, V.: *OAEP reconsidered*, Journal of Cryptology 15(4), 2008
21. Kremer, S., Ryan, M.D.: *Analysing the vulnerability of protocols to produce known-pair and chosen-text attacks*, Electronic Notes in TCS 128(5), Elsevier, 2005
22. Moore, J.H.: *Protocol failures in cryptosystems*, In: Proceedings of the IEEE 76(5), pp. 594–602, 1988
23. Paterson, K., Yau, A.:*Cryptography in theory and practice: The case of encryption in IPsec*, Advances in Cryptology-EUROCRYPT, pp. 12–29, 2006
24. Degabriele, J.P., Paterson, K.: *Attacking the IPsec standards in encryption-only configurations*, IEEE Symposium on Security and Privacy, pp. 335–349, 2007
25. Küsters, R., Truderung, T.: *Reducing protocol analysis with xor to xor-free case in horn theory based approach*, In Proc.: ACM Comp. & Comm. Security, 2008
26. Chevalier, Y., Vigneron, L.: *Automated unbounded verification of security protocols*, Computer Aided Verification, pub. Springer, pp. 125–171, 2002
27. Boyd, C., Mathuria, A.: *Protocols for authentication and key establishment*, pub. Springer, 2003
28. Bellovin, S.M.: *Problem areas for the IP security protocols*, In: 6th Usenix UNIX Security Symposium, 1996
29. Goldwasser, S., Micali, S.: *Probabilistic encryption*, In: Journal of computer and system sciences, 28(2), pp.270–299, 1984
30. Katz, J., Yung, M.: *Complete characterization of security notions for probabilistic private-key encryption*, In Proc.: ACM Symp. on Th. of Computing, 2000
31. Goldreich, O. and Goldwasser, S. and Micali, S.: *How to construct random functions*, In J.: JACM 33(4), pub. ACM, pp.792–807, 1986
32. Bellare, M., Kilian, J. and Rogaway, P.: *The security of the cipher block chaining message authentication code*, In J.: Elsevier Comp. and Sys. Sc. 61(3), 2000
33. Goldreich, O.: *Foundations of Cryptography: 1 & 2*, Cambridge Univ. Press, 2004
34. Nipkow, T., Paulson, L.C., and Wenzel, M.: *Isabelle/HOL: a proof assistant for higher-order logic*, Springer Verlag, 2002
35. Barthe, G., Grégoire, B., Heraud, S., and Béguelin, S.Z.: *Computer-Aided Security Proofs for the Working Cryptographer?*, CRYPTO, 2011

A Other Attacks on NSSK and DSSK protocols

The attack in Fig. 6 is on the ECB-version of the DSSK protocol, and it is quite similar to the attack on the ECB-version of the NSSK protocol reported in the paper.

Steps	Messages
Setup-(a)	
(1)	$\mathcal{I} \longrightarrow S : \mathcal{I}, B$
(2)	$S \longrightarrow \mathcal{I} : \{B\}_{S\mathcal{I}}, \{K_1\}_{S\mathcal{I}}, \{T_1\}_{S\mathcal{I}}, \{\{\mathcal{I}\}_{SB}\}_{S\mathcal{I}}, \{\{K_1\}_{SB}\}_{S\mathcal{I}}, \{\{T_1\}_{SB}\}_{S\mathcal{I}}$
Setup-(b)	
(1)	$\mathcal{I}(A) \longrightarrow S : A, K_1' = \{K_1\}_{SB}$
(2)	$S \longrightarrow \mathcal{I}(A) : \{\{K_1\}_{SB}\}_{SA}, \{K_2\}_{SA}, \{T_2\}_{SA}, \{\{A\}_{SK_1'}\}_{SA}, \{\{K_2\}_{SK_1'}\}_{SA}, \{\{T_2\}_{SK_1'}\}_{SA}$
Setup-(c)	
(1)	$\mathcal{I}(A) \longrightarrow S : A, K_1$
(2)	$S \longrightarrow \mathcal{I}(A) : \{K_1\}_{SA}, \{K_3\}_{SA}, \{T_3\}_{SA}, \{\{A\}_{SK_1}\}_{SA}, \{\{K_3\}_{SK_1}\}_{SA}, \{\{T_3\}_{SK_1}\}_{SA}$
Attack	
(1)	$A \longrightarrow S : A, B$
(2a)	$S \longrightarrow \mathcal{I}(A) : \{B\}_{SA}, \{K_4\}_{SA}, \{T_4\}_{SA}, \{\{A\}_{SB}\}_{SA}, \{\{K_4\}_{SB}\}_{SA}, \{\{T_4\}_{SB}\}_{SA}$
(2b)	$\mathcal{I}(S) \longrightarrow A : \{B\}_{SA}, \{K_1\}_{SA}, \{T_4\}_{SA}, \{\{A\}_{SB}\}_{SA}, \{\{K_1\}_{SB}\}_{SA}, \{\{T_4\}_{SB}\}_{SA}$
(3)	$A \longrightarrow B : \{\{A\}_{SB}\}_{SA}, \{\{K_1\}_{SB}\}_{SA}, \{\{T_4\}_{SB}\}_{SA}$

Fig. 6. Attack on ECB-version of DSSK Protocol

The attacks in Fig 7 and Fig. 11 are impersonation attacks, which are reported in the output of OFMC analysis of the ECB-versions of NSSK and DSSK protocols; the attack in Fig 8 was indicated in the feedback that we received initially. Interestingly, we can also attack the CBC-version of the NSSK protocol using the same idea as listed in Fig. 9 and Fig. 10, however, the CBC-version of the DSSK protocol cannot be attacked on the same lines.

In the ECB-version of the NSSK protocol, A shares N_A as the session key with $\mathcal{I}$, while in its CBC-version the session key is $iv_1 \oplus N_A \oplus c_2$; for the DSSK protocol, the session key is T_4.

(1)	$A \longrightarrow S : A, B, N_A$
(2a)	$S \longrightarrow \mathcal{I}(A) : \{N_A\}_{SA}, \{B\}_{SA}, \{K_4\}_{SA}, \{\{K_4\}_{SB}\}_{SA}, \{\{A\}_{SB}\}_{SA}$
(2b)	$\mathcal{I}(S) \longrightarrow A : \{N_A\}_{SA}, \{B\}_{SA}, \{N_A\}_{SA}, \{\{K_4\}_{SB}\}_{SA}, \{\{A\}_{SB}\}_{SA}$

Fig. 7. Another Attack (1) on ECB-version of NSSK Protocol

(1)	$A \longrightarrow S : A, B, N_A$
(2a)	$S \longrightarrow \mathcal{I}(A) : \{N_A\}_{SA}, \{B\}_{SA}, \{K_4\}_{SA}, \{\{K_4\}_{SB}\}_{SA}, \{\{A\}_{SB}\}_{SA}$
(2b)	$\mathcal{I}(S) \longrightarrow A : \{N_A\}_{SA}, \{B\}_{SA}, \{K_4\}_{SA}, \{K_4\}_{SA}, \{\{A\}_{SB}\}_{SA}$
(3)	$A \longrightarrow B : K_4, \{\{A\}_{SB}$

Fig. 8. Another Attack (2) on ECB-version of NSSK Protocol

(1)	$A \longrightarrow S : A, B, N_A$
(2a)	$S \longrightarrow \mathcal{I}(A) : iv_1, iv_2, c_1 = \{iv_1 \oplus N_A\}_{SA}, c_2 = \{c_1 \oplus B\}_{SA}, c_3 = \{c_2 \oplus K_4\}_{SA},\ c_4 = \{c_3 \oplus \{iv_2 \oplus K_4\}_{SB}\}_{SA}, c_5 = \{c_4 \oplus \{\{iv_2 \oplus K_4\}_{SB} \oplus B\}_{SB}\}_{SA}$
(2b)	$\mathcal{I}(S) \longrightarrow A : iv_1, iv_2, c_1 = \{iv_1 \oplus N_A\}_{SA}, c_2 = \{c_1 \oplus B\}_{SA}, c_1 = \{iv_1 \oplus N_A\}_{SA},\ c_4 = \{c_3 \oplus \{iv_2 \oplus K_4\}_{SB}\}_{SA}, c_5 = \{c_4 \oplus \{\{iv_2 \oplus K_4\}_{SB} \oplus B\}_{SB}\}_{SA}$

Fig. 9. Another Attack (1) on CBC-version of NSSK Protocol

(1)	$A \longrightarrow S : A, B, N_A$
(2a)	$S \longrightarrow \mathcal{I}(A) : iv_1, iv_2, c_1 = \{iv_1 \oplus N_A\}_{SA}, c_2 = \{c_1 \oplus B\}_{SA}, c_3 = \{c_2 \oplus K_4\}_{SA},\ c_4 = \{c_3 \oplus \{iv_2 \oplus K_4\}_{SB}\}_{SA}, c_5 = \{c_4 \oplus \{\{iv_2 \oplus K_4\}_{SB} \oplus B\}_{SB}\}_{SA}$
(2b)	$\mathcal{I}(S) \longrightarrow A : iv_1, iv_2, c_1 = \{iv_1 \oplus N_A\}_{SA}, c_2 = \{c_1 \oplus B\}_{SA}, c_3 = \{c_2 \oplus K_4\}_{SA}, c_3 = \{c_2 \oplus K_4\}_{SA}, c_5 = \{c_4 \oplus \{\{iv_2 \oplus K_4\}_{SB} \oplus B\}_{SB}\}_{SA}$
(3)	$\mathcal{I}(S) \longrightarrow A : c_3 \oplus c_2 \oplus K_4, c_3 \oplus c_4 \oplus \{\{iv_2 \oplus K_4\}_{SB} \oplus B\}_{SB}$

Fig. 10. Another Attack (2) on CBC-version of NSSK Protocol

(1)	$A \longrightarrow S : A, B$
(2a)	$S \longrightarrow \mathcal{I}(A) : \{B\}_{SA}, \{K_4\}_{SA}, \{T_4\}_{SA}, \{\{A\}_{SB}\}_{SA}, \{\{K_4\}_{SB}\}_{SA}, \{\{T_4\}_{SB}\}_{SA}$
(2b)	$\mathcal{I}(S) \longrightarrow A) : \{B\}_{SA}, \{T_4\}_{SA}, \{T_4\}_{SA}, \{\{A\}_{SB}\}_{SA}, \{\{K_4\}_{SB}\}_{SA}, \{\{T_4\}_{SB}\}_{SA}$

Fig. 11. Another Attack on ECB-version of DSSK Protocol

B Computational Indistinguishability of U_s and UV-PRF

First, we claim that $\text{PDF}(U_s) \equiv \text{PDF}(\{\widehat{U_s}\}_{\widehat{K}})$ for an adversary $\mathcal{I}$ who repeats the following indistinguishability experiment p times and asks q queries from an encryption oracle computing $\{\widehat{M}\}_{\widehat{K}}$, on adversary's input of M.

Experiment: The value of U_s is selected from public fair coin tosses, in particular, $\mathcal{I}$ cannot influence the outcome but he knows the outcome. Then, U_s is forwarded to a challenger. The challenger has a hidden binary value b chosen from a fair coin toss before the start of the game and remains the same in all experiments involving $\mathcal{I}$. If $b = 0$, the challenger returns another random variable U'_s. If $b = 1$, the challenger returns $\{\widehat{U_s}\}_{\widehat{K}}$.

The adversary wins if $\mathcal{I}$ has significant advantage in correctly computing the value of b at the end of a game. The relation $\text{PDF}(U_s) \equiv \text{PDF}(\{\widehat{U_s}\}_{\widehat{K}})$ holds if the adversary's advantage in a game involving p above experiments is negligible in s. By definition, UV-PRF uses internal coin tosses to compute its output if the input is a new value. In this case, the output will be uniformly distributed. If a UV-PRF is queried on an input that it has previously received then the reply will be deterministic. In the first case, the adversary can not gain any information and in the later case he may gain some information. So we calculate the probability of the event that a UV-PRF performs internal coin tosses, in a given experiment on input U_s. This probability represents an upper bound on adversary's advantage: $\text{Adv}_{\mathcal{I}}(p, q) \leq (p + q).2^{-s}$, which is negligible in s. Therefore $\mathcal{I}$ cannot win with a probability significantly greater than 0.5 (a random guess).

Next, we claim that $\text{PDF}(U_s) \equiv \text{PDF}(\{\widehat{U_s}\}^N_{\widehat{K}})$, where $\{\widehat{U_s}\}^N_{\widehat{K}})$ stands for N recursive calls to the UV-PRF, as done in the last (N_{th}) encrypted term in $\mathcal{SE}^{cbc*}(M)$ and $\mathcal{SE}^{cbc}(M)$. The experiment in this case is exactly same as above, except when $b = 1$, the challenger returns $\{\widehat{U_s}\}^N_{\widehat{K}}$. Clearly, the output of a recursive call involving N UV-PRFs will be uniformly distributed if at least one of the UV-PRF performs the internal coin tosses; this will occur with a probability greater/equal to $1 - (p.N.2^{-s})$. Consequently, the upper bound on adversary's advantage is as follows: $\text{Adv}_{\mathcal{I}}(p, q, N) \leq (pN + q).2^{-s}$
Since N is assumed to be polynomially bounded in s, therefore the adversary's advantage is negligible.

Next we claim that $\text{PDF}(\{\widehat{M}\}_{\widehat{K,U_s}}) \equiv \text{PDF}(\{\widehat{U_s}\}_{\widehat{K}})$. For the input/output size $s = |M|$, there are $2^s.(2^s!)$ total number of unique UV-PRFs; here, '!' stands for the factorial function. The family of UV-PRFs corresponding to $\{\widehat{M}\}_{\widehat{K,U_s}}$ contains 2^{2s} functions, and the family of UV-PRFs corresponding to $\{\widehat{U_s}\}_{\widehat{K}}$ contains 2^s functions. Since $2^s \ll 2^{2s} \ll 2^s.(2^s!)$, the probability that same UV-PRF occurs more than one in a family is negligible in s, namely less than $1/(2^s!)$. Therefore, the two families are indistinguishable and the relation $\text{PDF}(\{\widehat{M}\}_{\widehat{K,U_s}}) \equiv \text{PDF}(\{\widehat{U_s}\}_{\widehat{K}})$ holds.

Finally, as a consequence of the this equivalence, we have $\mathrm{PDF}(U_s) \equiv \mathrm{PDF}(\{\widehat{M}\}_{\widehat{K,U_s}})$ and $\mathrm{PDF}(U_s) \equiv \mathrm{PDF}(\{\widehat{M}\}^{N}_{\widehat{K,U_s}})$ for $\mathcal{I}$. □

C Specification and Results for Experimental Protocols

C.1 Protocol 1 (*ECB Encryption*)

```
Protocol: toy_1
Types: Agent A,B;
       Number NA,NB,D,K;
       Function sk,f
Knowledge: A: A,B,sk(A,B),f;
           B: A,B,sk(A,B),f
Actions:
A->B: NA
B->A: {|NA|}(sk(A,B)),{|NB|}(sk(A,B))
A->B: {|K|}sk(A,B),{|f(NA,NB)|}(sk(A,B))
B->A: {|D|}(f(K,NA,NB))
Goals:
A ->* B: D
```

C.2 Protocol 2 (*Bulk Encryption*)

```
Protocol: toy2
Types: Agent A,B;
       Number NA,NB,D,K;
       Function sk,f
Knowledge: A: A,B,sk(A,B),f;
           B: A,B,sk(A,B),f
Actions:
A->B: NA
B->A: {|NA,NB|}(sk(A,B))
A->B: {|f(NA,NB)|}(sk(A,B))
B->A: {|D|}(f(NA,NB))
Goals:
A ->* B: D
```

C.3 Protocol 2 (*ECB Encryption*)

```
Protocol: toy2n1
Types: Agent A,B;
       Number NA,NB,D,K;
       Function sk,f
Knowledge: A: A,B,sk(A,B),f;
           B: A,B,sk(A,B),f
```

```
Actions:
A->B: NA
B->A: {|NA|}(sk(A,B)),{|NB|}(sk(A,B))
A->B: {|f(NA,NB)|}(sk(A,B))
B->A: {|D|}(f(NA,NB))

Goals:
A ->* B: D

ATTACK TRACE
(x20,1) -> i: NA(1)
i -> (x602,1): NA(1)
(x602,1) -> i: {NA(1)}_(sk(x20.x602)).{NB(2)}_(sk(x20.x602))
i -> (x20,1): {NA(1)}_(sk(x20.x602)).{NA(1)}_(sk(x20.x602))
(x20,1) -> i: {f(NA(1).NA(1))}_(sk(x20.x602))
i -> (x20,1): {x601}_(f(NA(1).NA(1)))
i -> (i,17): x601
i -> (i,17): x601
```

C.4 Protocol 2 (*Randomized Encryption*)

```
Protocol: toy2n3
Types: Agent A,B;
       Number NA,NB,D,K,V1,V2,V3,V4;
       Function sk,f
Knowledge: A: A,B,sk(A,B),f;
           B: A,B,sk(A,B),f
Actions:
A->B: NA
B->A: V1,{|NA|}(sk(A,B),V1),V2,{|NB|}(sk(A,B),V2)
A->B: V3,{|f(NA,NB)|}(sk(A,B),V3)
B->A: V4,{|D|}(f(NA,NB),V4)

Goals:
A ->* B: D

ATTACK TRACE
(x20,1) -> i: NA(1)
i -> (x602,1): NA(1)
(x602,1) -> i: V1(2).{NA(1)}_(sk(x20.x602).V1(2)).V2(2).{NB(2)}_(sk(x20.x602).V2(2))
i -> (x20,1): V1(2).{NA(1)}_(sk(x20.x602).V1(2)).V1(2).{NA(1)}_(sk(x20.x602).V1(2))
(x20,1) -> i: V3(3).{f(NA(1).NA(1))}_(sk(x20.x602).V3(3))
i -> (x20,1): x510.{x511}_(f(NA(1).NA(1)).x510)
i -> (i,17): x511
i -> (i,17): x511
```

C.5 Protocol 3 (*Randomized Encryption*)

```
Protocol: Toy3
Types: Agent A,B;
       Number D,V1,V2,V3;
       Function sk
Knowledge: A: A,B,sk(A,B);
           B: A,B,sk(A,B)
Actions:
A->B: V1,{|A|}(sk(A,B),V1),V2,{|B|}(sk(A,B),V2),V3,{|V2|}(sk(A,B),V3)
B->A: {|D|}(sk(A,B),V2)
Goals:
B *->* A: D
```

C.6 Protocol 3 (*ECB Encryption*)

```
Protocol: Toy3n1
Types: Agent A,B;
       Number D;
       Function sk
Knowledge: A: A,B,sk(A,B);
           B: A,B,sk(A,B)
Actions:
A->B: {|A|}(sk(A,B)),{|B|}(sk(A,B))
B->A: {|D|}(sk(A,B))
Goals:
B *->* A: D

ATTACK TRACE (Hint: Use -classic option.)
(x702,1) -> i: {x702}_(sk(x702.x702)).{x702}_(sk(x702.x702))
(x702,2) -> i: {x702}_(sk(x702.x702)).{x702}_(sk(x702.x702))
i -> (x702,1): {x702}_(sk(x702.x702)).{x702}_(sk(x702.x702))
(x702,1) -> i: {D(3)}_(sk(x702.x702))
i -> (x702,1): {D(3)}_(sk(x702.x702))
i -> (x702,2): {D(3)}_(sk(x702.x702))
```

C.7 Protocol 4 (*Randomized Bulk Encryption*)

```
Protocol: Toy4
Types: Agent s,A,B;
       Number D,K,V1;
       Function sk
Knowledge: A: A,B,s,sk(A);
           B: A,B,s,sk(B);
           s: A,B,s,sk
Actions:
```

```
A->s: V1,{|A,B|}(sk(A),V1)
s->B: V1,{|A,B,K,{|K|}(sk(A))|}(sk(B),V1)
B->A: {|K|}(sk(A)),{|D|}(K)
Goals:
B *->* A: D
```

C.8 Protocol 4 (*Randomized Encryption*)

```
Protocol: Toy4n3
Types: Agent s,A,B;
       Number D,K,V1;
       Function sk
Knowledge: A: A,B,s,sk(A);
           B: A,B,s,sk(B);
           s: A,B,s,sk
Actions:
A->s:V1,{|A|}(sk(A),V1),{|B|}(sk(A),V1)
s->B:V1,{|A|}(sk(B),V1),{|B|}(sk(B),V1),{|K|}(sk(B),V1),{|{|K|}(sk(A))|}(sk(B),V1)
B->A:{|K|}(sk(A)),{|D|}(K)
Goals:
B *->* A: D
```

```
ATTACK TRACE
i -> (s,1): x305.{i}_(sk(i).x305).{x402}_(sk(i).x305)
(s,1) -> i: x305.{i}_(sk(x402).x305).{x402}_(sk(x402).x305).{K(1)}_(sk(x402).x305).
{{K(1)}_(sk(i))}_(sk(x402).x305)
i -> (x402,1): x305.{x402}_(sk(x402).x305).{x402}_(sk(x402).x305).{K(1)}_(sk(x402).x305).
{{K(1)}_(sk(i))}_(sk(x402).x305)
(x402,1) -> i: {K(1)}_(sk(i)).{D(2)}_K(1)
i -> (i,17): D(2)
i -> (i,17): D(2)
```

C.9 Protocol 4 (*Bulk Encryption*)

```
Protocol: Toy4n2
Types: Agent s,A,B;
       Number D,K;
       Function sk,f
Knowledge: A: A,B,s,sk(A);
           B: A,B,s,sk(B);
           s: A,B,s,sk
Actions:
A->s:{|A,B|}(sk(A))
s->B:{|A,B,K,{|K|}(sk(A))|}(sk(B))
B->A:{|K|}(sk(A)),{|D|}(K)
Goals:
B *->* A: D
```

```
ATTACK TRACE
(x601,1) -> i: {x601.x602}_(sk(x601))
(x601,2) -> i: {x601.i}_(sk(x601))
i -> (s,1): {x601.i}_(sk(x601))
(s,1) -> i: {x601.i.K(3).{K(3)}_(sk(x601))}_(sk(i))
i -> (x601,1): {K(3)}_(sk(x601)).{x506}_K(3)
```