Computationally Complete Symbolic Attacker and Key Exchange

Gergei Bana*, Koji Hasebe† and Mitsuhiro Okada‡
*MSR-INRIA Joint Centre,
Palaiseau, France
bana@math.upenn.edu
†Graduate School of Systems and Information Engineering
University of Tsukuba Tsukuba, Japan
hasebe@iit.tsukuba.ac.jp
‡Department of Philosophy
Keio University
Tokyo, Japan
mitsu@abelard.flet.keio.ac.jp

***Abstract*—In this paper we continue to develop and to illustrate the power of the recent technique of computationally complete symbolic attackers proposed by Bana and Comon-Lundh in [6] for computationally sound verification of security protocols. We now focus on the situation when keys are sent and then are used to encrypt 'securely'. We define predicates expressing 'key usability' meaning that a key has been uncorrupted and is still unbreakable and/or unforgeable, depending on what is desirable. We axiomatize our notions such that they are suitable for (but not limited to) inductive reasoning: if something is uncorrupted up-to a point, then certain newly sent messages do not destroy this property. We examine both IND-CCA2 and KDM-CCA2 encryptions, both symmetric and asymmetric situations. For unforgeability, we consider INT-CTXT encryptions. We illustrate how our notions can be applied in protocol proofs, in particular, how they handle key cycles.**

I. Introduction

The aim of computational soundness of protocol verification is that symbolic proofs imply computational security. Approaches to computational soundness in case of active adversaries can be divided into two groups. Works in one [1], [18], [2], [16], [22] define *symbolic adversaries*, and soundness theorems state that under certain circumstances, if there is no successful symbolic attack, then there is no successful computational attack. The other group aims to work directly in the computational model [19], [7], [14], [10], [11]. In this latter case, soundness means that the properties on which symbolic manipulations are conditioned, hold computationally.

The first group, where symbolic attacker is defined, gives hope that already existing automated tools may be adopted for computationally sound verification, but these soundness theorems require a large set of assumptions. These assumptions, as well as reasons why they are not realistic are discussed in [17]. Such assumptions are, for example, that bit strings can be unambiguously parsed into symbolic terms, or, that no key cycles occur, or, that all keys are honestly generated, that there is no dynamic corruption, and so on. Recently, Backes et al. in [3] showed a way to avoid some of these problems, such as key-cycles and badly generated keys, but they needed a very strong notion for the encryption scheme called PROG-KDM security, and they still used an entire page to list all the conditions needed to limit the implementations of security predicates for soundness. Unambiguous parsing of bit strings is also still necessary. So PROG-KDM security and the other conditions are necessary to receive computational guarantees with this analysis even if computational security of the analyzed protocol holds without these properties.

Recently, Bana and Comon-Lundh presented in [6] (and in an improved version [5]) a new kind of symbolic attacker. They called it *computationally complete symbolic adversary*, as it is capable of doing everything that a computational adversary is capable of. They observed that the discrepancy between symbolic and computational proofs emerges from the fact that while the usual computational security assumptions on the primitives (such as IND-CCA2 security of the encryption) define what the adversary cannot violate, and the security of the protocol is derived from the security of the primitives, symbolic adversaries are defined by listing all the adversarial capabilities (Dolev-Yao rules). Hence, to adjust the viewpoint of the symbolic analysis to that of the computational, instead of listing every kind of move a symbolic adversary is allowed to do, Bana and Comon-Lundh list a few rules (axioms) that the symbolic adversary is not allowed to violate. Anything that does not contradict these axioms is allowed for the adversary. Hence, a successful symbolic attack in their case means that the violation of the security property of the protocol is consistent with the axioms. The axioms that are introduced must be computationally sound with respect to a computational interpretation that they defined. Their main result was that once it is shown that no successful symbolic adversary can exist complying some set of axioms, then for any computational implementation satisfying that set of axioms, successful computational attacks are impossible as long as the number of sessions is bounded in the security parameter.

The difference between the the original Dolev-Yao technique and that of Bana and Comon-Lundh can be best understood from the following pictures. In the Dolev-Yao technique, as more and more rules are added, the symbolic adversarial capabilities are increasing, the symbolic adversary covers more and more of the computational capabilities. However, so far no-one has been able to come up with rules that properly cover all possible computational capabilities. Hence, as Figure 1 shows, there are always some computational capabilities that are uncovered by the symbolic ones. All computational soundness results that use the Dolev-Yao symbolic adversaries in the end have to impose some significant limitations on the computational implementation.

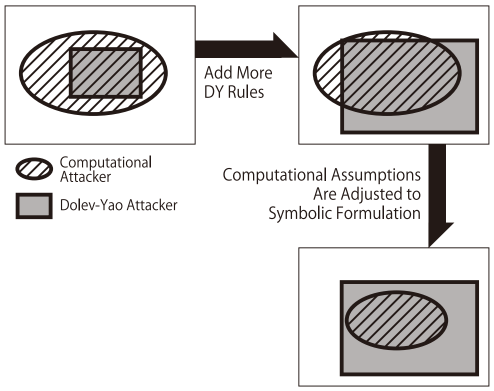

Figure 1. Computational Soundness of the Dolev-Yao Symbolic Adversary

On the other hand, in the approach of Bana and Comon-Lundh, without axioms, the symbolic adversary is allowed to do anything. As axioms are added, the symbolic adversary's capabilities are decreasing. The meaning of their main theorem is that as long as the axioms are computationally sound, the symbolic adversarial capabilities cover all of the computational adversarial capabilities. This is illustrated in Figure 2. Clearly, if the symbolic adversary is too strong, security of protocols cannot be proven. Therefore, the aim is, to create a *library of axioms* that are sound and are sufficient to prove actual protocols.

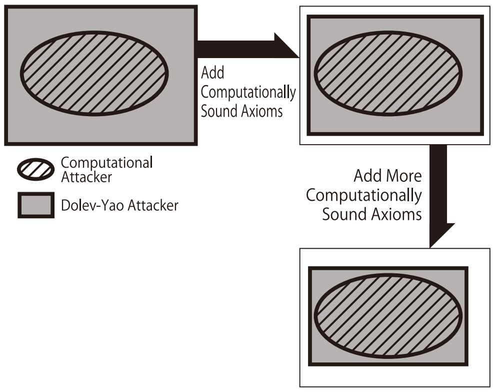

Figure 2. Computational Soundness of the Symbolic Adversary of Bana and Comon-Lundh

In the original work, Bana and Comon-Lundh only presented the general framework, and introduced some axioms as a proof of concept, but did not show that the technique can be used to verify actual protocols. This was done in a more recent publication [9] by Bana et al. They introduced some further modular, computationally sound axioms, and with them verified secrecy and authentication of an actual protocol, namely, the Needham-Schroeder-Lowe protocol. For the proof, they also needed an additional property, that is not sound in general but is necessary for the security of the NSL protocol, as without it there is an attack. However, as Backes et al. have pointed out in [3], the axioms in [9] were not suitable (except for very simple situations) when decryption keys were sent around in the course of the protocol. The current work aims to address this problem.

In order to deal with key exchange, the necessary element to incorporate in the framework is *key usability*, an idea introduced in [20] (working directly in the computational model). This notion is meant to express whether a properly generated key, at a certain point of the protocol execution is still usable for secure encryption or whether it has been corrupted. Clearly, if a decryption key (or just a key in the symmetric case) is sent in the clear, the encryption key belonging to it cannot be used for secure encryption any more. Furthermore, a key that was sent in a key cycle, may also have lost its capability to encrypt securely if the encryption scheme is only IND-CCA2 secure. On the other hand, encrypting a fresh nonce with a key does not corrupt its further usability in case of IND-CCA2 encryption. Or, having a key-cycle within a secure encryption with some other unsent key again does not corrupt the former key.

More precisely, for overall consistency of notation, instead of key usability we introduce the negation it, namely *key corruption* as a predicate. We define key corruption predicates both for symmetric and asymmetric encryptions, and both for IND-CCA2 [12] and KDM-CCA2 [4], [15] cases, and also for INT-CTXT [13] unforgeability. There are some essential differences from the way it was defined in [20], which we will explain when we define the notion.

Further essential new feature of this paper is the introduction of predicates representing adversarial derivability with oracle access. This makes the axioms quite a bit simpler than just using derivability as in [9]. Depending on whether IND-CCA2 or KDM-CCA2 oracles are used, and also depending on whether the encryption is symmetric or asymmetric, we define four such derivability with oracle access predicates.

Then, we introduce axioms and show that they are computationally sound. The axioms we introduce are entirely *modular. Introducing further primitives will not destroy the soundness of these axioms, they do not have to be proved again.* That is, if we want to prove a protocol that uses more primitives such as signatures etc. besides encryption, then we only have to introduce new axioms for the new primitives. For encryption, the current axioms can still be used unchanged. Hence, a library of axioms can be gradually developed by adding more and more axioms.

It is worth noting that with the new predicates for key usability and derivability with oracles, the axioms here corresponding to the secrecy and non-malleability axioms of [9] are now rather immediate consequences of the definitions of these predicates and do not need secure implementations of the encryption. The axioms that do need the security of the implementation are those saying that fresh keys are uncorrupted (i.e. securely encrypt).

We would like to emphasize that we introduce axioms for KDM-CCA2 security to be able to analyze protocols for which KDM security is computationally necessary. Unlike the work presented in [3], in our case, for those protocols that do not require KDM security for their computational soundness, the use of our IND-CCA2 axioms sufficient.

After presenting the axioms and their soundness proofs, we look at several simple examples to illustrate how to use the axioms with special focus on how IND-CCA2 axioms can be used if key cycles do not emerge. Finally, we present the result of our proof of the Amended Symmetric Needham-Schroeder Protocol. This protocol first distributes a session key, and then uses the distributed key to share a secret. Using the IND-CCA2 and INT-CTXT axioms, we proved that the key is securely distributed, that the shared nonce remains secret, and that agreement and authentication hold. The full proof of the protocol done by hand is posted online [8].

The technique of [6] allows to avoid *all* restrictions mentioned before on the computational world. *Once a protocol is proven secure in our symbolic model with respect to a set of axioms, then all properties that the computational implementation has to satisfy for computational security are included in the axioms.* Any number of bad keys are allowed to be generated by the adversary; any number of corrupted, uncorrupted, or dynamically corrupted parties can be present. As for parsing of bit strings into terms, previous soundness results relied on unambiguous parsing. Within this framework, we do not need such an assumption. We do not even need the condition that encryptions, pairing etc are length regular (i.e. encryption, pairing of inputs that have the same length also have the same length).

The only significant restriction remains that the technique is not capable to detect attacks for which unbounded number of sessions are necessary. However, the usual Dolev-Yao technique is not capable of doing this either. But we still need to characterize situations when nevertheless elimination of successful symbolic adversaries the way we define them means that there are no computational attacks even if unbounded number of sessions are allowed. For example, if only CCA2 encryptions and pairings are used to construct messages, we believe that this statement holds. Analysis of this problem is still left for future work.

The contributions of this paper include (i) syntax and computational semantics of key corruption and derivability with oracle access, (ii) a library of axioms for symmetric and asymmetric IND-CCA2 and KDM-CCA2 encryptions and INT-CTXT encryptions as well, (iii) soundness result of the axioms, (iv) short examples to illustrate how the axioms are used, (v) summary of the verification of the symmetric Needham-Schroeder protocol with this tool as a proof of concept. The NSL proof in [9] can also be done with the current set of axioms the same way as it was done there.

This paper is organized as follows: we start by recalling the framework of [6] (Section II) along with the new syntax in II-B. Symbolic execution in II-C is the same as before. In Section III, we first summarize the computational execution and the interpretation of the original equality and derivability predicates, and present the improved computational semantics of compound formulas of [5]. Section IV is devoted to the semantics of the new derivability predicates with oracles and their axioms, and Section V is the same for key usability. In Section VI we state the soundness theorem. In Section VII, we show a few simple examples of how inconsistency of certain formulas with the axioms can be proven. Finally, in VIII, we state the result of the verification of the amended symmetric NS protocol with our tool.

II. Symbolic Execution

The core of the framework used in this paper was introduced by Bana and Comon-Lundh in [6]. However, we have some major new additions here. Along with these new innovations, we present a brief summary of the original system.

The most important new aspect of the symbolic execution in [6] was to replace the Dolev-Yao technique's fixed definition of $x_1, ..., x_n \vdash y$ with some *derivability predicate* [1] $x_1, ..., x_n \rhd y$ for which the symbolic semantics is not fixed. Namely, while in the Dolev-Yao technique, $x_1, ..., x_n \vdash y$ meant that using only the Dolev-Yao rules y can be computed from $x_1, ..., x_n$, in our case $x_1, ..., x_n \rhd y$ is given some unfixed symbolic interpretation in an abstract model $\mathcal{M}$ for which we only require to satisfy some axioms. The axioms should be computationally sound, and instead of expressing what the adversary is allowed to do, they express what the adversary cannot violate. The axioms do imply that from symbolic computability, satisfaction of the

[1]Note, that in [6] this predicate also was denoted as $x_1, ..., x_n \vdash y$ although $\vdash$ is usually reserved for denoting deducibility in a proof system. We find that somewhat confusing, so we use the notation $x_1, ..., x_n \rhd y$ to emphasize that we do not mean some specific deducibility by it, it is a predicate.

derivability predicate follows, for example, $\{y\}_K, K \rhd y$. But in our system these rules are not what the adversary can at most do, but what it can certainly at least do (in other words, the adversary is not allowed to be unable to do it). The idea is that symbolic interpretation of $x_1, ..., x_n \rhd y$ should be at least as powerful as computability of y from $x_1, ..., x_n$ by some polynomial time algorithm, and so the only limitations that we want to put on symbolic satisfaction of $\rhd$ are limitations that are derived from computational computability.

One of the major innovations that we propose here is that as it turns out, axioms and deductions become simpler if we allow the use of some oracles for the adversary. When we consider IND-CCA2 public key encryption, then it is better to introduce a new derivability predicate, $x_1, ..., x_n \rhd^{\text{aic2}} y$ with the computational semantics meaning that the interpretation of y can be derived from the interpretation of $x_1, ..., x_n$ by a PPT adversary with the help of decryption oracles, that decrypt everything that are not results of encryptions on the left hand side. Similarly, for the symmetric case, we can introduce $x_1, ..., x_n \rhd^{\text{sic2}} y$ meaning that y can be derived from $x_1, ..., x_n$ with the help of decryption oracles and encryption oracles. The encryption oracles here are needed, because the IND-CCA2 definition for symmetric encryption allows the submission to the encryption oracle several times. In fact, for uniformity, we allow it for the public case too, as IND-CCA2 is equivalent for the case of multiple submissions to encryption oracles. Similarly, we will also define derivation with oracle accessibility for KDM-CCA2 encryptions, which are a bit more tricky. But encryption oracles using which keys? The answer is, keys generated by the honest agents during the execution. We will use the notation $\rhd^{\mathcal{O}}$ for such derivability with $\mathcal{O}$ being either aic2, sic2, akc2 or skc2, depending on whether we want asymmetric or symmetric IND-CCA2 oracles or asymmetric or symmetric KDM-CCA2 oracles.

Our next innovation is key usability for the case when keys are sent around. More precisely, to match the notation of derivability, we consider *key corruption*, which is the negation of key usability. We use the notation $x_1, ..., x_n \blacktriangleright^{\mathcal{O}} K$, where $\mathcal{O}$ again indexes whether we are talking about IND-CCA2, KDM-CCA2, symmetric or asymmetric encryption. The intuitive meaning being that K is corrupted by the messages $x_1, ..., x_n$ with access to the given oracles. For example, clearly, $K, x_2 \blacktriangleright^{\mathcal{O}} K$. Or, $\{K\}_{K'}, K' \blacktriangleright^{\mathcal{O}} K$. Or, if $x_1, ..., x_n \rhd^{\mathcal{O}} K$, then $x_1, ..., x_n \blacktriangleright^{\mathcal{O}} K$. But, presumably, if x_1 is just the first half of K, then $x_1 \blacktriangleright^{\mathcal{O}} K$ may still hold, while $x_1 \rhd^{\mathcal{O}} K$ does not. That is, while $x_1, ..., x_n \rhd^{\mathcal{O}} K$ clearly implies $x_1, ..., x_n \blacktriangleright^{\mathcal{O}} K$, the other way is not necessarily true. Nevertheless the two properties, as we will see, behave very similarly, so we chose similar notation for them. We also consider key usability for INT-CTXT *unforgeability*.

A. Terms and Frames

Terms are built out of a set of function symbols $\mathcal{F}$ that contains an unbounded set of names $\mathcal{N}$ and an unbounded set of handles $\mathcal{H}$. Names and handles are zero-arity function symbols. We will use names to denote items honestly generated by agents, while handles will denote inputs of the adversary. Let $\mathcal{X}$ be an unbounded set of input variables (not the same as first-order variables). A ground term is a term without variables. *Frames* are sequences of terms together with name binders: a frame ϕ can be written $(\nu\overline{n}).p_1 \mapsto t_1, \ldots, p_n \mapsto t_n$ where $p_1, \ldots p_n$ are place holders that do not occur in $t_1, \ldots, t_n$ and $\overline{n}$ is a sequence of names. The *variables* of ϕ are the variables of $t_1, \ldots, t_n$.

B. Formulas

Let $\mathcal{P}$ be a set of predicate symbols over tems. We assume here that $\mathcal{P}$ contains the binary predicate $=$ and is used as $t_1 = t_2$, a family of $n+1$ predicates $\rhd_n$, and the following families of predicates meaning various sorts of derivability:

- $t_1, ..., t_n \rhd_n^{\text{aic2}} t$ for derivability of the rhs from the lhs with access to IND-CCA2 oracles in asymmetric case
- $t_1, ..., t_n \rhd_n^{\text{sic2}} t$ for derivability of the rhs from the lhs with access to IND-CCA2 oracles in symmetric case
- $t_1, ..., t_n \rhd_n^{\text{akc2}} t$ for derivability of the rhs from the lhs with access to KDM-CCA2 oracles in asymmetric case
- $t_1, ..., t_n \rhd_n^{\text{skc2}} t$ for derivability of the rhs from the lhs with access to KDM-CCA2 oracles in symmetric case

and the following *key corruption* predicates:

- $t_1, ..., t_n \blacktriangleright_n^{\text{aic2}} K$ meaning the lhs corrupts secure asymmetric IND-CCA2 encryption with K
- $t_1, ..., t_n \blacktriangleright_n^{\text{sic2}} K$ meaning the lhs corrupts secure symmetric IND-CCA2 encryption with K
- $t_1, ..., t_n \blacktriangleright_n^{\text{akc2}} K$ meaning the lhs corrupts K with access to KDM-CCA2 oracles in asymmetric case
- $t_1, ..., t_n \blacktriangleright_n^{\text{skc2}} K$ meaning the lhs corrupts K with access to KDM-CCA2 oracles in symmetric case
- $t_1, ..., t_n \blacktriangleright_n^{\text{ic}} K$ meaning the lhs corrupts INT-CTXT unforgeability of encryptions with K.

For statements that are valid for all derivability or corruption, we will just use $\rhd_n^{\mathcal{O}}$ and $\blacktriangleright_n^{\mathcal{O}}$. We will drop the index n and just write $t_1, ..., t_n \rhd^{\mathcal{O}} t$ and $t_1, ..., t_n \blacktriangleright^{\mathcal{O}} K$.

As for the symbolic interpretation of the predicates (including $=$), we allow any *that does not contradict our axioms, which we will introduce later.*

Let $\mathcal{M}$ be any first-order structure that interprets the function and predicate symbols of the logic. We denote by $D_{\mathcal{M}}$ the domain of interpretation, and by $\rhd_{\mathcal{M}}^{\mathcal{O}}$, $\blacktriangleright_{\mathcal{M}}^{\mathcal{O}}$ and $=_{\mathcal{M}}$ the relations on $D_{\mathcal{M}}$ interpreting $\rhd^{\mathcal{O}}$, $\blacktriangleright^{\mathcal{O}}$, and $=$ respectively. Given an assigment σ of elements in $D_{\mathcal{M}}$ to the free variables of term t, we write $[\![t]\!]_{\mathcal{M}}^{\sigma}$ for the interpretation of $t\sigma$ in $\mathcal{M}$ ($[\![\_]\!]_{\mathcal{M}}^{\sigma}$ is the unique extension of σ into a homomorphism of $\mathcal{F}$-algebras). For any first order structure $\mathcal{M}$ over the functions $\mathcal{F}$ and predicates $\mathcal{P}$, the satisfaction relation $\mathcal{M}, \sigma \models \theta$, where σ is an assignment of the free variables of θ in the domain of $\mathcal{M}$, is defined as usual in first-order logic.

In this paper, we will use the notation $\{x\}_{dK}^{R}$ and $dec(y, dK)$ for both symmetric and asymmetric encryptions with random input R, where in the symmetric case, $eK = dK = K$. We use $\{\!|x|\!\}_{K}^{R}$ and $sdec(y, K)$ for symmetric encryption and decryption only.

C. Execution of a Protocol

Definition II.1. A *symbolic state* of the network consists of:

- a control state $q \in Q$ together with a sequence of names generated so far, $n_1, \ldots, n_k$
- a sequence constants called *handles* $h_1, \ldots, h_n$ (recording the attacker's inputs)
- a ground frame ϕ (the agents outputs)
- a set of formulas Θ (all conditions that must be satisfied in order to reach the state).

A *symbolic transition sequence* of a protocol Π is a sequence

$$(q_0(\overline{n_0}), \emptyset, \phi_0, \emptyset) \to \ldots \to (q_m(\overline{n_m}), \langle h_1, \ldots, h_m\rangle, \phi_m, \Theta_m)$$

if, for every $m-1 \geq i \geq 0$, there is a transition rule

$$(q_i(\overline{\alpha_i}), q_{i+1}(\overline{\alpha_{i+1}}), \langle x_1, \ldots, x_i\rangle, x, \psi, s)$$

such that $\overline{n} = \overline{\alpha_{i+1}} \setminus \overline{\alpha_i}$, $\phi_{i+1} = (\nu\overline{n}).(\phi_i \cdot p \mapsto s\rho_i\sigma_{i+1})$, $\overline{n_{i+1}} = \overline{n_i} \uplus \overline{n}$, $\Theta_{i+1} = \Theta_i \cup \{\phi_i \rhd h_{i+1}, \psi\rho_i\sigma_{i+1}\}$ where $\sigma_i = \{x_1 \mapsto h_1, \ldots, x_i \mapsto h_i\}$ and ρ_i is a renaming of the sequence $\overline{\alpha_i}$ into the sequence $\overline{n_i}$. We assume a renaming that ensures the freshness of the names $\overline{n}$: $\overline{n} \cap \overline{n_i} = \emptyset$.

Definition II.2. Given an interpretation $\mathcal{M}$, a transition sequence of Π

$$(q_0(\overline{n_0}), \emptyset, \phi_0, \emptyset) \to \ldots \to (q_m(\overline{n_m}), \langle h_1, \ldots, h_m\rangle, \phi_m, \Theta_m)$$

is *valid w.r.t.* $\mathcal{M}$ if, for every $m-1 \geq i \geq 0$, $\mathcal{M} \models \Theta_{i+1}$.

D. Satisfaction of Predicates, Constraints and First Order Formulas in Executions

$\mathcal{M}$ modeled, among others, the predicate $t_1, \ldots, t_n \rhd t$. In *executions* however, instead of this predicate, we consider a predicate that we write as $\hat{\phi}, t_1, \ldots, t_n \rhd t$. This is also an $n+1$-arity predicate. $\hat{\phi}$ is just a symbol, not an argument, and it represents the frame containing the messages that protocol agents sent out, that is, the information available from the protocol to the adversary. $=$ refers to equality up to negligible probability, and $\rhd$ means that the adversary is able to compute (with a PPT algorithm) the right side from the left. Similarly, we will use $\hat{\phi}, t_1, \ldots, t_n \rhd^{\mathcal{O}} t$, the meaning is derivability with the help of some oracles indexed by $\mathcal{O}$. We also use another predicate, $W(x)$, which just tells if x is the name of an agent. We also use a number of different *constraints*: $\mathsf{Handle}(h)$ means h is a handle, $\mathsf{RanGen}(x)$ means that x was honestly, randomly generated (i.e. appearing under ν in the frame); $x \sqsubseteq \hat{\phi}$ means that x was subterm of a message sent out by an agent (i.e. listed in the frame ϕ), $x \sqsubseteq \vec{x}$ means x is subterm of $\vec{x}$. $dK \sqsubseteq_{d} \hat{\phi}$ means dK occurs somewhere other than in a decryption position $dec(\ ,dK)$ in ϕ, and $dK \sqsubseteq_{d} \vec{x}$ is analogous (dK may also occur under a decryption, but it has to occur elsewhere too). Similarly, let $K \sqsubseteq_{ed} \hat{\phi}$ mean that symmetric key K occurs somewhere other than in an encryption or decryption position in ϕ, and $K \sqsubseteq_{ed} \vec{x}$ is analogous (K may also occur under an encryption or decryption, but it has to occur elsewhere too). Let us introduce the following abbreviations:

- $x \sqsubseteq \hat{\phi}, \vec{x} \ \equiv\ x \sqsubseteq \hat{\phi} \vee x \sqsubseteq \vec{x}$
- $\mathsf{fresh}(x; \hat{\phi}, \vec{x}) \ \equiv\ \mathsf{RanGen}(x) \wedge x \not\sqsubseteq \hat{\phi}, \vec{x}$
- $\mathsf{keyfresh}(K; \hat{\phi}, \vec{x})$
  - For asymmetric key:
    $\mathsf{keyfresh}(K; \hat{\phi}, \vec{x}) \ \equiv\ \mathsf{RanGen}(K) \wedge dK \not\sqsubseteq_{d} \hat{\phi}, \vec{x}$
  - For symmetric key:
    $\mathsf{keyfresh}(K; \hat{\phi}, \vec{x}) \ \equiv\ \mathsf{RanGen}(K) \wedge K \not\sqsubseteq_{ed} \hat{\phi}, \vec{x}$
- $\vec{x} \preccurlyeq \hat{\phi} \ \equiv\ h \sqsubseteq \vec{x} \wedge \mathsf{Handle}(h) \to \hat{\phi} \rhd h$

If $\mathcal{M}$ is a first-order model as before, satisfaction of predicates and constraints in a *symbolic* execution is defined as:

- Interpretation of predicates by $\mathcal{M}, \sigma, \langle t_1, \ldots, t_m\rangle, (n_1, \ldots, n_k)$, where σ is a substitution as above, $t_1, \ldots, t_m$ are closed terms, and $n_1, \ldots, n_k$ are names: (note the interpretation depends on $\mathcal{M}$) is defined as follows
  - $\mathcal{M}, \sigma, \langle t_1, \ldots, t_m\rangle, (n_1, \ldots, n_k) \models t = t'$ if $\mathcal{M}, \sigma \models t = t'$
  - $\mathcal{M}, \sigma, \langle t_1, \ldots, t_m\rangle, \overline{n} \models \hat{\phi}, s_1, \ldots, s_n \rhd t$ if $\mathcal{M}, \sigma \models s_1, \ldots, s_n, t_1, \ldots, t_m \rhd^{\mathcal{O}} t$.
  - $\mathcal{M}, \sigma, \langle t_1, \ldots, t_m\rangle, \overline{n} \models \hat{\phi}, s_1, \ldots, s_n \blacktriangleright t$ if $\mathcal{M}, \sigma \models s_1, \ldots, s_n, t_1, \ldots, t_m \blacktriangleright^{\mathcal{O}} t$.
  - $\mathcal{M}, \sigma, \langle t_1, \ldots, t_m\rangle, \overline{n} \models W(x)$ if $\mathcal{M}, \sigma \models W(x)$
- Interpretations of constraints by $\mathcal{M}, \sigma, \langle t_1, \ldots, t_m\rangle, (n_1, \ldots, n_k)$, where σ is a substitution as above, $t_1, \ldots, t_m$ are closed terms, and $n_1, \ldots, n_k$ are names: (do not depend on the model $\mathcal{M}$):
  - $\mathsf{Handle}(h)$ for h closed term:
    $\mathcal{M}, \sigma, \langle t_1, \ldots, t_m\rangle, (n_1, \ldots, n_k) \models \mathsf{Handle}(h)$ if $h \in \mathcal{H}$.
  - $\mathsf{RanGen}(s)$ for s closed term:
    $\mathcal{M}, \sigma, \langle t_1, \ldots, t_m\rangle, (n_1, \ldots, n_k) \models \mathsf{RanGen}(s)$ if $s \in \mathcal{N}$ and $\mathcal{M}, \sigma \models s = n_1 \vee \ldots \vee s = n_k$.
  - $t \sqsubseteq \hat{\phi}$, where t is closed term:
    $\mathcal{M}, \sigma, \langle t_1, \ldots, t_m\rangle, \overline{n} \models t \sqsubseteq \hat{\phi}$ if t is a subterm of some t_i
  - $t \sqsubseteq s_1, \ldots, s_n$, where $s_1, \ldots, s_n$, t are closed terms:
    $\mathcal{M}, \sigma, \langle t_1, \ldots, t_m\rangle, \overline{n} \models t \sqsubseteq s_1, \ldots, s_n$ if t is a subterm of some s_i
- Interpretation of any FOL formula in which there are no free variables under constraints by

$\mathcal{M}, \sigma, \langle t_1, \ldots, t_m\rangle, (n_1, \ldots, n_k)$ where σ is a substitution as above, is defined recursively as:

- Interpretations of $\theta_1 \wedge \theta_2$, $\theta_1 \vee \theta_2$, and $\neg\theta$ are defined as usual in FOL
- If x is not under a constraint in θ, interpretations of $\forall x\theta$ and $\exists x\theta$ are defined as usual in FOL.
- If x occurs under a constraint in θ, then
  - $\mathcal{M}, \sigma, \langle t_1, \ldots, t_n\rangle, (n_1, \ldots, n_k) \models \forall x\theta$ iff for every ground term t, $\mathcal{M}, \sigma, \langle t_1, \ldots, t_n\rangle, (n_1, \ldots, n_k) \models \theta\{x \mapsto t\}$
  - $\mathcal{M}, \sigma, \langle t_1, \ldots, t_n\rangle, (n_1, \ldots, n_k) \models \exists x\theta$ iff there is a ground term t, $\mathcal{M}, \sigma, \langle t_1, \ldots, t_n\rangle, (n_1, \ldots, n_k) \models \theta\{x \mapsto t\}$

- Satisfaction at step m: $\mathcal{M}, (q, \langle h_1, \ldots, h_m\rangle, \overline{n}, \phi_m, \Theta) \models \theta$ iff $\mathcal{M}, \phi_m, \overline{n} \models \theta$.

III. Computational Execution and Interpretation of Formulas

We now summarize the computational semantics. Proofs of Theorems 1 and 2 are in [5], however, as it is mentioned there, they are actually rather easy consequences of Fitting's embedding of classical logic into S4 [21], a very interesting relationship that we plan to detail elsewhere.

A. Computational Execution

We just briefly summarize the computational execution here, for more details, consult [6].

We consider a familly computational algebras, parametrized by a security parameter η, in which each function symbol is interpreted as a polynomially computable function on bitstrings (that may return an error message). Given then a sample τ of names (for every name n, its interpretation is a bitstring $\tau(n)$), every ground term t is interpreted as a bitstring $[\![t]\!]_\tau$ in such a way that $[\![\_]\!]_\tau$ is a homomorphism of $\mathcal{F}$-algebras. More generally, if σ is an assignment of the variables of t to bitstrings $[\![t]\!]_\tau^\sigma$ is the (unique) extension of τ (on names) and σ (on variables) as a homomorphism of $\mathcal{F}$-algebras.

Definition III.1. Given a set of transition rules, a *computational state* consists of

- a symbolic state s (that is itself a tuple $q(\overline{n}, \overline{h}, \phi, \Theta)$)
- a sequence of bitstrings $\langle b_1, \ldots, b_m\rangle$ (attacker outputs)
- a sequence $\langle b'_1, \ldots, b'_n\rangle$ of bitstrings (agents' outputs)
- the configuration γ of the attacker.

Definition III.2. Given a PPT interactive Turing machine $\mathcal{M}$ and a sample τ, a sequence of transitions $(s_0, \emptyset, \vec{b'_0}, \gamma_0) \to \ldots \to (s_m, \langle b_1, \ldots, b_m\rangle, \langle b'_1, \ldots, b'_m\rangle, \gamma_m)$ is *(computationally) valid w.r.t.* $\mathcal{M}$ *and* τ if

- $s_0 \to \cdots \to s_m$ is a transition sequence of the protocol
- for all $i = 0, \ldots m-1$, $s_i = (q_i(\overline{n_i}), \overline{h_i}, \phi_i, \Theta_i)$, $\phi_{i+1} = (\nu\overline{n}).\phi_i \cdot u_i$, $[\![u_i]\!]_\tau = b'_{i+1}$
- for every $i = 0, \ldots, m-1$, there is a configuration γ'_i of the machine $\mathcal{M}$ such that $\gamma_i \rhd_M^* \gamma'_i \rhd_M^* \gamma_{i+1}$ and γ'_i is in a sending state, the sending tape containing b_{i+1}, γ_{i+1} is in a receiving state, the receiving tape containing b'_{i+1}
- for every $i = 0, \ldots, m-1$, $\tau, \{x_1 \mapsto b_1, \ldots, x \mapsto b_{i+1}\} \models^c \Theta_{i+1}$.

B. Computational satisfaction of formulas

We introduce the computational interpretation of the original predicates, $=$ and $\rhd$ here and the recursive semantics of compound formulas. Interpretations of the new predicates are presented later.

Let $\mathcal{M}$ be an interactive PPT Turing machine with a special challenge control state q_c. We may regard this machine as an attacker, who moves to the state q_c when he thinks that he is ready to break the security property. As usual, the machine takes the security parameter 1^η as an initial input. Since such an execution is probabilistic, for each security parameter η, there is an underlying finite probability space, (Ω^η, p^η), the elements of which are denoted by ω^η. Without loss of generality, we can assume that there is a common random tape from which each participant is reading random bits. Each ω^η is one particular random string, and $\tau(\omega^\eta)$ is the assignment of all fixed bit string evaluations $\tau(\overline{n})$ of names given for ω^η. For a given n name, we just use simply $n(\omega^\eta)$ for the bit string $\tau(\omega^\eta)(n)$.

By a *non-negligible set of coins* S, we mean $S = (S^\eta)_{\eta\in\mathbb{N}}$, where for all $\eta \in \mathbb{N}$, $S^\eta \subseteq \Omega^\eta$, and $p^\eta(S^\eta)$ is non-negligible function of η. For $S_1 = (S_1^\eta)_{\eta\in\mathbb{N}}$ and $S_2 = (S_2^\eta)_{\eta\in\mathbb{N}}$, we use the notation $S_1 \subseteq S_2$ if for all $\eta \in \mathbb{N}$, $S_1^\eta \subseteq S_2^\eta$. In what follows, S is any such non-negligible set.

We recall the interpretations of $=$ and $\rhd$ from [6]: Let σ be a sequence of PT machines (e.g. one for each free variable x_i of θ): $\mathcal{A}_{x_1}, \ldots, \mathcal{A}_{x_n}$. Inputs of these are the the pairs (η, ω^η). For example, and $\mathcal{A}_x$ can be the evaluation of any name (in which case $\mathcal{A}_x(\eta, \omega^\eta) = n(\omega^\eta)$, or any value for a handle computed by the adversary, or some more complex object. Let $\sigma(\omega^\eta)$ denote the assignments $x_1 \mapsto \mathcal{A}_{x_1}(\eta, \omega^\eta), \ldots, x_n \mapsto \mathcal{A}_{x_n}(\eta, \omega^\eta)$. For a statement $statement(\eta, \omega^\eta)$, and a fixed S, instead of "for all $\eta \in \mathbb{N}$ and $\omega^\eta \in S^\eta, statement(\eta, \omega^\eta)$", we will simply write "for all $\omega \in S, statement(\omega)$.

- For the equality predicate, $\mathcal{M}, \Pi, S, \sigma \models^c t_1 = t_2$ iff there is an overwhelming subset $S' \subseteq S$ such that, for all $\omega \in S'$, $[\![t_1]\!]_{\tau(\omega)}^{\sigma(\omega)} = [\![t_2]\!]_{\tau(\omega)}^{\sigma(\omega)}$.
- For the deducibility predicate, $\mathcal{M}, \Pi, S, \sigma \models^c \hat{\phi}, t_1, \ldots, t_n \rhd t$ if for all non-negligible $S' \subseteq S$, there is a non-negligible $S'' \subseteq S'$ and a PT Turing machine $\mathcal{A}$ such that for all $\omega \in S''$, $\mathcal{A}([\![\phi_m(\omega)]\!]_{\tau(\omega)}^{\sigma(\omega)}, [\![t_1]\!]_{\tau(\omega)}^{\sigma(\omega)}, \ldots, [\![t_n]\!]_{\tau(\omega)}^{\sigma(\omega)}, a(\omega), r(\omega)) = [\![t]\!]_{\tau(\omega)}^{\sigma(\omega)}$ where $m(\omega)$ is the step at which $\mathcal{M}$ reached the challenge state, $a(\omega)$ stands for the protocol adversary's

output and $r(\omega)$ is some random input from the random string.

Compound formulas are computed by the following rules. Note, that this interpretation of disjunction and existential quantifiers are defined differently from [6], and so far has only been published in eprint [5].

- $\mathcal{M}, \Pi, S, \sigma \models^c \theta_1 \wedge \theta_2$ iff $\mathcal{M}, \Pi, S, \sigma \models^c \theta_1$ and $\mathcal{M}, \Pi, S, \sigma \models^c \theta_2$.
- $\mathcal{M}, \Pi, S, \sigma \models^c \theta_1 \vee \theta_2$ iff for any $S' \subseteq S$ non-negligible, there is a $S'' \subseteq S$ non-negligible such that either $\mathcal{M}, \Pi, S'', \sigma \models^c \theta_1$ or $\mathcal{M}, \Pi, S'', \sigma \models^c \theta_2$.
- $\mathcal{M}, \Pi, S, \sigma \models^c \theta_1 \to \theta_2$ iff for any $S' \subseteq S$ non-negligible, $\mathcal{M}, \Pi, S', \sigma \models^c \theta_1$ implies $\mathcal{M}, \Pi, S', \sigma \models^c \theta_2$
- $\mathcal{M}, \Pi, S, \sigma \models^c \neg\theta$ iff for any S', $\mathcal{M}, \Pi, S', \sigma \not\models^c \theta$
- $\mathcal{M}, \Pi, S, \sigma \models^c \exists x.\theta$ iff for any $S' \subseteq S$ non-negligible, there is a $S'' \subseteq S$ non-negligible and a PT machine $\mathcal{A}_x$ such that $\mathcal{M}, \Pi, S'', \sigma, \mathcal{A}_x \models^c \theta$
- $\mathcal{M}, \Pi, S, \sigma \models^c \forall x.\theta$ iff for any probabilistic polynomial time machine $\mathcal{A}_x$, $\mathcal{M}, \Pi, S, \sigma, \mathcal{A}_x \models^c \theta$

$\mathcal{M}, \Pi \models^c \theta$ iff $\mathcal{M}, \Pi, \Omega \models^c \theta$ and $\Pi \models^c \theta$ if $\mathcal{M}, \Pi \models^c \theta$ for every $\mathcal{M}$ and q_c.

Despite that semantics of the compound formulas is not as usual in first-order logic, as a consequence of Fitting's embedding [21] of classical logic into S4 (explained in [5]) we have the next theorem. In particular, De-Morgan identities, double negation *etc*. hold.

Theorem III.3 (Consequence of Fitting's Embedding of Classical Logic into S4)**.** *With the above semantics, first-order deduction rules are sound.*

And then, as a consequence, the following trace mapping and soundness theorems hold. Proofs are in [6] and [5].

Theorem III.4 (Trace Mapping)**.** *Let Π be a protocol, $s_1 \to \ldots \to s_m$ be a symbolic transition sequence of Π and $\mathcal{M}$ be a probabilistic polynomial time interactive Turing machine. If there is a non-negligible set of coins S such that, for any $\tau \in S$, there is a sequence of transitions $(s_0, \vec{b_0}, \vec{b'_0}, \gamma_0) \to \cdots \to (s_m, \vec{b_m}, \vec{b'_m}, \gamma_m)$ that is computationally valid w.r.t. $\mathcal{M}$, τ and γ_m is in the challenge state q_c, then for any formula θ, $\mathcal{M}, \Pi, S \models^c \theta$ implies there is a symbolic model $\mathcal{S}$ such that $s_0 \to \cdots \to s_m$ is a valid symbolic execution w.r.t. $\mathcal{S}$ and $\mathcal{S} \models \theta$.*

Theorem III.5 (Soundness)**.** *For a bounded number of sessions, if there is a computational attack, there is also a symbolic attack.*

C. Axioms

We recall the core axioms presented in [9]. As usual, unquantified variables are universally quantified. Unless noted otherwise, they are always sound. Naturally, these axioms do not depend at all on any security assumptions such as CCA2. We remind the reader, that $\mathsf{fresh}(x; \hat{\phi}, \vec{x})$ means that x is generated freshly, independently of the past of the protocol and of $\vec{x}$, and $\vec{x} \preccurlyeq \hat{\phi}$ means that all handles in $\vec{x}$ can be computed from the past of the protocol; that is, $\vec{x}$ cannot contain information from the future.

- **Equality is a Congruence.**
  - $x = x$, and the substitutability (congruence) property of equal terms holds for $=$ and $\rhd$.
- **Core Axioms for the Derivability Predicate.**
  - Self derivability: $\hat{\phi}, \vec{x}, x \rhd x$
  - Increasing capabilities: $\hat{\phi}, \vec{x} \rhd y \longrightarrow \hat{\phi}, \vec{x}, x \rhd y$
  - Commutativity: If $\vec{x}'$ is a permutation of $\vec{x}$, then $\hat{\phi}, \vec{x} \rhd y \longrightarrow \hat{\phi}, \vec{x}' \rhd y$
  - Transitivity of derivability: $\hat{\phi}, \vec{x} \rhd \vec{y} \wedge \hat{\phi}, \vec{x}, \vec{y} \rhd \vec{z} \longrightarrow \hat{\phi}, \vec{x} \rhd \vec{z}$
  - Functions are derivable: $\hat{\phi}, \vec{x} \rhd f(\vec{x})$
    This axiom is sound as long as functions are interpreted as PT computable algorithms.
- **Axioms for Freshly Generated Items.**
  - No telepathy: $\mathsf{fresh}(x; \hat{\phi}) \longrightarrow \hat{\phi} \not\rhd x$
    This axiom is sound as long as $\mathsf{RanGen}()$ items are generated so that they can only be guessed with negligible probability.
  - Fresh items do not help to compute: $\mathsf{fresh}(x; \hat{\phi}, \vec{x}, y) \wedge \vec{x}, y \preccurlyeq \hat{\phi} \wedge \hat{\phi}, \vec{x}, x \rhd y \longrightarrow \hat{\phi}, \vec{x} \rhd y$
- **Equations for the fixed function symbols.** For example, for symmetric encryption $\mathit{sdec}(\{\!|x|\!\}_K^R, K) = x$, and for pairing, $\pi_1(\langle x, y\rangle) = x$; $\pi_2(\langle x, y\rangle) = y$. Function of error is error $f(..., \bot, ...) = \bot$, etc

IV. Semantics of Derivability with Oracles and Axioms

A. Semantics of Derivability with Oracles

Here we define the semantics of our newly added derivability predicates.

Definition IV.1. Semantics of Derivability with Oracles: Let $\mathcal{M}, \Pi, S, \sigma$ be as before.

- $\mathcal{M}, \Pi, S, \sigma \models^c \hat{\phi}, \vec{x} \rhd^{\text{aic2}} x$ (or $\mathcal{M}, \Pi, S, \sigma \models^c \hat{\phi}, \vec{x} \rhd^{\text{sic2}} x$) if for all non-negligible $S' \subseteq S$, there is a non-negligible $S'' \subseteq S'$ and a PT Turing machine $\mathcal{A}^{\mathcal{O}^{\text{aic2}}}$ (or $\mathcal{A}^{\mathcal{O}^{\text{sic2}}}$) with access to oracles $\mathcal{O}^{\text{aic2}}$ (or $\mathcal{O}^{\text{sic2}}$) such that for all $\omega \in S''$,
$$\mathcal{A}^{\mathcal{O}^{\text{aic2}}}([\![\phi_m]\!]_{\tau(\omega)}^{\sigma(\omega)}, [\![\vec{x}]\!]_{\tau(\omega)}^{\sigma(\omega)}, a(\omega), r(\omega)) = [\![x]\!]_{\tau(\omega)}^{\sigma(\omega)}$$
(or $\mathcal{A}^{\mathcal{O}^{\text{sic2}}}([\![\phi_m]\!]_{\tau(\omega)}^{\sigma(\omega)}, [\![\vec{x}]\!]_{\tau(\omega)}^{\sigma(\omega)}, a(\omega), r(\omega)) = [\![x]\!]_{\tau(\omega)}^{\sigma(\omega)}$) where $a(\omega)$ stands for the protocol adversary's output and $r(\omega)$ is some random input from the random string. The oracles $\mathcal{O}^{\text{aic2}}$ (or $\mathcal{O}^{\text{sic2}}$) are the same as those for multi-user IND-CCA2 encryption and decryption oracles. The oracles have all the honest keys from $\hat{\phi}$,

and decrypt everything except those bit strings that are results of honest encryptions in $\hat{\phi}, \vec{x}$.
We can shorten this as

$$\mathcal{M}, \Pi, S'', \sigma \| \models^c \mathcal{A}^{\mathcal{O}}(\hat{\phi}, \vec{x}) = x,$$

implicitely assuming the algorithm has access to the protocol adversary's knowledge and to random bits.

- $\mathcal{M}, \Pi, S, \sigma \models^c \hat{\phi}, \vec{x} \rhd^{\text{akc2}} x$ (or $\mathcal{M}, \Pi, S, \sigma \models^c \hat{\phi}, \vec{x} \rhd^{\text{skc2}} x$) if for all non-negligible $S' \subseteq S$, there is a non-negligible $S'' \subseteq S'$ and a $\vec{y}$ with $\mathcal{M}, \Pi, S'', \sigma \models^c \vec{y} \preccurlyeq \hat{\phi}$, and a $\vec{K}$ list of honest keys (i.e. $\mathsf{RanGen}(K_i)$ for all i), such that the lists $\vec{y}$ and $\vec{K}$ have the same length, and a PT Turing machine $\mathcal{A}^{\mathcal{O}^{\text{akc2}}}$ (or $\mathcal{A}^{\mathcal{O}^{\text{skc2}}}$) with access to oracles $\mathcal{O}^{\text{akc2}}$ (or $\mathcal{O}^{\text{skc2}}$) such that, denoting by $\{\vec{y}\}_{e\vec{K}}$ the list of encryptions of $\vec{y}$ with the corresponding honest encryption keys, for all $\omega \in S''$,

$$\mathcal{A}^{\mathcal{O}^{\text{akc2}}}(\llbracket\phi_m\rrbracket_{\tau(\omega)}^{\sigma(\omega)}, \llbracket\vec{x}\rrbracket_{\tau(\omega)}^{\sigma(\omega)}, \llbracket\{\vec{y}\}_{e\vec{K}}\rrbracket_{\tau(\omega)}^{\sigma(\omega)}, a(\omega), r(\omega)) = \llbracket x\rrbracket_{\tau(\omega)}^{\sigma(\omega)}$$

(or $\mathcal{A}^{\mathcal{O}^{\text{skc2}}}(\llbracket\phi_m\rrbracket_{\tau(\omega)}^{\sigma(\omega)}, \llbracket\vec{x}\rrbracket_{\tau(\omega)}^{\sigma(\omega)}, \llbracket\{\!|\vec{y}|\!\}_{\vec{K}}\rrbracket_{\tau(\omega)}^{\sigma(\omega)}, a(\omega), r(\omega)) = \llbracket x\rrbracket_{\tau(\omega)}^{\sigma(\omega)}$) where $a(\omega)$ stands for the protocol adversary's output and $r(\omega)$ is some random input from the random string. The oracles $\mathcal{O}^{\text{akc2}}$ (or $\mathcal{O}^{\text{skc2}}$) are the same as those for multi-user KDM-CCA2 encryption and decryption oracles. The decryption oracles have all the honest keys from $\hat{\phi}$, and decrypt everything except those bit strings that are results of honest encryptions in $\hat{\phi}, \vec{x}, \{\vec{y}\}_{e\vec{K}}$. We can shorten this as

$$\mathcal{M}, \Pi, S'', \sigma \| \models^c \mathcal{A}^{\mathcal{O}}(\hat{\phi}, \vec{x}, \{\vec{y}\}_{e\vec{K}}) = x,$$

We also assume, that the oracles use randomness from the same base Ω to avoid the technical difficulty of extending the random space.

Note that in the KDM case, besides allowing $\mathcal{A}$ to use the usual KDM oracles, a list of encryptions, $\{\vec{y}\}_{e\vec{K}}$ is also provided to the algorithm. For example, $\mathcal{A}$ is allowed to use the encryption of a nonce even if he does not know the nonce. While through KDM oracles, it is possible for $\mathcal{A}$ to receive encryptions of keys that $\mathcal{A}$ does not know, the KDM oracles do not allow requests for encryptions of other unknown items. This is necessary for receiving nice axioms for the KDM case, and we explain the reason there.

B. Axioms for Derivability with Oracles

The following axioms (except for the second and last entry of the core axioms for derivability predicates) are very similar to the ones in the previous section, and are just as trivial. The second entry of the core axioms for derivability predicates with oracles is also trivially computationally sound as if something is derivable with the help of some oracles, then it is also derivable with more powerful oracles. We explain the last entry when we introduce it:

- **Core Axioms for the Derivability Predicate with Oracles.**
  - Equal terms are substitutable on the right hand side of $\rhd^{\mathcal{O}}$. It is not true on the left hand side, because the usability of the decryption oracles depends on the encryptions on the left hand side.
  - More oracles help more: If the oracles of $\mathcal{O}$ are more powerful then the oracles of $\mathcal{O}'$, then $\hat{\phi}, \vec{x} \rhd^{\mathcal{O}'} x \longrightarrow \hat{\phi}, \vec{x} \rhd^{\mathcal{O}} x$.
    In particular, $\hat{\phi}, \vec{x} \rhd x \longrightarrow \hat{\phi}, \vec{x} \rhd^{\mathcal{O}} x$. But KDM oracles are also more powerful than IND oracles.
  - Increasing capabilities: $\hat{\phi}, \vec{x} \rhd^{\mathcal{O}} y \longrightarrow \hat{\phi}, \vec{x}, x \rhd^{\mathcal{O}} y$
  - Commutativity: If $\vec{x}'$ is a permutation of $\vec{x}$, then $\hat{\phi}, \vec{x} \rhd^{\mathcal{O}} y \longrightarrow \hat{\phi}, \vec{x}' \rhd^{\mathcal{O}} y$
  - Transitivity:
    $\hat{\phi}, \vec{x} \rhd^{\mathcal{O}} \vec{y} \wedge \hat{\phi}, \vec{x}, \vec{y} \rhd^{\mathcal{O}} \vec{z} \longrightarrow \hat{\phi}, \vec{x} \rhd^{\mathcal{O}} \vec{z}$
  - Decryptions of adversarial ciphers do not help: If $\mathcal{O}$ is either IND or KDM CCA2, either symmetric or asymmetric, then

$$\begin{aligned} &\mathsf{RanGen}(K) \wedge \hat{\phi}, \vec{x} \rhd^{\mathcal{O}} y \wedge \hat{\phi}, \vec{x}, dec(y, dK) \rhd^{\mathcal{O}} z \\ &\wedge\ \forall x R(y = \{x\}_{eK}^R \to \{x\}_{eK}^R \not\sqsubseteq \hat{\phi}, \vec{x}) \\ &\longrightarrow\ \hat{\phi}, \vec{x} \rhd^{\mathcal{O}} z \end{aligned}$$

Here, in case of symmetric encryption, $dK = eK = K$. This expresses the fact that for the derivation in $\hat{\phi}, \vec{x} \rhd^{\mathcal{O}} y$, the algorithm can request the decryption oracles as long as the submitted message is not an honest encryption of $\hat{\phi}, \vec{x}$. So the decryption of a y cannot help as long as the adversary can compute y and as long as y differs from any encryption, because the decryption can also be received from the decryption oracle. A little more precisely, if $\mathcal{M}, \Pi, S, \sigma$ satisfies the premise, then it is easy to show by the rules of the semantics that for all $S' \subseteq S$, there is a $S'' \subseteq S'$ and a $\mathcal{A}_1$ and $\mathcal{A}_2$ algorithems with access to $\mathcal{O}$ such that $\mathcal{A}_1$ computes y from $\hat{\phi}, \vec{x}$ and $\mathcal{A}_2$ computes z from $\hat{\phi}, \vec{x}, dec(y, dK)$, but y is not an honest encryption from $\hat{\phi}, \vec{x}$. Then, we can combine $\mathcal{A}_1$ and $\mathcal{A}_2$ into a single algorithm accessing the oracles computing z from $\hat{\phi}, \vec{x}$, because to get $dec(y, dK)$ he can just request the decryption oracle. That means $\mathcal{M}, \Pi, S, \sigma$ satisfies the conclusion too.
Note, that this replaces the non-malleability axiom of [9] for the derivability predicate. While in [9], the non-malleability axiom was quite a bit more complex, and the IND-CCA2 property was necessary for the soundness of the axiom, here, soundness of this axiom follows purely from the definition of the semantics of $\rhd^{\mathcal{O}}$, and it is not necessary that the encryption is CCA2 secure. With tiny modifications, it is possible to rewrite the NSL proof presented in [9] for using the $\rhd^{\mathcal{O}}$ predicate and this simpler axiom instead of the $\rhd$ predicate with the non-malleability axiom there.

- **Axioms for Freshly Generated Items.**
  - No telepathy: $\mathsf{fresh}(x;\hat{\phi}) \;\longrightarrow\; \hat{\phi}\not\rhd^{\mathcal{O}} x$ (implies no-telepathy axiom without oracles). This is sound as long as $\mathsf{RanGen}()$ means generation with negligible guessing probability only.
  - Fresh items do not help to compute: $\mathsf{fresh}(x;\hat{\phi},\vec{x},y) \;\wedge\; \vec{x},y \preccurlyeq \hat{\phi} \;\wedge\; \hat{\phi},\vec{x},x \rhd^{\mathcal{O}} y \;\longrightarrow\; \hat{\phi},\vec{x} \rhd^{\mathcal{O}} y$

  These axioms are sound for the same reason as the corresponding ones for $\rhd$ were.

Note that $\hat{\phi},\vec{x},x \rhd^{\mathcal{O}} \vec{x}$ is implied by the more oracles help more axiom and the self-derivability axiom of derivability predicate. Similarly, $\hat{\phi},\vec{x} \rhd^{\mathcal{O}} f(\vec{x})$.

V. Key Usability

A. Semantics of Key Usability for the CCA2 encryptions

The idea of key usability is that a key has been uncorrupted, that is, it can be used for safe encryption. To match the computability predicate, we define the negation of it, that is, key corruption. The intuitive meaning of $\hat{\phi},\vec{x} \blacktriangleright^{\mathcal{O}} K$ is that $\hat{\phi},\vec{x}$ corrupts the key (in the presence of oracles) and it cannot be used for safe encryption any more.

Definition V.1 (Key Corruption of IND-CCA2 Encryption Schemes)**.** We say that $\mathcal{M},\Pi,S,\sigma \models^c \hat{\phi},\vec{x} \blacktriangleright^{\mathcal{O}} K$ if either $\mathcal{M},\Pi,S,\sigma \models^c \vec{x} \not\preccurlyeq \hat{\phi} \vee \neg\mathsf{RanGen}(K)$, or for all $S' \subseteq S''$, there are $S'' \subseteq S'$, PT algorithms $\mathcal{A}_1^{\mathcal{O}}$, $\mathcal{A}_2^{\mathcal{O}}$, and y, R s.t.

- $\mathcal{M},\Pi,S'',\sigma \models^c y \preccurlyeq \hat{\phi}$
- R is generated statistically independently of the interpretations of $\hat{\phi}$, $\vec{x}$, y and $\mathcal{A}_1^{\mathcal{O}}(\hat{\phi},\vec{x})$
- $\mathcal{M},\Pi,S'',\sigma \models^c \mathcal{A}_2^{\mathcal{O}}(\hat{\phi},\vec{x},\{\mathcal{A}_1^{\mathcal{O}}(\hat{\phi},\vec{x})\}_{eK}^R) = y$ but $\mathcal{M},\Pi,S'',\sigma \models^c \mathcal{A}_2^{\mathcal{O}}(\hat{\phi},\vec{x},\{0^{|\mathcal{A}_1^{\mathcal{O}}(\hat{\phi},\vec{x})|}\}_{eK}^R) \neq y$ or v.v.

As before, $\mathcal{M},\Pi,S'',\sigma \models^c \mathcal{A}_2^{\mathcal{O}}(\hat{\phi},\vec{x},\{\mathcal{A}_1^{\mathcal{O}}(\hat{\phi},\vec{x})\}_{eK}^R) = y$ means that for all $\omega \in S''$, the algorithms $\mathcal{A}_1$ and $\mathcal{A}_2$ implied on the semantics of the terms as indicated (and on the output of the protocol adversary and some additional possible random input) give back the semantics of y. That is, $\mathcal{A}_2^{\mathcal{O}}$ computes y with the help of encryption of $\mathcal{A}_1^{\mathcal{O}}(\hat{\phi},\vec{x})$, but is not computed with the encryption of a string of 0's with the same length as $\mathcal{A}_1^{\mathcal{O}}(\hat{\phi},\vec{x})$. This is just another way of saying that the adversary can create a string from the past of the protocol and $\vec{x}$, namely $\mathcal{A}_1^{\mathcal{O}}(\hat{\phi},\vec{x})$, the encryption of which is potentially distinguishable from the encryption of 0's. The adversary uses y for the distinguishing. y may actually not be accessible for the attacker, but if such a y can be potentially be given to the adversary computed from the past of the protocol (i.e. $y \preccurlyeq \hat{\phi}$), then the security of the encryption is broken. y can for example be a secret nonce. The above corruption condition in this case would mean that with the encryption the attacker can compute the secret nonce, but without the encryption it cannot.

The negation of our key corruption notion, that is, our key usability is weaker from that of [20] in the sense that we do not have indistinguishability here, but explicit computability of y. On the other had, it is stronger in the sense that our adversary is allowed to use something potentially unknown to him, namely y for the distinguishing of the correct encryption from the encryption of 0's. We designed this notion so that it results nice axioms in our setting.

Definition V.2 (Key Corruption of KDM-CCA2 Encryption Schemes)**.** We say that $\mathcal{M},\Pi,S,\sigma \models^c \hat{\phi},\vec{x} \blacktriangleright^{\mathcal{O}} K$ if either $\mathcal{M},\Pi,S,\sigma \models^c \vec{x} \not\preccurlyeq \hat{\phi} \vee \neg\mathsf{RanGen}(K)$, or for all $S' \subseteq S''$, there are $S'' \subseteq S'$, a PT algorithm $\mathcal{A}^{\mathcal{O}}$, and x, y, R s.t.

- $\mathcal{M},\Pi,S'',\sigma \models^c x,y \preccurlyeq \hat{\phi} \wedge \mathsf{fresh}(R;\hat{\phi},\vec{x},x,y,K)$
- and $\mathcal{M},\Pi,S'',\sigma \models^c \mathcal{A}^{\mathcal{O}}(\hat{\phi},\vec{x},\{x\}_{eK}^R) = y$ but $\mathcal{M},\Pi,S'',\sigma \models^c \mathcal{A}^{\mathcal{O}}(\hat{\phi},\vec{x},\{0^{|x|}\}_{eK}^R) \neq y$ or v.v.

Note the difference between the IND and KDM cases is that in the latter, x does not have to be computed from $\hat{\phi},\vec{x}$.

B. Axioms for Key Corruption, Secrecy

We now present the axioms for key corruption. First the core axioms. It may be surprising, but soundness of these axioms do not need CCA2 security. All follow purely from the definition of key corruption. Proofs are in Appendix A:

- **Core Axioms for the Key Corruption Predicate.**
  - Equal terms are substitutable on the rhs of $\blacktriangleright^{\mathcal{O}}$.
  - Derivability implies corruption: $\hat{\phi},\vec{x} \rhd^{\mathcal{O}} K \;\longrightarrow\; \hat{\phi},\vec{x} \blacktriangleright^{\mathcal{O}} K$
    If K is computable for the adversary, then it is corrupted. Note, this axiom and the self derivability axiom (from IV-B) imply that $\hat{\phi},\vec{x},K \blacktriangleright^{\mathcal{O}} K$
  - Increasing capabilities for key corruption: $\hat{\phi},\vec{x} \blacktriangleright^{\mathcal{O}} K \;\longrightarrow\; \hat{\phi},\vec{x},x \blacktriangleright^{\mathcal{O}} K$
  - Commutativity: If $\vec{x}'$ is a permutation of $\vec{x}$, then $\hat{\phi},\vec{x} \blacktriangleright^{\mathcal{O}} K \;\longrightarrow\; \hat{\phi},\vec{x}' \blacktriangleright^{\mathcal{O}} K$
  - Transitivity: $\hat{\phi},\vec{x} \rhd^{\mathcal{O}} \vec{y} \;\wedge\; \hat{\phi},\vec{x},\vec{y} \blacktriangleright^{\mathcal{O}} K \;\longrightarrow\; \hat{\phi},\vec{x} \blacktriangleright^{\mathcal{O}} K$
    The intuitive reason is very clear: $\vec{y}$ just contains extra information, that can be computed from $\hat{\phi},\vec{x}$, so it is not actually needed in the corruption. This, and the functions are derivable axiom imply $\hat{\phi},\vec{x},f(\vec{x}) \blacktriangleright^{\mathcal{O}} K \;\longrightarrow\; \hat{\phi},\vec{x} \blacktriangleright^{\mathcal{O}} K$. With the increasing capabilities axiom, we also have $\hat{\phi},f(\vec{x}) \blacktriangleright^{\mathcal{O}} K \;\longrightarrow\; \hat{\phi},\vec{x} \blacktriangleright^{\mathcal{O}} K$. We will refer to these as *function application*.
  - Decryptions of adversarial ciphers do not help: If $\mathcal{O}$ is either IND or KDM CCA2, either symmetric or asymmetric, then

$$\begin{aligned}&\mathsf{RanGen}(K) \;\wedge\; \mathsf{RanGen}(K') \;\wedge\; \hat{\phi},\vec{x} \rhd^{\mathcal{O}} y\\ &\;\wedge\; \hat{\phi},\vec{x},dec(y,dK) \blacktriangleright^{\mathcal{O}} K'\\ &\;\wedge\; \forall x R(y = \{x\}_{eK}^R \to \{x\}_{eK}^R \not\sqsubseteq \hat{\phi},\vec{x})\\ &\longrightarrow\quad \hat{\phi},\vec{x} \blacktriangleright^{\mathcal{O}} K'\end{aligned}$$

This axiom is very similar to the one about the derivability predicate, and the reason is again that the adversary trying to break key usability is allowed to use encryption and decryption oracles.

- Uncorrupted keys securely encrypt:
  - If $\mathcal{O}$ is either aic2 or sic2, then

$$\begin{aligned}&\mathsf{RanGen}(K)\ \wedge\ \mathsf{fresh}(R;\hat{\phi},\vec{x},x,y,K)\\&\quad\wedge\ \vec{x},x,y\preccurlyeq\hat{\phi}\ \wedge\ \hat{\phi},\vec{x},\{x\}^R_{eK}\rhd^{\mathcal{O}}y\\&\quad\longrightarrow\ \hat{\phi},\vec{x},x\blacktriangleright^{\mathcal{O}}K\ \vee\ \hat{\phi},\vec{x}\rhd^{\mathcal{O}}y\end{aligned}$$

This formula means that if the key is uncorrupted, that is, $\hat{\phi},\vec{x},x\not\blacktriangleright^{\mathcal{O}}K$, then $\{x\}^R_{eK}$ cannot help in deriving y. In other words, if it is possible to derive y with $\{x\}^R_{eK}$, then it is also possible to derive it without $\{x\}^R_{eK}$. The freshness and random generation conditions ensure that $\{x\}^R_{eK}$ is indeed a good encryption (e.g. $\{N\}^N_{eK}$ or $\{N\}^{eK}_{eK}$ are not good), and also that y cannot depend on $\{x\}^R_{eK}$ (e.g. $y=\{x\}^R_{eK}$ is not good). Moreover, $\vec{x},x,y\preccurlyeq\hat{\phi}$ ensures that handles in these terms are given values the adversary can compute (as some trivial examples taken $x=h$, the handle h cannot be dK if dK was never sent, or it cannot be R either).

This formula is completely analogous to the secrecy axiom in [9] but $dK\sqsubseteq\hat{\phi},\vec{x},x$ there is replaced now with $\hat{\phi},\vec{x},x\ \blacktriangleright^{\mathrm{aic2}}\ K$ as we can now allow dK to appear inside a secure encryption for example.

  - If $\mathcal{O}$ is either akc2 or skc2, then

$$\begin{aligned}&\mathsf{RanGen}(K)\ \wedge\ \mathsf{fresh}(R;\hat{\phi},\vec{x},x,y,K)\\&\quad\wedge\ \vec{x},x,y\preccurlyeq\hat{\phi}\ \wedge\ \hat{\phi},\vec{x},\{x\}^R_{eK}\rhd^{\mathcal{O}}y\\&\quad\longrightarrow\ \hat{\phi},\vec{x}\blacktriangleright^{\mathcal{O}}K\ \vee\ \hat{\phi},\vec{x}\rhd^{\mathcal{O}}y\end{aligned}$$

Notice, that the difference here from the axiom for IND-CCA2 security is that in $\hat{\phi},\vec{x}\ \blacktriangleright^{\mathcal{O}}\ K$ now there is no x. This corresponds to the fact that the encrypted message may contain the decryption key, or it may leak it somehow together with $\hat{\phi},\vec{x}$. For more discussion, see again Section VII.

It may be surprising however, that these axioms *do not require any security of the encryption*. It is purely a consequence of the definition of key corruption and derivability predicates. (The axiom that requires CCA2 security is the fresh keys are uncorrupted axiom later.) Here it becomes clear why we needed the additional encryptions $\{\vec{y}\}_{\vec{K}}$ in the computational semantics of KDM-CCA2 key usability. Because the x may contain elements besides the key that are inaccessible to the adversary on the left hand side of $\hat{\phi},\vec{x}\blacktriangleright^{\mathcal{O}}K$.

- Encryptions with uncorrupted keys do not corrupt:
  - IND-CCA2 case. If $\mathcal{O}$ is either aic2 or sic2, then

$$\begin{aligned}&\mathsf{RanGen}(K)\wedge\mathsf{RanGen}(K')\wedge\mathsf{fresh}(R;\hat{\phi},\vec{x},x,K,K')\\&\quad\wedge\ \vec{x},x\preccurlyeq\hat{\phi}\ \wedge\ \hat{\phi},\vec{x},\{x\}^R_{eK'}\blacktriangleright^{\mathcal{O}}K\\&\quad\longrightarrow\ \hat{\phi},\vec{x},x\blacktriangleright^{\mathcal{O}}K'\ \vee\ \hat{\phi},\vec{x}\blacktriangleright^{\mathcal{O}}K\end{aligned}$$

That is, if $\hat{\phi},\vec{x},\{x\}^R_{eK'}$ corrupted K, then either K is already corrupted without $\{x\}^R_{eK'}$, or K' was already corrupted by $\hat{\phi},\vec{x},x$. Note that this includes x, the encrypted term. This means that x itself (with $\hat{\phi},\vec{x}$) should not corrupt K' if we want $\{x\}^R_{eK'}$ to be safe. This is the generalization of that key cycles may corrupt CCA2 encryption. In Section VII we will see how this axiom deals with key cycles.

  - KDM-CCA2 case. If $\mathcal{O}$ is akc2 or skc2, then

$$\begin{aligned}&\mathsf{RanGen}(K)\wedge\mathsf{RanGen}(K')\wedge\mathsf{fresh}(R;\hat{\phi},\vec{x},x,K,K')\\&\quad\wedge\ \vec{x},x\preccurlyeq\hat{\phi}\ \wedge\ \hat{\phi},\vec{x},\{x\}^R_{eK'}\blacktriangleright^{\mathcal{O}}K\\&\quad\longrightarrow\ \hat{\phi},\vec{x}\blacktriangleright^{\mathcal{O}}K'\ \vee\ \hat{\phi},\vec{x}\blacktriangleright^{\mathcal{O}}K\end{aligned}$$

This is basically the same as the previous one, except again that $\hat{\phi},\vec{x}\blacktriangleright K'$ does not contain x.

Again, as it is clear from the soundness proofs in Section A, the reason for soundness of these axioms follows directly from the definition of key usability, and it does not depend on what encryption is used.

In the above formulas, K and K' could be allowed to encrypt different kinds of encryptions, not necessarily the same, we just did not want to overload our formulas.

The next axioms express that freshly generated items have not had the opportunity to corrupt or to become corrupted.

- **Axioms for Freshly Generated Items.**
  - Fresh keys are not corrupted: The intuition of this axiom is that if K is fresh, then it can be used for secure encryption: $\mathsf{keyfresh}(K;\hat{\phi})\ \longrightarrow\ \hat{\phi}\ \not\blacktriangleright^{\mathcal{O}}K$ This axiom is sound if the encryption is CCA2 secure. Depending on which $\mathcal{O}$ is in the axiom, the encryption needs to have the corresponding level of security. *This is the only axiom where the security of the encryption is necessary.* The reader may wonder that proving the KDM case, what happens to the $\{\vec{y}\}_{\vec{K}}$'s as the encryption oracles only fill in the gaps of the keys, not other unknown items. However, in a KDM attack created by the failure of the axiom, the attacker simulates the protocol, and all honestly generated items except for the key in question are available to him.
  - Fresh items do not corrupt: they were generated independently and as they have not been sent out, they have not had a chance to corrupt other items: $\mathsf{fresh}(x;\hat{\phi},\vec{x},y)\ \wedge\ \vec{x},y\preccurlyeq\hat{\phi}\ \wedge\ \hat{\phi},\vec{x},x\blacktriangleright^{\mathcal{O}}y\ \longrightarrow\ \hat{\phi},\vec{x}\blacktriangleright^{\mathcal{O}}y$

C. Key Usability for Unforgeability

Definition V.3. We define INT-CTXT corruption as: We say that $\mathcal{M},\Pi,S,\sigma \models^c \hat{\phi},\vec{x} \blacktriangleright^{\text{ic}} K$, if $\mathcal{M},\Pi,S,\sigma \models^c \vec{x} \not\preccurlyeq \hat{\phi} \vee \neg\mathsf{RanGen}(K)$, or for all $S' \subseteq S''$, there are $S'' \subseteq S'$, a PT algorithm $\mathcal{A}^{\mathcal{O}^{\text{sic2}}}$, such that

$$\mathcal{M},\Pi,S'',\sigma \models^c sdec(\mathcal{A}^{\mathcal{O}^{\text{sic2}}}(\hat{\phi},\vec{x}),K) \neq \bot \\ \wedge\ \forall z R(\mathcal{A}^{\mathcal{O}^{\text{sic2}}}(\hat{\phi},\vec{x}) = \{\!|z|\!\}_K^R \to \{\!|z|\!\}_K^R \not\sqsubseteq \hat{\phi},\vec{x})$$

And on S'', $\mathcal{A}^{\mathcal{O}^{\text{sic2}}}(\hat{\phi},\vec{x})$ is not equal any of the outputs of the encryption oracles.

The reason for using oracle $\mathcal{O}^{\text{sic2}}$ is that the definition of INT-CTXT security [13] allows the use of encryption and decryption oracles.

D. Axiom for Unforgeability Key Usability

- Equal terms are substitutable on the rhs of $\blacktriangleright^{\text{ic}}$.
- Derivability implies corruption:
  $\hat{\phi},\vec{x} \rhd K \ \longrightarrow\ \hat{\phi},\vec{x} \blacktriangleright^{\text{ic}} K$
- Increasing capabilities for key corruption:
  $\hat{\phi},\vec{x} \blacktriangleright^{\text{ic}} K \ \longrightarrow\ \hat{\phi},\vec{x},x \blacktriangleright^{\text{ic}} K$
- Commutativity: If $\vec{x}'$ is a permutation of $\vec{x}$, then $\hat{\phi},\vec{x} \blacktriangleright^{\text{ic}} K \ \longrightarrow\ \hat{\phi},\vec{x}' \blacktriangleright^{\text{ic}} K$.
- Transitivity: $\hat{\phi},\vec{x} \rhd \vec{y} \wedge \hat{\phi},\vec{x},\vec{y} \blacktriangleright^{\text{ic}} K \ \longrightarrow\ \hat{\phi},\vec{x} \blacktriangleright^{\text{ic}} K$
- Decryptions of adversarial ciphers do not help: If $\mathcal{O}$ is IND or KDM CCA2, symmetric or asymmetric, then

$$\begin{aligned}&\mathsf{RanGen}(K)\ \wedge\ \mathsf{RanGen}(K')\ \wedge\ \hat{\phi},\vec{x} \rhd^{\mathcal{O}} y\\ &\wedge\ \hat{\phi},\vec{x}, sdec(y,dK) \blacktriangleright^{\text{ic}} K'\\ &\wedge\ \forall x R(y = \{\!|x|\!\}_{eK}^R \to \{\!|x|\!\}_{eK}^R \not\sqsubseteq \hat{\phi},\vec{x})\\ &\longrightarrow\quad \hat{\phi},\vec{x} \blacktriangleright^{\text{ic}} K'\end{aligned}$$

- Uncorrupted key's encryption cannot be faked:

$$\begin{aligned}&\mathsf{RanGen}(K)\ \wedge\ \hat{\phi},\vec{x} \rhd y\ \wedge\ dec(y,dK) \neq \bot\\ &\wedge\ \forall x R(y = \{x\}_{eK}^R \to \{x\}_{eK}^R \not\sqsubseteq \hat{\phi},\vec{x})\\ &\longrightarrow\quad \hat{\phi},\vec{x} \blacktriangleright^{\text{ic}} K\end{aligned}$$

  This is means the adversary cannot compute a y which decrypts to something meaningful. This is exactly what we need from the INT-CTXT property, namely, that the encryption cannot be faked. Again, soundness of this axiom does not need INT-CTXT encryption.

- Encryptions with uncorrupted keys do not corrupt:
  - For the IND case, we have

$$\begin{aligned}&\mathsf{RanGen}(K) \wedge \mathsf{RanGen}(K') \wedge \mathsf{fresh}(R;\hat{\phi},\vec{x},x,K,K')\\ &\wedge\ \vec{x},x \preccurlyeq \hat{\phi}\ \wedge\ \hat{\phi},\vec{x},\{\!|x|\!\}_{K'}^R \blacktriangleright^{\text{ic}} K\\ &\longrightarrow\ \hat{\phi},\vec{x},x \blacktriangleright^{\text{sic2}} K'\ \vee\ \hat{\phi},\vec{x} \blacktriangleright^{\text{ic}} K\end{aligned}$$

  - For the KDM case, we have

$$\begin{aligned}&\mathsf{RanGen}(K) \wedge \mathsf{RanGen}(K') \wedge \mathsf{fresh}(R;\hat{\phi},\vec{x},x,K,K')\\ &\wedge\ \vec{x},x \preccurlyeq \hat{\phi}\ \wedge\ \hat{\phi},\vec{x},\{\!|x|\!\}_{K'}^R \blacktriangleright^{\text{ic}} K\\ &\longrightarrow\ \hat{\phi},\vec{x} \blacktriangleright^{\text{skc2}} K'\ \vee\ \hat{\phi},\vec{x} \blacktriangleright^{\text{ic}} K\end{aligned}$$

  Again, soundness of these axioms follow directly from the key corruption definitions.

- Fresh keys are not INT-CTXT corrupted if encryption is INT-CTXT secure:
  - $\mathsf{keyfresh}(K;\hat{\phi}) \ \longrightarrow\ \hat{\phi} \not\blacktriangleright^{\text{ic}} K$. The intuition of this axiom is that if the encryption is INT-CTXT secure and if K is fresh, then the adversary cannot fake encryptions with this key.
- Fresh items do not corrupt: $\mathsf{fresh}(x;\hat{\phi},\vec{x},K)\ \wedge\ \vec{x} \preccurlyeq \hat{\phi}\ \wedge\ \hat{\phi},\vec{x},x \blacktriangleright^{\text{ic}} K \ \longrightarrow\ \hat{\phi},\vec{x} \blacktriangleright^{\text{ic}} K$

VI. Soundness

We have the following soundness theorem for the axioms we introduced. Proof is included in the Appendix.

Theorem VI.1. With the computational interpretations of derivability and key corruption predicates, the axioms are computationally sound. For the "fresh keys are not corrupted", it is necessary that the implementation of the encryption satisfies the corresponding (symmetric or asymmetric, IND or KDM-CCA2 security, or INT-CTXT unforgeability). Soundness of the other axioms do not require that. Furthermore, no-telepathy axiom requires that freshly generated items are guessable only with negligible probability.

VII. Simple Examples

Now let us see on a few simple examples how inconsistency can be shown with the above axioms. In [9], the authors presented some of the most basic examples, so the ones that we create here are a little more complex, all are related to sending keys around. We use symmetric encryption in these examples.

Example VII.1. Suppose the first messages in a frame are

$$\phi_3 \equiv \nu_{KK_{AB}NR_1R_2}((A,B),\{\!|K|\!\}_{K_{AB}}^{R_1},\{\!|h_2,N|\!\}_K^{R_2}),$$

where the symmetric encryption is CCA2 (or KDM) secure. We want to show that $\phi_3 \rhd N$ is inconsistent with the axioms, that is, N remains secret. Let now $\mathcal{O}$ denote either symmetric IND-CCA2 or symmetric KDM-CCA2. Suppose $\phi_3 \rhd N$ holds. Then $\phi_3 \rhd^{\mathcal{O}} N$ by the more oracles help more axiom. That is the same as $\phi_2,\{\!|h_2,N|\!\}_K^{R_2} \rhd^{\mathcal{O}} N$. By the no-telepathy axiom, $\phi_2 \not\rhd^{\mathcal{O}} N$ as $\mathsf{fresh}(N,\phi_2)$ holds (which follows directly from the definition of the freshness constraint, not from axioms). By the 'uncorrupted key securely encrypts' axiom for CCA2 symmetric case, with the roles $\vec{x} \equiv \langle\rangle$, $x \equiv \langle h_2,N\rangle$, $y \equiv N$, since we assumed $\phi_2,\{\!|h_2,N|\!\}_K^{R_2} \rhd^{\mathcal{O}} N$, we also have that either $\phi_2 \rhd^{\mathcal{O}} N$ (already ruled out) or (depending on $\mathcal{O}$) $\phi_2,h_2,N \blacktriangleright^{\text{sic2}} K$

or $\phi_2 \blacktriangleright^{\text{skc2}} K$. In the IND-CCA2 case, by the 'fresh items do not corrupt' axiom, we then have $\phi_2, h_2 \blacktriangleright^{\text{sic2}} K$ as N does not appear in ϕ_2. Since the handle is always derived from the frame, $\phi_2 \rhd^{\text{sic2}} h_2$ holds, hence $\phi_2 \rhd^{\text{sic2}} h_2$ and by the transitivity axiom applied for $\phi_2, h_2 \blacktriangleright^{\text{sic2}} K$ and $\phi_2 \rhd^{\text{sic2}} h_2$, we have $\phi_2 \blacktriangleright^{\text{sic2}} K$, just as we had in the KDM case earlier. But that is the same as (now for both IND and KDM cases) $\phi_1, \{\!|K|\!\}^{R_1}_{K_{AB}} \blacktriangleright^{\mathcal{O}} K$. By the 'encryptions with uncorrupted keys do not corrupt' axiom, with roles $K' \equiv K_{AB}$, $\vec{x} \equiv \langle\rangle$ and $x \equiv K$, we have that either $\phi_1 \blacktriangleright^{\mathcal{O}} K$, or $\phi_1, K \blacktriangleright^{\text{sic2}} K_{AB}$ or $\phi_1 \blacktriangleright^{\text{sic2}} K_{AB}$. However, $\phi_1 \not\blacktriangleright^{\mathcal{O}} K$ because of the 'fresh keys are not corrupted' axiom, and the same is true for $\phi_1 \not\blacktriangleright^{\mathcal{O}} K_{AB}$. So for the KDM case we are done, and finally for the IND case, again by the 'fresh items do not corrupt' axiom, $\phi_1, K \not\blacktriangleright^{\text{sic2}} K_{AB}$, and we arrived at a contradiction in both IND and KDM cases.

Example VII.2. Now suppose

$$\phi_3 \equiv \nu_{K K_{AB} N}((A,B), \{\!|K|\!\}^{R_1}_{K_{AB}}, \{\!|K_{AB}, h_2, N|\!\}^{R_2}_{K})$$

and that $\phi_3 \rhd^{\mathcal{O}} N$ holds. There is a key cycle in this example. For IND-CCA2 security, from the 'encryptions with uncorrupted keys do not corrupt' axiom we arrive at $\phi_2, K_{AB}, h_2, N \blacktriangleright^{\text{sic2}} K$ if we follow the same steps as we did in Example VII.1. Then the same way as before, we can remove h_2 and N, and since $\phi_2 \equiv \phi_1, \{\!|K|\!\}^{R_1}_{K_{AB}}$, receive $\phi_1, \{\!|K|\!\}^{R_1}_{K_{AB}}, K_{AB} \blacktriangleright^{\text{sic2}} K$. But this does not lead to a contradiction! According to the equational theory, $K = \mathit{sdec}(\{\!|K|\!\}^{R_1}_{K_{AB}}, K_{AB})$, and by the 'functions are computable' axiom, we get $\phi_1, \{\!|K|\!\}^{R_1}_{K_{AB}}, K_{AB} \rhd^{\text{sic2}} K$. So we always have $\phi_1, \{\!|K|\!\}^{R_1}_{K_{AB}}, K_{AB} \blacktriangleright^{\text{sic2}} K$ too by the 'derivability implies corruption' axiom, there is no contradiction. However, if we have KDM security, then just as in the previous example, using the 'encryption with uncorrupted keys do not corrupt' axiom, we immediately arrive at $\phi_2 \blacktriangleright^{\text{skc2}} K$, and the rest of the derivation is the same as in the previous example.

Example VII.3. Now consider

$$\phi_3 = \nu_{K K_{AB} N}((A,B), \{\!|K|\!\}^{R_1}_{K_{AB}}, \{\!|\{\!|K_{AB}|\!\}^{R_2}_{K'}, h_2, N|\!\}^{R_3}_{K}).$$

Strictly speaking, K and K_{AB} are still in cycles, but they do not disturb each other because of K'. Again, assuming IND-CCA2 security, from $\phi_3 \rhd^{\text{sic2}} N$ first $\phi_2, \{\!|K_{AB}|\!\}^{R_2}_{K'}, h_2, N \blacktriangleright^{\text{sic2}} K$ is derived using the 'encryption with uncorrupted key does not corrupt' axiom as in Example VII.1. Then as in Example VII.1, h_2 and N are removed: $\phi_2, \{\!|K_{AB}|\!\}^{R_2}_{K'} \blacktriangleright^{\text{sic2}} K$. At this point, the 'encryptions with uncorrupted keys do not corrupt' axiom implies that either $\phi_2 \blacktriangleright^{\text{sic2}} K$ or $\phi_2, K_{AB} \blacktriangleright^{\text{sic2}} K'$. In the former case, we are back in the situation of Example VII.1 and we arrive at a contradiction. In the latter case, by function (encryption) application on $\phi_1, \{\!|K|\!\}^{R_1}_{K_{AB}}, K_{AB} \blacktriangleright^{\text{sic2}} K'$, we receive that $\phi_1, K, K_{AB}, R_1 \blacktriangleright^{\text{sic2}} K'$, but by 'the fresh items do not corrupt' axiom all of them can be removed, and receive that $\phi_1 \blacktriangleright^{\text{sic2}} K'$ contradicting the 'fresh keys are uncorrupted' axiom.

VIII. The Symmetric Needham-Schroeder Protocol

With the axioms that we presented, we have proven the amended symmetric Needham-Schroeder protocol:

1. $A \to B : A$
2. $B \to A : \{A, N_1\}_{K_{BT}}$
3. $A \to T : \langle A, B, N_2, \{A, N_1\}_{K_{BT}} \rangle$
4. $T \to A : \{N_2, B, K, \{K, N_1, A\}_{K_{BT}}\}_{K_{AT}}$
5. $A \to B : \{K, N_1, A\}_{K_{BT}}$
6. $B \to A : \{N_3\}_K$
7. $A \to B : \{N_3 - 1\}_K$

This protocol first has a key distribution part, and then the distributed key is used to securely encrypt a nonce. We showed that no symbolic (hence computational) attacker succeeds the following way (motivated by [23]). Using IND-CCA2 and INT-CTXT axioms, we first showed by an inductive technique that the key K from the trusted party meant for honest A and B are never corrupted (corruption is inconsistent with the axioms and agent checks). Then, again with an inductive technique we showed that N_3 is not leaked. Finally, agreement and authentication are shown. Besides the presented axioms, we also needed that adding 1 and subtracting 1 are inverses of each other, and $x - 1 \neq x$. We needed an additional property, namely, that applying the first projection of a pairing on an honestly generated nonce cannot be itself with more than negligible probability. Finally, triples, quadruples were constructed out of pairs. The detailed proof is available online [8]. We assumed that A is running the initiator role in all his sessions, and B is running the responder's role. There is only one trusted party. They all are allowed to run any number of multiple parallel sessions with honest and corrupted agents.

On a note about dynamic corruption, the proof works even if the protocol allows the release of the key K at a later time. Secrecy can still be proven until that point, authentication that was carried out earlier can still be verified.

IX. Conclusions

In this paper we further expanded the framework proposed by Bana and Comon-Lundh [6], where one does not define explicitly the Dolev-Yao adversarial capabilities but rather the limitations on these capabilities. We have shown how key distribution can be handled. The proofs with this technique are computationally sound without the need of any further assumptions such as no bad keys, etc that are assumed in other literature. We presented a modular set of axioms that are computationally sound for implementations using IND-CCA2, KDM-CCA2 and INT-CTXT secure encryptions respectively. We illustrated their power via simple examples and the verification of an entire protocol.

Plans for future work include extension to unbounded number of sessions, equivalence properties and automation.

X. Acknowledgements

This work was partially supported by FCT project ComFormCrypt PTDC/EIA-CCO/113033/2009.

References

[1] M. Backes, B. Pfitzmann, and M. Waidner. A composable cryptographic library with nested operations. In *CCS'03*, 2003.

[2] M. Backes, B. Pfitzmann, and M. Waidner. The reactive simulatability (rsim) framework for asynchronous systems. *Information and Computation*, 205(12), 2007.

[3] Michael Backes, Ankit Malik, and Dominique Unruh. Computational soundness without protocol restrictions. In *ACM CCS 2012*, pages 699–711. ACM Press, October 2012. Preprint on IACR ePrint 2012/486.

[4] Michael Backes, Birgit Pfitzmann, and Andre Scedrov. Key-dependent message security under active attacks - brsim/uc-soundness of dolev-yao-style encryption with key cycles. *Journal of Computer Security*, 16(5):497–530, 2008.

[5] G. Bana and H. Comon-Lundh. Towards unconditional soundness: Computationally complete symbolic attacker. Available at IACR ePrint Archive, Report 2012/019.

[6] G. Bana and H. Comon-Lundh. Towards unconditional soundness: Computationally complete symbolic attacker. In *Proceedings of POST'12*, LNCS, 2012.

[7] G. Bana, K. Hasebe, and M. Okada. Computational semantics for basic protocol logic - a stochastic approach. In *ASIAN'07*, volume 4846 of *LNCS*, pages 86–94. Springer, 2007.

[8] G. Bana, K. Hasebe, and M. Okada. Symbolic verification of the amended needham-schroeder shared-key protocol, 2013. http://prosecco.gforge.inria.fr/personal/gebana/asns.pdf.

[9] Gergei Bana, Pedro Adão, and Hideki Sakurada. Computationally Comlete Symbolic Attacker in Action. In *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2012)*, Leibniz International Proceedings in Informatics (LIPIcs), Dagstuhl, Germany, 2012. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[10] G. Barthe, B. Grégoire, and S. Zanella Béguelin. Formal certification of code-based cryptographic proofs. In *POPL'09*, pages 90–101. ACM, 2009.

[11] Gilles Barthe, Benjamin Grégoire, and Santiago Zanella Béguelin. Formal certification of code-based cryptographic proofs. In Zhong Shao and Benjamin C. Pierce, editors, *POPL*, pages 90–101. ACM, 2009.

[12] Mihir Bellare, Anand Desai, David Pointcheval, and Phillip Rogaway. Relations among notions of security for public-key encryption schemes. In Hugo Krawczyk, editor, *CRYPTO*, volume 1462 of *Lecture Notes in Computer Science*, pages 26–45. Springer, 1998.

[13] Mihir Bellare and Chanathip Namprempre. Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. *J. Cryptology*, 21(4):469–491, 2008.

[14] B. Blanchet. A computationally sound mechanized prover for security protocols. *IEEE Transactions on Dependable and Secure Computing*, 5(4):193–207, 2008.

[15] Jan Camenisch, Nishanth Chandran, and Victor Shoup. A public key encryption scheme secure against key dependent chosen plaintext and adaptive chosen ciphertext attacks. In Antoine Joux, editor, *EUROCRYPT*, volume 5479 of *Lecture Notes in Computer Science*, pages 351–368. Springer, 2009.

[16] H. Comon-Lundh and V. Cortier. Computational soundness of observational equivalence. In *CCS'08*, 2008.

[17] H. Comon-Lundh and V. Cortier. How to prove security of communication protocols? A discussion on the soundness of formal models w.r.t. computational ones. In *STACS'11*, volume 9 of *Leibniz International Proceedings in Informatics*, pages 29–44, March 2011.

[18] V. Cortier and B. Warinschi. Computationally sound, automated proofs for security protocols. In *ESOP'05*, volume 3444 of *LNCS*, pages 157–171, 2005.

[19] A. Datta, A. Derek, J. C. Mitchell, V. Shmatikov, and M. Turuani. Probabilistic polynomial-time semantics for a protocol security logic. In *ICALP'05*, volume 3580 of *LNCS*, pages 16–29. Springer, 2005.

[20] Anupam Datta, Ante Derek, John C. Mitchell, and Bogdan Warinschi. Computationally sound compositional logic for key exchange protocols. In *CSFW '06: Proceedings of the 19th IEEE workshop on Computer Security Foundations*, pages 321–334, Washington, DC, USA, 2006. IEEE Computer Society.

[21] Melvin Fitting. An embedding of classical logic in s4. *The Journal of Symbolic Logic*, 35(4):529–534, 1970.

[22] Ralf Küsters and Max Tuengerthal. Computational soundness for key exchange protocols with symmetric encryption. In Ehab Al-Shaer, Somesh Jha, and Angelos D. Keromytis, editors, *ACM Conference on Computer and Communications Security*, pages 91–100. ACM, 2009.

[23] F. Javier Thayer, Jonathan C. Herzog, and Joshua D. Guttman. Strand spaces: Proving security protocols correct. *Journal of Computer Security*, 7(1):191–230, 1999.

Appendix

We prove now Theorem VI.1.

Proof: As the soundness proofs of the axioms for derivability with oracles is the same as the soundness proofs below (and also as the proofs in [9]), we skip that and focus on key corruption.

First we prove Axioms in Section V-B about key corruption in the CCA2 cases.

- Substitutability of equal terms: The reason is that the corruption of the item on the right hand side of $\blacktriangleright^{\mathcal{O}}$

only depends on the bit string that is associated to the term there, and not on the structure of the term. This is in contrast with the left hand side. The notion that anything can be submitted to the decryption oracle that is not an honest encryption on the left clearly depends on the term structure on the left, so there substitutability does not hold.

- Derivability implies corruption: Soundness of this axiom is rather trivial, but we write it out for clarity. In order to show that in any protocol execution, $\mathcal{M},\Pi \models \forall\vec{x},K(\hat{\phi},\vec{x} \rhd^{\mathcal{O}} K \longrightarrow \hat{\phi},\vec{x} \blacktriangleright^{\mathcal{O}} K)$, by the computational semantics we have to show that for any evaluation σ of the variables, and for any S non-negligible set, $\mathcal{M},\Pi,S,\sigma \models \hat{\phi},\vec{x} \rhd^{\mathcal{O}} K$ implies $\mathcal{M},\Pi,S,\sigma \models \hat{\phi},\vec{x} \blacktriangleright^{\mathcal{O}} K$. So suppose $\mathcal{M},\Pi,S,\sigma \models \hat{\phi},\vec{x} \rhd^{\mathcal{O}} K$ holds. To show $\mathcal{M},\Pi,S,\sigma \models \hat{\phi},\vec{x} \blacktriangleright^{\mathcal{O}} K$, let's take any $S' \subseteq S$. By $\mathcal{M},\Pi,S,\sigma \models \hat{\phi},\vec{x} \rhd^{\mathcal{O}} K$, there is a $S'' \subseteq S'$ and an algorithm $\mathcal{B}$ such that $\mathcal{M},\Pi,S'',\sigma \| \!\!=^{c} \mathcal{B}(\hat{\phi},\vec{x}) = K$. In the IND-CCA2 case, we can chose $\mathcal{A}_1$ in the key corruption definition simply to be some (only negligibly guessable) random bit string $r(\omega)$, and y can also be the same. Then, $\mathcal{A}_2$ can simply take the key, do the decryption. If the encryption is that of $r(\omega)$ then $r(\omega)$ is received back. If not, then 0's are received everywhere on S'', which is overwhelmingly different from our chosen y. KDM case is exactly the same except that r in that case becomes the interpretation of x, and $\mathcal{A}_2$ becomes $\mathcal{A}$.
- Increasing capabilities for key corruption: Soundness of this axiom is again easy. Assuming $\hat{\phi},\vec{x} \blacktriangleright^{\mathcal{O}} K$, there is an x, y, etc as in the definition of key corruption. The same items will also be good for $\hat{\phi},\vec{x},x' \blacktriangleright^{\mathcal{O}} K$.
- Commutativity: Completely trivial, the definition of key corruption is invariant under the change of the order of the list $\vec{x}$.
- Transitivity: Assuming $\mathcal{M},\Pi,S,\sigma \models \hat{\phi},\vec{x} \rhd^{\mathcal{O}} \vec{y}$ and $\mathcal{M},\Pi,S,\sigma \models \hat{\phi},\vec{x},\vec{y} \blacktriangleright^{\mathcal{O}} K$, to show $\mathcal{M},\Pi,S,\sigma \models \hat{\phi},\vec{x} \blacktriangleright^{\mathcal{O}} K$, take an arbitrary $S' \subseteq S$. By $\mathcal{M},\Pi,S,\sigma \models \hat{\phi},\vec{x} \rhd^{\mathcal{O}} \vec{y}$, there is a $S'' \subseteq S'$ and an $\mathcal{A}^{\mathcal{O}}$ such that $\mathcal{M},\Pi,S'',\sigma \| \!\!= \mathcal{A}^{\mathcal{O}}(\hat{\phi},\vec{x}) = \vec{y}$. By $\mathcal{M},\Pi,S,\sigma \models \hat{\phi},\vec{x},\vec{y} \blacktriangleright^{\mathcal{O}} K$, there is a $S''' \subseteq S''$ such that for this S''', the conditions in the definition of key corruption (in place of S'') hold. Also, $\mathcal{M},\Pi,S''',\sigma \| \!\!= \mathcal{A}^{\mathcal{O}}(\hat{\phi},\vec{x}) = \vec{y}$. So in the key corruption definition applied to $\mathcal{M},\Pi,S,\sigma \models \hat{\phi},\vec{x},\vec{y} \blacktriangleright^{\mathcal{O}} K$, $\mathcal{A}_1^{\mathcal{O}}$ and $\mathcal{A}_2^{\mathcal{O}}$ can run $\mathcal{A}^{\mathcal{O}}$ as a subroutine to compute $\vec{y}$, so they do not need it as an input. Since there is such a $S''' \subseteq S'$ for all $S' \subseteq S$, we get $\mathcal{M},\Pi,S,\sigma \models \hat{\phi},\vec{x} \blacktriangleright^{\mathcal{O}} K$.
- Decryptions of adversarial ciphers do not help: We explained soundness of this axiom at its introduction (more precisely at the similar axiom for $\rhd^{\mathcal{O}}$, which is completely analogous).
- Secrecy of CCA2 encryption: For the IND case, we have

$$\begin{aligned}&\mathsf{RanGen}(K) \ \wedge\ \mathsf{fresh}(R;\hat{\phi},\vec{x},x,y,K)\\ &\quad\wedge\ \vec{x},x,y \preccurlyeq \hat{\phi}\ \wedge\ \hat{\phi},\vec{x},\{x\}_{eK}^{R} \rhd^{\mathcal{O}} y\\ &\quad\longrightarrow\ \hat{\phi},\vec{x},x \blacktriangleright^{\mathcal{O}} K\ \vee\ \hat{\phi},\vec{x} \rhd^{\mathcal{O}} y\end{aligned}$$

Soundness of this follows easily from our definition of key corruption and derivability with oracle access. Note, that CCA2 security of the encryption is not needed. Let's move $\hat{\phi},\vec{x},x \blacktriangleright^{\mathcal{O}} K$ to the premise, it becomes $\hat{\phi},\vec{x},x \not\blacktriangleright^{\mathcal{O}} K$. Let us denote by θ the premise received this way. We have to show that for any σ evaluation of free variables, and S non-negligible set, if $\mathcal{M},\Pi,S,\sigma \models \theta$, then $\mathcal{M},\Pi,S,\sigma \models \hat{\phi},\vec{x} \rhd^{\mathcal{O}} y$. So let us suppose $\mathcal{M},\Pi,S,\sigma \models \theta$. To show $\mathcal{M},\Pi,S,\sigma \models \hat{\phi},\vec{x} \rhd^{\mathcal{O}} y$, take any non-negligible set $S' \subseteq S$. As $\mathcal{M},\Pi,S,\sigma \models \theta$ implies $\mathcal{M},\Pi,S,\sigma \models \hat{\phi},\vec{x},\{x\}_{eK}^{R} \rhd^{\mathcal{O}} y$ by the semantics of compound formulas, for such an S', by the definition of derivability, there is a subset $S'' \subseteq S'$ and an algorithm $\mathcal{A}^{\mathcal{O}}$ such that $\mathcal{M},\Pi,S'',\sigma \| \!\!= \mathcal{A}^{\mathcal{O}}(\hat{\phi},\vec{x},\{x\}_{eK}^{R}) = y$. In the definition of key corruption, taking $\mathcal{A}_1^{\mathcal{O}}$ to be the algorithm that picks x from $\hat{\phi},\vec{x},x$, and taking $\mathcal{A}_2^{\mathcal{O}}$ to be the same as $\mathcal{A}^{\mathcal{O}}$, by $\mathcal{M},\Pi,S'',\sigma \models \hat{\phi},\vec{x},x \not\blacktriangleright^{\mathcal{O}} K$, we also have that $\mathcal{M},\Pi,S'',\sigma \| \!\!= \mathcal{A}^{\mathcal{O}}(\hat{\phi},\vec{x},\{0^{|x|}\}_{eK}^{R}) = y$. Since the length of x can be polynomially guessed, there is an algorithm $\mathcal{B}^{\mathcal{O}}$ and a non-negligible subset $S''' \subseteq S''$ with $\mathcal{M},\Pi,S''',\sigma \| \!\!= \mathcal{B}^{\mathcal{O}}(\hat{\phi},\vec{x}) = y$. Since we have such an $S''' \subseteq S'$ for any arbitrary $S' \subseteq S$, by the definition of derivability this means $\mathcal{M},\Pi,S,\sigma \models \hat{\phi},\vec{x} \rhd^{\mathcal{O}} y$.
The argument for the KDM case is completely analogous, except that we have $\mathcal{M},\Pi,S'',\sigma \models \hat{\phi},\vec{x} \not\blacktriangleright^{\mathcal{O}} K$ there. Note, that as in the axiom for the KDM case, we only require about x that $x \preccurlyeq \hat{\phi}$, such the encryption of such an item is should not be indistinguishable from the encryption of 0's. That is why KDM oracles are not sufficient in the definition of key usability for the KDM case, we need the encryptions of $\vec{y}$ there.
- Finally, we have to show that encryptions with uncorrupted keys do not corrupt. Note again, that we do not need CCA2 security of the encryption, we only need the definition of key corruption.

$$\begin{aligned}&\mathsf{RanGen}(K) \wedge \mathsf{RanGen}(K') \wedge \mathsf{fresh}(R;\hat{\phi},\vec{x},x,K,K')\\ &\quad\wedge\ \vec{x},x \preccurlyeq \hat{\phi}\ \wedge\ \hat{\phi},\vec{x},\{x\}_{eK'}^{R} \blacktriangleright^{\mathcal{O}} K\\ &\quad\longrightarrow\ \hat{\phi},\vec{x},x \blacktriangleright^{\mathcal{O}} K'\ \vee\ \hat{\phi},\vec{x} \blacktriangleright^{\mathcal{O}} K\end{aligned}$$

Instead, just as in the previous entry, we show the following:

$$\begin{aligned}&\mathsf{RanGen}(K) \wedge \mathsf{RanGen}(K') \wedge \mathsf{fresh}(R;\hat{\phi},\vec{x},x,K,K')\\ &\quad\wedge \vec{x},x \preccurlyeq \hat{\phi}\ \wedge \hat{\phi},\vec{x},x,\{x\}_{eK'}^{R} \blacktriangleright^{\mathcal{O}} K' \wedge \hat{\phi},\vec{x} \not\blacktriangleright^{\mathcal{O}} K\\ &\quad\longrightarrow\ \hat{\phi},\vec{x} \blacktriangleright^{\mathcal{O}} K.\end{aligned}$$

Just as in the previous entry, we have to show that for all S non-negligible sets and σ evaluations of variables, if $\mathcal{M}, \Pi, S, \sigma$ satisfies the premise, then it satisfies the conclusion as well. So let us suppose it satisfies the premise. We want to show $\mathcal{M}, \Pi, S, \sigma \models \hat{\phi}, \vec{x} \blacktriangleright^{\mathcal{O}} K$. Following the definition of key corruption, take any subset $S' \subseteq S$. Take the IND-CCA2 case. By the definition of key corruption applied for $\mathcal{M}, \Pi, S, \sigma \models \hat{\phi}, \vec{x}, x, \{x\}^R_{eK'} \blacktriangleright^{\mathcal{O}} K'$, there are $S'' \subseteq S'$, y, R' $\mathcal{A}_1^{\mathcal{O}}$, $\mathcal{A}_2^{\mathcal{O}}$ with the appropriate conditions, in particular,

$$\mathcal{M},\Pi,S'',\sigma \| \models^c \mathcal{A}_2^{\mathcal{O}}(\hat{\phi}, \vec{x}, \{x\}^R_{eK'}, \{\mathcal{A}_1^{\mathcal{O}}(\hat{\phi}, \vec{x}, \{x\}^R_{eK'})\}^{R'}_{eK})) = y$$

but

$$\mathcal{M},\Pi,S'',\sigma \| \models^c \mathcal{A}_2^{\mathcal{O}}(\hat{\phi}, \vec{x}, \{x\}^R_{eK'}, \{0^{|\mathcal{A}_1^{\mathcal{O}}(\hat{\phi}, \vec{x}, \{x\}^R_{eK'})|}\}^{R'}_{eK})) \neq y.$$

Now, we also have $\mathcal{M}, \Pi, S'', \sigma \models \hat{\phi}, \vec{x}, x \not\blacktriangleright^{\mathcal{O}} K'$ from the satisfaction of the premise. Again, applying the computational semantics of key corruption to this situation, it gives us with the above that

$$\mathcal{M}, \Pi, S'', \sigma \| \models^c \mathcal{A}_2^{\mathcal{O}}(\hat{\phi}, \vec{x}, \{0^{|x|}\}^R_{eK'}, \{\mathcal{A}_1^{\mathcal{O}}(\hat{\phi}, \vec{x}, \{0^{|x|}\}^R_{eK'})\}^{R'}_{eK})) = y$$

and

$$\mathcal{M}, \Pi, S'', \sigma \| \models^c \mathcal{A}_2^{\mathcal{O}}(\hat{\phi}, \vec{x}, \{0^{|x|}\}^R_{eK'}, \{0^{|\mathcal{A}_1^{\mathcal{O}}(\hat{\phi}, \vec{x}, \{0^{|x|}\}^R_{eK'})|}\}^{R'}_{eK})) \neq y$$

because we can apply the key usability condition to $\mathcal{A}_x$ as $\mathcal{A}_1^{\mathcal{O}}$ in the definition of key corruption, and the combination of the $\mathcal{A}_1^{\mathcal{O}}$ and $\mathcal{A}_2^{\mathcal{O}}$ above as $\mathcal{A}_2^{\mathcal{O}}$ in the definition of key corruption. Note also, that R satisfies the appropriate freshness conditions. Note further that as the encryption oracle can be called for, the encryption $\{\mathcal{A}_1^{\mathcal{O}}(\hat{\phi}, \vec{x}, ...)\}^{R'}_{eK}$ can be computed by $\mathcal{A}_2^{\mathcal{O}}$ even in the case of symmetric encryption as it can compute $\mathcal{A}_1^{\mathcal{O}}(\hat{\phi}, \vec{x}, ...)$ first, and then submit to the encryption oracle. Since the length of x can be guessed, denoting by $\mathcal{B}_i^{\mathcal{O}}$ the algorithm that guesses it and then uses $\mathcal{A}_i^{\mathcal{O}}$, we have that there is an S''' non-negligible set (where the guessing of the length was correct) such that

$$\mathcal{M}, \Pi, S''', \sigma \| \models^c \mathcal{B}_2^{\mathcal{O}}(\hat{\phi}, \vec{x}, \{\mathcal{B}_1^{\mathcal{O}}(\hat{\phi}, \vec{x})\}^{R'}_{eK})) = y$$

and

$$\mathcal{M}, \Pi, S''', \sigma \| \models^c \mathcal{B}_2^{\mathcal{O}}(\hat{\phi}, \vec{x}, \{0^{|\mathcal{B}_1^{\mathcal{O}}(\hat{\phi}, \vec{x})|}\}^{R'}_{eK})) \neq y.$$

Since there is such an $S''' \subseteq S'$ for each $S' \subseteq S$, this exactly means that

$$\mathcal{M}, \Pi, S, \sigma \models \hat{\phi}, \vec{x} \blacktriangleright^{\mathcal{O}} K.$$

Again, the KDM case is proven entirely analogously.

- Fresh keys are not corrupted: Here is where IND-CCA2 and KDM-CCA2 security of the encryption is used. If the key has not been sent out, it can be used for safe encryption. The proof is very similar to the one presented in [5], and goes as follows. What is proven is that if $\mathcal{M}, \Pi, S, \sigma$ satisfies freshness of key K, then $\mathcal{M}, \Pi, S, \sigma \models \hat{\phi} \blacktriangleright^{\mathcal{O}} K$ leads to a CCA2 attack to the encryption. That is, for every $S' \subseteq S$, there are $S'' \subseteq S'$, etc. such that, most importantly,

$$\mathcal{M}, \Pi, S'', \sigma \| \models^c \mathcal{A}_2^{\mathcal{O}}(\hat{\phi}, \vec{x}, \{\mathcal{A}_1^{\mathcal{O}}(\hat{\phi}, \vec{x})\}^R_{eK})) = y \quad (1)$$

but

$$\mathcal{M}, \Pi, S'', \sigma \| \models^c \mathcal{A}_2^{\mathcal{O}}(\hat{\phi}, \vec{x}, \{0^{|\mathcal{A}_1^{\mathcal{O}}(\hat{\phi}, \vec{x})|}\}^R_{eK})) \neq y \quad (2)$$

(or vice versa). Clearly, what the CCA2 attacker has to do is to simulate the protocol execution such that

  – except for K, the CCA2 attacker generates all other keys
  – encryptions (except for that of $\mathcal{A}_1^{\mathcal{O}}(\hat{\phi}, \vec{x})$) with K are done by submitting two identical requests to the encryption oracle
  – the attacker keeps a table records of what encryption belongs to what plaintext
  – decryptions of ciphertexts provided by the encryption oracle are done by looking it up in the table
  – decryptions of strings not provided by the oracle are done by submitting to the decryption oracle
  – when the challenge state is reached, the interpretation of $\vec{x}$ and y as well as $\mathcal{A}_1^{\mathcal{O}}(\hat{\phi}, \vec{x})$ are computed
  – $\mathcal{A}_1^{\mathcal{O}}(\hat{\phi}, \vec{x})$ submitted to the encryption oracle along with a string of 0's of the same length
  – apply $\mathcal{A}_2^{\mathcal{O}}$ to whatever is received back.
  – the result given by $\mathcal{A}_2^{\mathcal{O}}$ is compared with the interpretation of y.

  Note that $\vec{x}, y \preccurlyeq \hat{\phi}$ ensures that $\vec{x}$ and y can be computed by simulating the protocol. Now, because of (1) and (2), on S'' the result is identical with y if and only if the encryption oracle encrypted the correct bit string and not the 0's. So on S'', it is possible to tell what the oracle encrypted. However, the attacker does not know when he is inside S'' and when outside. To overcome this problem, the CCA2 attacker does either of the following:

  – encrypts a string of 0's of the same length himself either once (or twice, fixed at the beginning which)
  – applies $\mathcal{A}_2^{\mathcal{O}}$ on this encryption (these ecryptions) too
  – if $\mathcal{A}_2^{\mathcal{O}}$ applied on the bit string received from the encryption oracle gives y back, and if on the encryption(s) of 0's done by the adversary $\mathcal{A}_2^{\mathcal{O}}$ fails to give y, then the CCA2 attacker outputs 1, meaning that his guess is that the oracle encrypted the correct bit string.
  – otherwise, that is, if $\mathcal{A}_2^{\mathcal{O}}$ applied on the bit string received from the encryption oracle does not give y back, or if on (at least one of) the encryption(s) of

0's done by the adversary $\mathcal{A}_2^{\mathcal{O}}$ gives y the adversary throws a coin and outputs the result.

According to our assumptions, on S'', a non-negligible set, if $\mathcal{A}_2^{\mathcal{O}}$ is applied on the correct encryption, it gives y back, if it is applied on the encryption of 0's, then it does not give y back. Hence, conditioned on this set, the CCA2 adversary always wins if the encryption oracle encrypts the correct bit string (probability $1/2$ conditioned on S''), and it wins by $1/2$ if the wrong bit string was encrypted (again $1/2$), because in this case a coin is tossed. Hence, conditioned on S'', the adversary wins by probability $1/2 + 1/2 \times 1/2 = 3/4$ This gives a $1/4$ advantage on S''. As S'' is non-negligible, $1/4$ of it is also non-negligible. However, what he gained inside S'', he may lose outside S''. To overcome this, the adversary is doing his own encryption of 0's once or twice as defined above. Clearly, inside S'', $\mathcal{A}_2^{\mathcal{O}}$ never gives y back applied on this encryption. Outside S'', it is possible that for certain values of the random input of the encryption of 0's $\mathcal{A}_2^{\mathcal{O}}$ gives y back, while on other values it does not give y back. In those cases, when $\mathcal{A}_2^{\mathcal{O}}$ does not give y back on the encryption of 0's computed by the adversary and $\mathcal{A}_2^{\mathcal{O}}$ gives y back on the encryption of 0's computed by the oracle, the adversary makes the wrong guess. Other than this, a coins is tossed, so the probability of wrong guesses other then this case is $1/2$. So we have to focus on the case when the mistake made in the above situation exactly cancels the advantage on S'', which is $1/4 \times p(S'')$. This is why we defined two different algorithms. If in one case, the above mistake exactly cancels the gain on S'', then in the other case it does not cancel. The reason is, that in the case when there is some non-negligible probability both for getting the correct y on an encryption of 0's and something different, the double checking of the encryption of 0's by the adversary himself will pull away the failure probability by a non-negligible factor relative to the case when it is encrypted only once. So either one of the two attacks will win, they cannot both fail.
The proof for KDM-CCA2 is exactly analogous. The only difference is, that instead of $\mathcal{A}_1^{\mathcal{O}}(\hat{\phi}, \vec{x})$, there is an x there, that the attacker of the KDM encryption may not be able to compute because it is a term that can also contain the key K. Other than this (and the encryption randomness), the rest is names, and handles, the attacker can himself generate while simulating the protocol. So when it comes to submitting it to the encryption oracle, it will simply be submitted as a function of the key. That is why it is enough to assume $x \preccurlyeq \hat{\phi}$. Note on the other hand that *the protocol adversary* need not be able to compute x (that would be $\hat{\phi} \rhd^{\mathcal{O}} x$).

- Fresh items do not corrupt: The idea is exactly the same as in case of the derivability predicate. A fresh item can just as well be created by the adversary, it cannot help him.

We now turn to the case of INT-CTXT key corruption.

Proofs of the first six axioms and the last one are entirely identical to the proofs for CCA2 key corruption. The soundness of the "encryptions with uncorrupted keys do not corrupt" axiom is also analogous: We again show the following:

$$\begin{aligned}&\mathsf{RanGen}(K) \wedge \mathsf{RanGen}(K') \wedge \mathsf{fresh}(R; \hat{\phi}, \vec{x}, x, K, K')\\ &\quad\wedge\ \vec{x}, x \preccurlyeq \hat{\phi}\ \wedge \hat{\phi}, \vec{x}, \{\!|x|\!\}_{K'}^{R} \blacktriangleright^{\mathrm{ic}} K\ \wedge\ \hat{\phi}, \vec{x}, x \not\blacktriangleright^{\mathrm{sic2}} K'\\ &\longrightarrow\ \hat{\phi}, \vec{x} \blacktriangleright^{\mathrm{ic}} K\end{aligned}$$

Again, we have to show that for all S non-negligible sets and σ evaluations of variables, if $\mathcal{M}, \Pi, S, \sigma$ satisfies the premise, then it satisfies the conclusion as well. So let us suppose it satisfies the premise. We want to show $\mathcal{M}, \Pi, S, \sigma \models \hat{\phi}, \vec{x} \blacktriangleright^{\mathrm{ic}} K$. Following the definition of key corruption, take any subset $S' \subseteq S$. By the definition of key corruption applied to $\mathcal{M}, \Pi, S, \sigma \models \hat{\phi}, \vec{x}, \{\!|x|\!\}_{K'}^{R} \blacktriangleright^{\mathrm{ic}} K$ there is a $S'' \subseteq S$, and a PT algorithm $\mathcal{A}^{\mathcal{O}^{\mathrm{sic2}}}$, such that

$$\begin{aligned}&\mathcal{M}, \Pi, S'', \sigma |\!\models^{c} \mathit{sdec}(\mathcal{A}^{\mathcal{O}^{\mathrm{sic2}}}(\hat{\phi}, \vec{x}, \{\!|x|\!\}_{K'}^{R}), K) \neq \bot\\ &\wedge \forall z R' (\mathcal{A}^{\mathcal{O}^{\mathrm{sic2}}}(\hat{\phi}, \vec{x}, \{\!|x|\!\}_{K'}^{R}) = \{\!|z|\!\}_{K}^{R'} \rightarrow \{\!|z|\!\}_{K}^{R'} \not\sqsubseteq \hat{\phi}, \vec{x}, \{\!|x|\!\}_{K'}^{R})\end{aligned}$$

and on S'', $\mathcal{A}^{\mathcal{O}^{\mathrm{sic2}}}(\hat{\phi}, \vec{x}, , \{\!|x|\!\}_{K'}^{R})$ is not equal any of the outputs of the encryption oracles. Now, we also have $\mathcal{M}, \Pi, S'', \sigma \models \hat{\phi}, \vec{x}, x \not\blacktriangleright^{\mathrm{sic2}} K'$ from the satisfaction of the premise. This gives us

$$\begin{aligned}&\mathcal{M}, \Pi, S'', \sigma |\!\models^{c} \mathit{sdec}(\mathcal{A}^{\mathcal{O}^{\mathrm{sic2}}}(\hat{\phi}, \vec{x}, \{\!|0^{|x|}|\!\}_{K'}^{R}), K) \neq \bot\\ &\wedge \forall z R' (\mathcal{A}^{\mathcal{O}^{\mathrm{sic2}}}(\hat{\phi},\vec{x},\{\!|0^{|x|}|\!\}_{K'}^{R}) = \{\!|z|\!\}_{K}^{R'} \rightarrow \{\!|z|\!\}_{K}^{R'} \not\sqsubseteq \hat{\phi},\vec{x},\{\!|0^{|x|}|\!\}_{K'}^{R})\end{aligned}$$

and on S'', $\mathcal{A}^{\mathcal{O}^{\mathrm{sic2}}}(\hat{\phi}, \vec{x}, , \{\!|0^{|x|}|\!\}_{K'}^{R})$ is not equal any of the outputs of the encryption oracles. Again, as the length of x can be guessed, there is a non-negligible $S''' \subseteq S''$ and a $\mathcal{B}^{\mathcal{O}^{\mathrm{sic2}}}$ such that

$$\begin{aligned}&\mathcal{M}, \Pi, S''', \sigma |\!\models^{c} \mathit{sdec}(\mathcal{B}^{\mathcal{O}^{\mathrm{sic2}}}(\hat{\phi}, \vec{x}), K) \neq \bot\\ &\wedge \forall z R' (\mathcal{B}^{\mathcal{O}^{\mathrm{sic2}}}(\hat{\phi}, \vec{x}) = \{\!|z|\!\}_{K}^{R'} \rightarrow \{\!|z|\!\}_{K}^{R'} \not\sqsubseteq \hat{\phi}, \vec{x})\end{aligned}$$

and on S''', $\mathcal{B}^{\mathcal{O}^{\mathrm{sic2}}}(\hat{\phi}, \vec{x})$ is not equal any of the outputs of the encryption oracles. And that exactly means $\mathcal{M}, \Pi, S, \sigma \models \hat{\phi}, \vec{x} \blacktriangleright^{\mathrm{ic}} K$. Again, the KDM case is entirely analogous.

Proof of the uncorrupted key's encryption cannot be faked axiom is immediate from the semantics of $\blacktriangleright^{\mathrm{ic}}$. If

$$\begin{aligned}\mathcal{M}, \Pi, S, \sigma \models &\mathsf{RanGen}(K)\ \wedge\ \hat{\phi}, \vec{x} \rhd y\ \wedge\ \mathit{dec}(y, dK) \neq \bot\\ &\wedge\ \forall x R (y = \{x\}_{eK}^{R} \rightarrow \{x\}_{eK}^{R} \not\sqsubseteq \hat{\phi}, \vec{x})\end{aligned}$$

Then for all $S' \subseteq S$, there is a $S'' \subseteq S'$ and an algorithm $\mathcal{A}$ such that $\mathcal{M}, \Pi, S''', \sigma |\!\models^{c} \mathcal{A}(\hat{\phi}, \vec{x}) = y$. Furthermore, the last conjunct means that the output of the algorithm is not any of the encryptions in $\hat{\phi}, \vec{x}$, and the third conjunct means the decryption does not fail. This is exactly means that $\mathcal{M}, \Pi, S, \sigma \models \hat{\phi}, \vec{x} \blacktriangleright^{\mathrm{ic}} K$.

The only remaining axiom is the fresh keys are not corrupted axiom for the INT-CTXT case. But that is rather easy. Suppose, the encryption is INT-CTXT secure. $\mathcal{M}, \Pi, S, \sigma \models \hat{\phi} \blacktriangleright^{\mathrm{ic}} K$ means there is a $S'' \subseteq S$, and a PT algorithm $\mathcal{A}^{\mathcal{O}^{\mathrm{sic2}}}$, such that

$$\begin{aligned}\mathcal{M}, \Pi, S'', \sigma \| \models^{c} & sdec(\mathcal{A}^{\mathcal{O}^{\mathrm{sic2}}}(\hat{\phi}), K) \neq \bot \\ & \wedge \forall z R' (\mathcal{A}^{\mathcal{O}^{\mathrm{sic2}}}(\hat{\phi}) = \{\!|z|\!\}_K^{R'} \to \{\!|z|\!\}_K^{R'} \not\sqsubseteq \hat{\phi})\end{aligned}$$

and on S'', $\mathcal{A}^{\mathcal{O}^{\mathrm{sic2}}}(\hat{\phi})$ is not equal any of the outputs of the encryption oracles. But that exactly means that there is a non-negligible set (namely S''), on which $\mathcal{A}^{\mathcal{O}^{\mathrm{sic2}}}$ can produce a ciphertext, contradicting the INT-CTXT property. ■