Deciding Secrecy of Security Protocols for an Unbounded Number of Sessions: The Case of Depth-bounded Processes

Emanuele D'Osualdo
University of Kaiserslautern, Germany
dosualdo@cs.uni-kl.de

Luke Ong
University of Oxford, UK
lo@cs.ox.ac.uk

Alwen Tiu
Nanyang Technological University, Singapore
atiu@ntu.edu.sg

***Abstract*—We introduce a new class of security protocols with an unbounded number of sessions and unlimited fresh data for which the problem of secrecy is decidable. The only constraint we place on the class is a notion of *depth-boundedness*. Precisely we prove that, restricted to messages of up to a given size, secrecy is decidable for all depth-bounded processes. This decidable fragment of security protocols captures many real-world symmetric key protocols, including Needham-Schroeder Symmetric Key, Otway-Rees, and Yahalom.**

I. Introduction

Security protocols are distributed programs that are designed to achieve secure communications using cryptography. They are extensively deployed today to improve the security of communication applications, ranging from electronic payments and internet banking to e-voting. However protocol design is notoriously error-prone. Because the financial and societal costs of failure are often prohibitively high, formal verification of security protocols is widely acknowledged as a necessity. In contrast to other safety critical systems, a distinctive feature of the security properties of protocols is that they must hold in the presence of an adversary or intruder, and this makes them challenging to verify. An important example of such a security property is *secrecy*: to verify that a protocol satisfies secrecy amounts to checking whether it can leak a given (secret) message to the environment as a result of interference by the intruder.

Known Undecidability and Decidability Results

In essence, to verify secrecy, we need a way of analysing the set of messages that the intruder knows; if a message does not belong to this set, then that message is not leaked. Here we assume a model of intruders as defined by Dolev and Yao [1]. The difficulty is that this set of messages is in general infinite, because the Dolev-Yao intruder is in control of three sources of infinity: a) messages of unbounded size, b) an unbounded set of nonces (and other freshly generated data such as session keys), and c) an unbounded number of sessions. Indeed the secrecy problem was proved to be undecidable by Durgin et al. [2]. Amadio et al. [3] and Heintze and Tygar [4] showed that the problem is undecidable even if the set of atomic terms is fixed and finite, assuming that terms of arbitrary size can be substituted for nonces.

Several decidability results have been obtained by restricting the three sources of infinity identified above. Durgin et al. [2] proved that secrecy is DEXPTIME-complete when both the number of nonces and the size of messages are bounded. Rusinowitch and Turuani [5] and Comon-Lundh et al. [6] proved that nonsecrecy is NP-complete when the number of sessions is bounded. Of course, analysing a protocol for a fixed finite number of sessions does not prove secrecy.

A direction of investigation which has proved fruitful does not constrain the above sources of infinity *a priori*, but restricts the format of messages, so that the encrypted messages become *context explicit*. In a pioneering paper [7], Lowe considered messages with encrypted components that are textually distinct, and each encrypted component must include all protocol roles. In a similar vein, Ramanujam and Suresh [8] (and subsuming [9]) introduced a non-unifiability condition: between any two terms occurring in distinct communications, no encrypted subterm of one can be unified with a subterm of the other. Ramanujam and Suresh [10] also studied more elaborate, dynamic schemes that tag encryptions with fresh nonces when composed during a run.

All these results assume somewhat ad hoc features; for example, the protocol must have an "honest run" (i.e. without intruder interaction), or satisfy subtle type constraints. Dissatisfied with the rather ad hoc nature of these models, Fröschle [11] introduced the class of *well-founded protocols* which is designed to exclude a common feature in all protocols used in undecidability proofs: the message format allows honest information to be propagated unboundedly without the intruder manipulating it.

Closely related to well-founded protocol is a recent decidability result for trace equivalence of security protocols for an unbounded number of sessions and unrestricted nonces by Chrétien et al. [12]. Equivalences can encode reachability properties such as (strong) secrecy. Chrétien et al. give a procedure for deciding trace equivalence for protocols that are simple (each process communicates on a distinct channel), type-compliant, and have acyclic dependency graphs. This class of protocols includes many tagged variants of symmetric key protocols in the literature.

Our Contributions

We show that the problem of secrecy is decidable for a class of security protocols with an unbounded number of sessions and unlimited fresh data. We use a cryptographic process calculus based on the π-calculus [13], but presented in a style that makes the intruder's knowledge base (*qua* a set of messages) explicit. I.e. a protocol and the intruder's knowledge base are represented together as a process term. As is standard, we use an inference system, $\Gamma \vdash M$ (meaning "intruder can derive message M from the set Γ of messages"), to define what the intruder knows. A technical innovation crucial to our decidability proof is the internalisation of the inference system as a congruence relation of the process terms, called *knowledge congruence*, written $\equiv_{\mathsf{kn}}$. By construction $\equiv_{\mathsf{kn}}$ contains the standard structural congruence; further it is sound and complete with respect to derivability, i.e., $M_1, \cdots, M_n \vdash M$ if, and only if,

$$\langle M_1 \rangle \parallel \cdots \parallel \langle M_n \rangle \equiv_{\mathsf{kn}} \langle M_1 \rangle \parallel \cdots \parallel \langle M_n \rangle \parallel \langle M \rangle$$

where $\langle M \rangle$ is the process representation of the message M. Every finite set of messages is knowledge-equivalent to a unique irreducible set of messages. Further, every process is knowledge congruent to a process in irreducible standard form.

Our decidability result relies on two process measures, and their corresponding notions of boundedness. The first is a measure on the size of messages. The second measure is *depth*, an adaptation of a concept introduced by Meyer [14] for the π-calculus. A subterm of a process has nesting of restriction $k \in \mathbb{N}$ just if it is in the scope of k restrictions; the nesting of restriction of a process is just the maximum nesting of restriction of its subterms. The *depth* of a process is then defined as the minimal nesting of restrictions in its knowledge congruence class.

Given $s, k \in \mathbb{N}$, we say that a process is (s,k)-*bounded* if all processes reachable from it have depth at most k, when only messages of size up to s are allowed. Intuitively, bounding depth means that, as a result of any interaction—be it honest or malicious—the protocol cannot produce what we call *encryption chains* of unbounded length $\{N_1\}_{N_2}, \{N_2\}_{N_3}, \ldots, \{N_{n-1}\}_{N_n}$, for secret nonces $N_1, \ldots, N_n$.

Our main result (Theorem 7) says that, restricted to messages of up to a given size s, secrecy is decidable for all (s,k)-bounded processes, for all $k \in \mathbb{N}$.

Besides the obvious applications, this result's relevance is chiefly conceptual: we show, for the first time, that limiting the length of the encryption chains is enough, together with the necessary message size limitation common to other approaches, to obtain decidability of secrecy. As importantly, we show how to technically implement the restriction so that only the relevant encryption chains are considered, and position the result in the literature.

Our proof of decidability is an application of the theory of well-structured transition system (WSTS) [15, 16]. Recall that the coverability problem of an effective WSTS is decidable. The main technical argument lies in the proof that with respect to the process reduction relation, and a notion of *knowledge embedding*, the set of processes reachable from a given (s,k)-bounded process forms an effective WSTS. Secrecy queries are then encoded as appropriate instances of the coverability problem.

The decidable fragment of security protocols that we have identified captures many real-world symmetric key protocols, including Needham-Schroeder Symmetric Key, Otway-Rees, and Yahalom. Using these and other examples, we show that our decidable class of protocols is incomparable to those in [11, 12]. The message algebra is deliberately chosen as the simplest that illustrates our approach. The techniques used in our decidability result are robust enough to handle asymmetric key cryptographic protocols as well.

Outline

In Section II, we present a formal model of the Dolev-Yao intruder. In Section III, we introduce our cryptographic process calculus. After presenting its syntax, knowledge congruence and operational semantics, we define the secrecy problem. In Section IV, we define the notion of depth, and (s,k)-bounded processes, and prove the main decidability result. We present in Section V a general method, amenable to automation, for proving that a protocol is (s,k)-bounded. Then in Section VI we examine several examples of real-world symmetric protocols and compare our model with those of Fröschle [11] and Chrétien et al. [12]. Finally in Section VII we briefly discuss future directions and conclude.

II. An Intruder Model

We give a formal model of the intruder, which describes the capabilities of the intruder in deducing messages from a given set of messages (e.g. those intercepted during a protocol run). We follow the Dolev-Yao model [1], which assumes perfect encryption, i.e., the intruder cannot decrypt an encrypted message without knowing the decryption key. For simplicity, we model only symmetric encryption/decryption, but the approach described here can be extended to cover other operators [17]. We model the intruder's capability using a proof system formalized in sequent calculus [17] so that we can reuse some results already established in [17], notably cut-elimination, to simplify proofs of some properties of our intruder model. We note however that our decidability results can be established using other representations of the intruder's capability, e.g., as natural deduction proof systems or as rewriting systems. The proof techniques used here are a straightforward adaptation from related work [17, 18]. We also provide some detailed proofs in Appendix A.

Let $\mathcal{N}$, ranged over by lowercase letters $a, b, \ldots$, be an enumerable set of names. The set $\mathbb{M}$ of messages is defined as the set of terms constructed as follows:

$$M, N, K ::= a \mid (M, N) \mid \{M\}_K$$

where (-, -) denotes the pairing operator and $\{M\}_K$ denotes the symmetric encryption of the message M with encryption

$$\frac{M \in \Gamma}{\Gamma \vdash M}\ \textsc{Id} \qquad \frac{\Gamma \vdash N \qquad \Gamma, N \vdash M}{\Gamma \vdash M}\ \textsc{Cut}$$

$$\frac{\Gamma, (M,N), M, N \vdash M'}{\Gamma, (M,N) \vdash M'}\ \mathrm{P}_L \qquad \frac{\Gamma \vdash M \qquad \Gamma \vdash N}{\Gamma \vdash (M,N)}\ \mathrm{P}_R$$

$$\frac{\Gamma, \{M\}_K \vdash K \qquad \Gamma, \{M\}_K, M, K \vdash N}{\Gamma, \{M\}_K \vdash N}\ \mathrm{E}_L$$

$$\frac{\Gamma \vdash M \qquad \Gamma \vdash K}{\Gamma \vdash \{M\}_K}\ \mathrm{E}_R$$

Figure 1. A proof system for Dolev-Yao intruders

key K. Projection and decryption are not modelled explicitly in the messages, but will instead be encoded via pattern matching in the operational semantics, which will be discussed later.

The set of names occurring in a set of messages Γ is denoted by $\mathrm{names}(\Gamma)$. The *size* of a message is defined as the height of its syntax tree, i.e. $\mathrm{size}(a) := 1$ and $\mathrm{size}((M,N)) := \mathrm{size}(\{M\}_N) := 1+\max(\mathrm{size}(M), \mathrm{size}(N))$. We write $\mathbb{M}_s^X$ for the set of messages with size bounded by s and names in $X \subseteq \mathcal{N}$, that is $\mathbb{M}_s^X := \{M \mid \mathrm{size}(M) \leq s, \mathrm{names}(M) \subseteq X\}$.

The intruder's capability is defined via the sequent calculus proof system given in Figure 1. This sequent system is a fragment of the sequent system for an intruder model in [17]. For a set Γ of messages, the sequent $\Gamma \vdash M$ means that the message M can be derived by using knowledge of the messages in Γ. As is standard, we write Γ, M to denote the set $\Gamma \uplus \{M\}$.

Lemma 1 (Cut-elimination [17])**.** *The cut rule is admissible.*

Lemma 2. *If $\Gamma \vdash M$ is provable then $\Gamma, \Gamma' \vdash M$ is provable for all Γ'.*

Lemma 3. *If $\Gamma, f(X,Y) \vdash M$ is provable, where f is either the pairing or the encryption operator, then $\Gamma, X, Y \vdash M$ is provable.*

Lemma 4. *Let a be a name that occurs neither in Γ nor in M, then if $\Gamma, a \vdash M$ is provable, $\Gamma \vdash M$ is also provable.*

Definition 1 (Knowledge order, equivalence)**.** Let Γ_1, Γ_2 be two sets of messages. We define the *knowledge quasi order* $\leq_{\mathsf{kn}}$ by: $\Gamma_1 \leq_{\mathsf{kn}} \Gamma_2$ if for all M, if $\Gamma_1 \vdash M$ then $\Gamma_2 \vdash M$. The two sets Γ_1 and Γ_2 are said to be *knowledge equivalent* (or simply *equivalent*) if $\Gamma_1 \leq_{\mathsf{kn}} \Gamma_2$ and $\Gamma_2 \leq_{\mathsf{kn}} \Gamma_1$. We write $\Gamma_1 \sim_{\mathsf{kn}} \Gamma_2$ when Γ_1 and Γ_2 are equivalent.

We want to define a canonical "core" subset of a set of messages Γ which is sufficient for deriving any message derivable from Γ. In order to do so, we introduce a simple rewriting system that simplifies a set of messages.

Definition 2. We define a rewrite relation $\longrightarrow$ on sets of messages as follows:

$$\frac{}{\Gamma, (M,N) \longrightarrow \Gamma, M, N}\ \mathrm{R}_P \qquad \frac{\Gamma, \{M\}_K \vdash K}{\Gamma, \{M\}_K \longrightarrow \Gamma, M, K}\ \mathrm{R}_E$$

Lemma 5. *The rewriting system in Definition 2 is terminating and confluent.*

Lemma 6. *If $\Gamma_1 \longrightarrow \Gamma_2$ then $\Gamma_1 \sim_{\mathsf{kn}} \Gamma_2$.*

Definition 3 (Irreducible set)**.** We say that Γ is *irreducible* if there exists no Γ' such that $\Gamma \longrightarrow \Gamma'$.

By Lemma 5, for every set of messages Γ there exists a unique irreducible Γ' such that $\Gamma \longrightarrow^* \Gamma'$; we call such a Γ' the irreducible form of Γ, denoted by $\mathrm{ird}(\Gamma)$. By Lemma 6, we have $\Gamma \sim_{\mathsf{kn}} \mathrm{ird}(\Gamma)$.

III. A Model of Cryptographic Protocols

We present a cryptographic process calculus based (ultimately) on the π-calculus [13, 19–23]. Fix a finite signature $\mathcal{Q}$ of process names (ranged over by Q) each of which has a fixed arity $\mathrm{ar}(\mathsf{Q}) \in \mathbb{N}$. The syntax of processes follows the grammar:

$$\begin{array}{ll} P ::= \mathbf{0} \mid \nu x.P \mid P \parallel P \mid \langle M \rangle \mid \mathsf{Q}[\vec{x}] & \text{process} \\ A ::= \mathbf{in}(\vec{x} : M).P \mid A + A & \text{action} \end{array}$$

We use the vector notation for names $\vec{x} = x_1, \ldots, x_n$ (for some $n \in \mathbb{N}$ which is left unspecified when irrelevant). We reserve $\vec{x}, \vec{y}, \ldots$ for bound names, with the convention that the names in a vector of bound names are all pairwise distinct. We often abuse notation by writing vectors for the set of their components, e.g. $a \in \vec{b}$ reads $a \in \{b_i \mid \vec{b} = b_1, \ldots, b_n, 1 \leq i \leq n\}$. The notation $\nu\vec{x}.P$ is a shorthand for $\nu x_1. \cdots \nu x_n.P$. If $\Gamma = \{M_1, \ldots, M_k\}$ is a finite set of messages, then $\langle \Gamma \rangle := \langle M_1 \rangle \parallel \ldots \parallel \langle M_k \rangle$.

In an action $\mathbf{in}(\vec{x} : M).P$ we call $\vec{x} : M$ the *pattern*, $\mathbf{in}(\vec{x} : M)$ the *input prefix* and P the *continuation*, and we say P is *under a prefix*. A subterm Q of a term P is called *active* in P if it is not under a prefix. The process $\langle M \rangle$ is called an *active message* and is essentially a degenerate form of the notion of an active substitution in the applied π-calculus [19], where we omit the domain of the substitution. An active message is essentially a message output by a process that is captured by the environment (intruder). Processes of the form $\langle M \rangle$ or $\mathsf{Q}[\vec{a}]$ are called *sequential*. The internal action $\boldsymbol{\tau}$, can be understood as an abbreviation for $\mathbf{in}(x : x)$, for a fresh x.

The calculus has two binders: the names $\vec{x}$ are bound in both $\nu\vec{x}.P$ and $\mathbf{in}(\vec{x} : M).P$. We denote the set of free names of a term P with $\mathrm{fn}(P)$ and the set of bound names with $\mathrm{bn}(P)$. As is standard, α-conversion can be used to rename bound names to fresh names without changing the structure of a term. Therefore, we require, wlog, that $\mathrm{fn}(P) \cap \mathrm{bn}(P) = \emptyset$.

A program consists of an initial term P and a finite set Δ of definitions of the form $\mathsf{Q}[x_1, \ldots, x_n] := A$, with $\mathrm{ar}(\mathsf{Q}) = n$ and $\mathrm{fn}(A) \subseteq \{x_1, \ldots, x_n\}$. Notice that we require every definition to be prefixed by an action. This simplifies the technical developments without sacrificing expressivity.

We assume there is at most one definition for each Q in $\mathcal{Q}$. We write $\mathbb{P}$ for the set of all processes over an underlying signature $\mathcal{Q}$.

Pattern matching is formalised using substitutions of bound names to messages. We write $\sigma = [\,M_1/x_1,\ \dots,\ M_n/x_n\,]$, often abbreviated with $[\,\vec{M}/\vec{x}\,]$, for the substitution with $\sigma(x_i) = M_i$ for all $1 \leq i \leq n$.

Definition 4. A pattern $\vec{x} : M$ is said to be *admissible*, if for all $y \in \vec{x}$ we have $M, \vec{z} \vdash y$ where $\vec{z} = \mathrm{fn}(M) \setminus \vec{x}$.

We require every pattern in a program to be admissible. Thanks to admissible patterns, we can model decryption and projections using pattern matching. For instance, an action $\mathbf{in}(x, y : \{(x, y)\}_k).P$ can match a message $\{(a, \{b\}_c)\}_k$ resulting in $P[\,a/x,\ \{b\}_c/y\,]$; note that the encryption key k is not bound so it is already known by the action: the same action cannot match a message $\{(a, b)\}_{k'}$ where $k' \neq k$.

Admissible patterns rule out patterns that bind encryption keys so that decryption is possible only if the required keys are known. For instance, the pattern in the previous example is admissible because $\{(x, y)\}_k, k \vdash x$, and $\{(x, y)\}_k, k \vdash y$, while the pattern $x, y : \{(x, b)\}_y$ is not admissible because $\{(x, b)\}_y, b \not\vdash y$. Note that multiple decryptions and projections can be modelled by a single admissible pattern, for example $x, y, z : \{(x, (\{y\}_{(z,x)}, z))\}_k$ can be implemented by first decrypting the message with the known key k, obtaining (by projection) $x, \{y\}_{(z,x)}$ and z, and then pairing z and x to obtain the key needed to decrypt y.

Remark 1. We choose to adopt a variant of π-calculus with guarded choice ($A + A$) and guarded recursive definitions. A common alternative is to omit choice and use replication ($!P$) instead of definitions. Choice is not strictly necessary to specify common protocols and can indeed be simulated in the calculus without choice. Similarly, guarded replication can be simulated by using definitions: $P = !\big(\sum_{i \in I} \mathbf{in}(\vec{y}_i : M_i).Q_i\big)$, with $\vec{x} = \mathrm{fn}(P)$, can be encoded by a definition $\mathsf{Rep}_P[\vec{x}] := \sum_{i \in I} \mathbf{in}(\vec{y}_i : M_i).(Q_i \parallel \mathsf{Rep}_P[\vec{x}])$. It may seem that our undecidability proofs (see Theorems 1 and 2) are made weaker by the adoption of these powerful primitives, but they can be replicated in weaker calculi, as studied more thoroughly in the literature (see for example [3] or [24, §3.3]). The undecidability results rely on folklore encodings, briefly presented to motivate the restriction that we use to prove our main decidability result (Theorem 7). In this respect, the adoption of such rich primitives strenghtens our result and allows a more natural representation for common protocols.

A. Structural and Knowledge Congruences

Structural congruence, $\equiv$, is the least relation that respects α-conversion of bound names, and is associative and commutative with respect to $\parallel$ and $+$ with $\mathbf{0}$ as the neutral element, and satisfies the laws:

$$\begin{aligned} \nu a.\mathbf{0} &\equiv \mathbf{0} \\ \nu a.\nu b.P &\equiv \nu b.\nu a.P \\ P \parallel \nu a.Q &\equiv \nu a.(P \parallel Q) \quad (\text{if } a \notin \mathrm{fn}(P)) \end{aligned}$$

The second law is called *exchange*, the third *scope extrusion*.

It is easy to show that every process is congruent to a process $\mathrm{sf}(P)$ in *standard form*, i.e. of the form

$$\nu\vec{x}.\big(\langle M_1\rangle \parallel \cdots \parallel \langle M_m\rangle \parallel \mathsf{Q}_1[\,\vec{N}_1\,] \parallel \cdots \parallel \mathsf{Q}_k[\,\vec{N}_k\,]\big) \qquad (*)$$

where every name in $\vec{x}$ occurs free in some subterm. When we write $\mathrm{sf}(P) = \nu\vec{x}.(\langle\vec{M}\rangle \parallel Q)$ we implicitly assert that Q is a parallel composition of processes $\mathsf{Q}[\,\vec{N}\,]$; and all the active messages are collected in $\vec{M}$. Note that $\vec{M}$ represents all the messages that have been sent through the insecure communication medium. They are therefore both available to the intended receiver, and leaked to the intruder. We define $\mathrm{msg}(P) = \vec{M} \cup \vec{N}_1 \cup \ldots \cup \vec{N}_k$ when $\mathrm{sf}(P)$ is the expression ($*$). Thus $\mathrm{msg}(P)$ is the set of messages appearing in a term. When $m = 0, k = 0, \vec{x} = \emptyset$, the expression ($*$) is $\mathbf{0}$.

It is useful to incorporate the knowledge reduction (Definition 2) in the structural congruence so that the irreducible forms are canonical members of the congruence classes. The *knowledge congruence* relation is obtained by extending structural congruence with the rewriting relation of Definition 2. Formally, the knowledge congruence $\equiv_{\mathsf{kn}}$ is the smallest congruence that includes $\equiv$ and satisfies:

$$\begin{aligned} \langle(M, N)\rangle &\equiv_{\mathsf{kn}} \langle M\rangle \parallel \langle N\rangle && \textit{Diffusion} \\ \langle\{M\}_K\rangle \parallel \langle K\rangle &\equiv_{\mathsf{kn}} \langle M\rangle \parallel \langle K\rangle && \textit{Decryption} \\ \langle M\rangle \parallel \langle M\rangle &\equiv_{\mathsf{kn}} \langle M\rangle && \textit{Persistence} \end{aligned}$$

Persistence is a consequence of the ability of the intruder to duplicate messages: as reflected by the semantics, receiving a message does not consume it, the intruder needs to acquire it only once to be able to replay it at any time.

Lemma 7. *If $\Gamma \vdash M$ then there is a $\Gamma' \supseteq \Gamma$ with $\langle\Gamma\rangle \equiv_{\mathsf{kn}} \langle\Gamma'\rangle$ and $M \in \Gamma'$.*

As a consequence, knowledge congruence is sound and complete with respect to derivability.

Corollary 1. *$\Gamma \vdash M$ if and only if $\langle\Gamma\rangle \equiv_{\mathsf{kn}} \langle\Gamma\rangle \parallel \langle M\rangle$.*

Definition 5. We say that a process P is *irreducible* if the active messages of its standard form are an irreducible set of messages, that is: $\mathrm{sf}(P) = \nu\vec{x}.(\langle\Gamma\rangle \parallel Q)$ and $\Gamma = \mathrm{ird}(\Gamma)$.

Every process P is knowledge congruent to a process in *irreducible standard form*, written $\mathrm{isf}(P)$, defined as $\mathrm{isf}(P) := \nu\vec{x}.(\langle\mathrm{ird}(\Gamma)\rangle \parallel Q)$, where $\mathrm{sf}(P) = \nu\vec{x}.(\langle\Gamma\rangle \parallel Q)$.

Lemma 8. *For all P, $\mathrm{sf}(P) \equiv_{\mathsf{kn}} \mathrm{isf}(P)$.*

Proof. Assume $\Gamma = \Gamma_1 \longrightarrow \ldots \longrightarrow \Gamma_n = \mathrm{ird}(\Gamma)$. Let $\mathrm{sf}(P) = \nu\vec{x}.(\langle\Gamma\rangle \parallel Q)$, then $\mathrm{isf}(P) = \nu\vec{x}.(\langle\mathrm{ird}(\Gamma)\rangle \parallel Q)$. We show that every rewriting step $\Gamma_i \longrightarrow \Gamma_{i+1}$, for $1 \leq i \leq n$, can be replicated using knowledge congruence laws. If $\Gamma_i \longrightarrow \Gamma_{i+1}$

is due to an application of Rule R_P then $\langle\Gamma_i\rangle \equiv_{\mathsf{kn}} \langle\Gamma_{i+1}\rangle$ by the diffusion law. If $\Gamma_i \longrightarrow \Gamma_{i+1}$ is justified by Rule R_E, then $\Gamma_i = \Gamma', \{M\}_K$, $\Gamma_{i+1} = \Gamma', M, K$ and $\Gamma_i \vdash K$. By Corollary 1, $\langle\Gamma_i\rangle \equiv_{\mathsf{kn}} \langle\Gamma_i\rangle \parallel \langle K\rangle$ and therefore $\langle\Gamma_i\rangle \equiv_{\mathsf{kn}} \langle\Gamma', \{M\}_K, K\rangle \equiv_{\mathsf{kn}} \langle\Gamma', M, K\rangle = \langle\Gamma_{i+1}\rangle$ $\square$

B. Operational Semantics

A process consists of a number of sequential processes (modelling the principals of the protocol) and some messages that have been communicated through a global, insecure medium. An input action can be fired if the intruder can produce a message that matches the action's pattern. Since the intruder can also replay genuine messages as they are, this interaction models both legitimate and malicious communications. The result of firing an action is the activation of its continuation: this can have the effect of changing the state of the principal, asynchronously sending new messages and creating new principals.

We write $\mathsf{Q}[\vec{M}] \triangleq A$ if $\mathsf{Q}[\vec{x}] := A' \in \Delta$ and $A = A'[\vec{M}/\vec{x}]$, up to commutativity and associativity of $+$.

We formalise the calculus' semantics by defining the transition relation $\to_\Delta$ between processes, dependent on the definitions Δ: $P \to_\Delta Q$ holds if

1) $P \equiv_{\mathsf{kn}} \nu\vec{a}.(\langle\Gamma\rangle \parallel \mathsf{Q}[\vec{M}'] \parallel C)$,
2) $\mathsf{Q}[\vec{M}'] \triangleq \mathbf{in}(\vec{x} : N).P' + A$,
3) $\Gamma, \vec{c} \vdash N[\vec{M}/\vec{x}]$, for some messages $\vec{M}$ and fresh names $\vec{c} = \mathrm{names}(\vec{M}) \setminus (\vec{a} \cup \mathrm{fn}(P))$, and
4) $Q \equiv_{\mathsf{kn}} \nu\vec{a}.\nu\vec{c}.(\langle\Gamma\rangle \parallel \langle\vec{c}\rangle \parallel P'[\vec{M}/\vec{x}] \parallel C)$.

We often omit Δ when it is clear from the context.

Notice how the intruder can inject fresh names $\vec{c}$ (which can be used as keys) and use them in conjunction with the leaked knowledge Γ to produce messages that can be consumed by a principal, who is unable to distinguish the genuine messages from the counterfeit. That the matching message is *derived* from Γ is worth noting; it means that decryption and projections can simply be implemented by pattern matching.

The processes that are reachable from P, under the definitions in Δ, form the set $\mathrm{Reach}_\Delta(P) := \{Q \mid P \to_\Delta^* Q\}$. The set of traces, i.e. transition sequences, from P under Δ is the set

$$\mathrm{Traces}_\Delta(P) := \{Q_0 \cdots Q_n \mid P \equiv_{\mathsf{kn}} Q_0 \to_\Delta \cdots \to_\Delta Q_n\}.$$

C. The Secrecy Problem

We are interested in checking whether a protocol is secure, which, for the purposes of this paper, amounts to checking whether a secret can be leaked to the environment as the result of interference by the intruder. To mark a name as a secret, we assume $\mathcal{Q}$ contains a special symbol Secret of arity 1, with the convention that if a process $\mathsf{Secret}[M]$ and the message $\langle M\rangle$ become active, the secret M is leaked.

Definition 6 (Secrecy)**.** The *secrecy* problem asks, given a set of definitions Δ, a process P, and a message M, if there is a process Q such that $P \to_\Delta^* Q$ and $Q \equiv_{\mathsf{kn}} \nu\vec{x}.(\mathsf{Secret}[M] \parallel \langle M\rangle \parallel Q')$ with $\vec{x} \cap \mathrm{fn}(P) = \emptyset$. We call such Q a *leak*. We call *secrecy relative to* $\mathbb{C}$ the same problem where all the processes in the transitions to reach the leak (and the leak itself) are required to be in the class of processes $\mathbb{C}$.

We now show how informal specifications of protocols can be encoded in our process calculus and how secrecy queries can be encoded as coverability problems. In the following, we use the notation $\prod_{i\in I} P_i$, for some index set I, to denote parallel compositions of processes P_i.

Example 1 (Needham-Schroeder Symmetric Key (NSSK) protocol). The well-known Needham-Schroeder symmetric key establishment protocol [25] can be described as follows:

$$\begin{array}{lll}
(1) & A \to S: & A, B, N_A\\
(2) & S \to A: & \{N_A, B, K_{AB}, \{K_{AB}, A\}_{K_{BS}}\}_{K_{AS}}\\
(3) & A \to B: & \{K_{AB}, A\}_{K_{BS}}\\
(4) & B \to A: & \{N_B\}_{K_{AB}}\\
(5) & A \to B: & \{N_B - 1\}_{K_{AB}}
\end{array}$$

Here N_A, N_B are nonces, and K_{AB} is a fresh session key. K_{BS} and K_{AS} are long-term symmetric keys shared by the server and A and B, respectively.

In Fig. 2 we present a model of the protocol using our process calculus. Notice the definition of A_1 is prefixed with a $\boldsymbol{\tau}$ action to fit our syntax for definitions. Here we use the abbreviation $(M_1, M_2, \ldots, M_n)$ for $(M_1, (M_2, \cdots (M_{n-1}, M_n) \cdots))$. We also omit the pairing brackets inside an encryption. Since our calculus does not have arithmetic operators, the term $N_B - 1$ in the response from A to B in the last step is modelled as a pair (N_B, N_B). The definitions in the figure correspond to different steps for the roles (A, B and S) in the protocol. The initial configuration of the protocol is $(NS_{\mathcal{P}} \parallel \prod_{(x,k)\in\mathcal{P}} \langle x\rangle)$ where

$$NS_{\mathcal{P}} = \prod_{\substack{(x,k_{xs})\in\mathcal{P}\\(y,k_{ys})\in\mathcal{P}\setminus\{(x,k_{xs})\}}} (S_1[x, y, k_{xs}, k_{ys}] \parallel \mathsf{A}_1[x, y, k_{xs}] \parallel \mathsf{B}_1[x, y, k_{ys}])$$

and $\mathcal{P}$ is the set of principals and shared keys between the principals and the server. For example, with $\mathcal{P} = \{(a, k_{as}), (b, k_{bs})\}$ the above process represents the protocol with two honest principals a and b, sharing, respectively, the symmetric keys k_{as} and k_{bs} with the server.

Example 2 (NSSK with dishonest principal). In the encoding in Example 1, the principals in $\mathcal{P}$ are considered honest principals. If we want to model the case where the intruder can be among the principals, the protocol specification would be $(NS_{\mathcal{P}} \parallel \prod_{(x,k)\in\mathcal{P}} \langle x\rangle \parallel \langle k_{is}\rangle)$ where $(i, k_{is}) \in \mathcal{P}$ and $NS_{\mathcal{P}}$ is defined as in Example 1. Here the free name i represents the intruder. Note that we expose the shared key k_{is} between i and the server to the environment.

Example 3 (Secrecy queries). To reason about the secrecy of certain messages, we mark them with the label Secret. For example, to reason about the secrecy of N_B (Example 2, where the intruder is also a principal of the protocol), we modify the definition for B_2 in Figure 2 as follows:

$$\mathsf{B}_2[x, y, k_{ys}, k, s] := \mathbf{in}(\{s, s\}_k).\mathsf{Secret}[(x, y, s)]$$

$$
\begin{aligned}
\mathsf{S}_1[\,x, y, k_{xs}, k_{ys}\,] &:= \mathbf{in}(n_x : (x, y, n_x)).\nu k.\langle\{n_x, y, k, \{k, x\}_{k_{ys}}\}_{k_{xs}}\rangle \parallel \mathsf{S}_1[\,x, y, k_{xs}, k_{ys}\,]\\
\mathsf{A}_1[\,x, y, k_{xs}\,] &:= \boldsymbol{\tau}.\nu n.(\langle(x, y, n)\rangle \parallel \mathsf{A}_2[\,x, y, k_{xs}, n\,] \parallel \mathsf{A}_1[\,x, y, k_{xs}\,])\\
\mathsf{A}_2[\,x, y, k_{xs}, n\,] &:= \mathbf{in}(m, k : \{n, y, k, m\}_{k_{xs}}).(\langle m\rangle \parallel \mathsf{A}_3[\,x, y, k_{xs}, k\,])\\
\mathsf{A}_3[\,x, y, k_{xs}, k\,] &:= \mathbf{in}(s : \{s\}_k).\langle\{s, s\}_k\rangle\\
\mathsf{B}_1[\,x, y, k_{ys}\,] &:= \mathbf{in}(k : \{k, x\}_{k_{ys}}).\nu s.(\langle\{s\}_k\rangle \parallel \mathsf{B}_2[\,x, y, k_{ys}, k, s\,]) \parallel \mathsf{B}_1[\,x, y, k_{ys}\,]\\
\mathsf{B}_2[\,x, y, k_{ys}, k, s\,] &:= \mathbf{in}(\{s, s\}_k).\mathbf{0}
\end{aligned}
$$

Figure 2. An encoding of the symmetric Needham-Schroeder protocol.

Secrecy of nonce N_B in an exchange between honest participants A and B can then be specified as the problem of finding a process Q such that $P \to_{\Delta}^{*} Q$ and $Q \equiv_{\mathsf{kn}} \nu\vec{x}, s.(\mathsf{Secret}[\,(a, b, s)\,] \parallel \langle(a, b, s)\rangle \parallel Q')$. Here it is important that we associate the secret with the honest principals a and b, as otherwise the secrecy property will be trivially violated, e.g., if the intruder initiates a honest session (as itself) with one of the honest principals.

Example 4 (Replay attack). The NSSK protocol in Example 1 is vulnerable to a replay attack [26]. That is, if a past session key K_{AB} is known to the intruder, then the intruder (impersonating B) can replay Step (3) to A, tricking A into accepting the old session key, and use it to decrypt $\{N_B\}_{K_{AB}}$ to obtain N_B. To reason about replay attacks, we can modify the definition of B_2 as follows:

$$
\mathsf{B}_2[x, y, k_{ys}, k, s] := \begin{array}{l}\mathbf{in}(\{s, s\}_k).\mathsf{Secret}[\,(x, y, s)\,] +\\ \mathbf{in}(\{s, s\}_k).\langle k\rangle\end{array}
$$

That is, at the last step of the protocol, B can non-deterministically choose to mark the nonce N_B as secret or to expose the current session key.

D. Undecidability of the Secrecy Problem

Before we turn to our decidability results, we justify the restrictions we will impose (in Section IV) to prove decidability of secrecy, by showing that the problem is undecidable in the general case. To this end, it is sufficient to encode a 2-counter Minsky machine (2MM) as a cryptographic protocol and get undecidability of secrecy as a consequence of undecidability of control-state reachability for 2MM. Though such reductions are not new (see e.g. [3, 24]), we present two encodings of a counter with different properties which will be useful for the rest of the paper.

A first encoding exploits the fact that if no bound on the size of the messages is enforced, then the value of a counter can be encoded in the size of a message: a counter c_i is encoded by some process $\langle\{\{\{\ldots\{a\}_a\ldots\}_a\}_a\}_{c_i}\rangle$ where the number of encryptions with a represents the current value of the counter. The encoding, detailed in Appendix C, takes a 2MM $\mathcal{A}$ and produces $\mathcal{S}[\![\mathcal{A}]\!] = (P_{\mathcal{A}}, \Delta_{\mathcal{A}})$, with the property that a program location i is reachable in $\mathcal{A}$ if and only if from $P_{\mathcal{A}}$ one can reach a process containing $\mathsf{S}_\mathsf{i}[\,\vec{x}\,]$, for some $\vec{x}$.

Theorem 1. *The secrecy problem for unconstrained processes is undecidable.*

Since the above encoding relies on messages of unbounded size, it is common to put an artificial bound on the allowed size. As is well-known, this restriction is not enough to get a decidable secrecy problem. Although the result is not new, we think it informative to examine a proof. The undecidability result cannot be derived from the previous encoding: bounding the size of messages corresponds to bounding the counters' value, which yields a decidable control-state reachability problem. We therefore adapt the above encoding to reduce control-state reachability of 2MM to secrecy of a protocol where messages have bounded size. Specifically our encoding will only require messages of size no greater than 2. The key idea is to encode a natural number n with what we call an *encryption chain* of length n:

$$
EC_n = \nu x_1, \ldots, x_n.\langle\{z\}_{x_1}, \{x_1\}_{x_2}, \{x_2\}_{x_3}, \cdots, \{x_n\}_k\rangle.
$$

The value n is encoded by the number of decryptions needed to reach a secret z from knowledge of k. Using this idea, a 2MM $\mathcal{A}$ is encoded into $\mathcal{D}[\![\mathcal{A}]\!] = (P'_{\mathcal{A}}, \Delta'_{\mathcal{A}})$ with the property that a program location i of $\mathcal{A}$ is reachable if and only if one can reach, from $P'_{\mathcal{A}}$ a process containing $\mathsf{D}_\mathsf{i}[\,\vec{x}\,]$, for some $\vec{x}$.

Let $\mathbb{S}_s := \{P \in \mathbb{P} \mid \forall m \in \mathrm{msg}(P)\colon\ \mathrm{size}(m) \leq s\}$ be the set of processes containing active messages of size at most s. Since for any 2MM $\mathcal{A}$ with encoding $\mathcal{D}[\![\mathcal{A}]\!] = (P, \Delta)$ we have $\mathrm{Reach}_{\Delta}(P) \subseteq \mathbb{S}_2$, we conclude the following.

Theorem 2. *Secrecy relative to $\mathbb{S}_s$ is undecidable, for any $s \geq 2$.*

The case with $s < 2$ is hardly relevant as no encryption is then possible. The result motivates the restriction presented in the following section, which has the effect of bounding the length of the encryption chains a protocol can create.

IV. Depth Boundedness

Motivated by the undecidability results of the previous section, we introduce a measure on processes, their *depth*, adapting a concept introduced by Meyer [14, 27] for the π-calculus. By bounding the depth of processes we will be able to rule out encryption chains of unbounded length.

A. Depth and Notions of Boundedness

Definition 7 (Depth)**.** The *nesting of restrictions* of a term is given by the function

$$\begin{aligned} \mathrm{nest}_{\mathsf{v}}(\mathsf{Q}[\vec{a}]) := \mathrm{nest}_{\mathsf{v}}(\langle M\rangle) := \mathrm{nest}_{\mathsf{v}}(\mathbf{0}) &:= 0 \\ \mathrm{nest}_{\mathsf{v}}(\nu x.P) &:= 1 + \mathrm{nest}_{\mathsf{v}}(P) \\ \mathrm{nest}_{\mathsf{v}}(P \parallel Q) &:= \max(\mathrm{nest}_{\mathsf{v}}(P), \mathrm{nest}_{\mathsf{v}}(Q)). \end{aligned}$$

The *depth* of a term is defined as the minimal nesting of restrictions in its knowledge congruence class,

$$\mathrm{depth}(P) := \min\{\mathrm{nest}_{\mathsf{v}}(Q) \mid Q \equiv_{\mathsf{kn}} P\}.$$

It is important to note that here we measure depth modulo the coarser knowledge congruence (as opposed to structural congruence), which enables us to "cast off" encryption chains that are not essential. For example, consider

$$P = \nu a, b, c.(\langle a\rangle \parallel \langle\{b\}_a\rangle \parallel \langle\{c\}_b\rangle \parallel \langle c\rangle)$$

we have $\mathrm{nest}_{\mathsf{v}}(P) = 3$; however, by minimising the scopes of the restrictions using the exchange and scope extrusion laws, we can obtain the term

$$Q = \nu b.\big(\nu a.(\langle a\rangle \parallel \langle\{b\}_a\rangle) \parallel \nu c.(\langle\{c\}_b\rangle \parallel \langle c\rangle)\big)$$

with $Q \equiv P$ and $\mathrm{nest}_{\mathsf{v}}(Q) = 2$. The reader can check that every other process structurally congruent to P has nesting of restriction at least 2. But if one admits knowledge congruence, the term P can be transformed to the irreducible process $P' = (\nu a.\langle a\rangle \parallel \nu b.\langle b\rangle \parallel \nu c.\langle c\rangle)$ with $P \equiv_{\mathsf{kn}} P'$ and $\mathrm{nest}_{\mathsf{v}}(P') = 1$, from which it follows that $\mathrm{depth}(P) = 1$.

Example 5. Consider again the encryption chain EC_n introduced in the previous section. For any n the term EC_n is irreducible, and has $\mathrm{nest}_{\mathsf{v}}(EC_n) = n$. It is however possible to extrude the restrictions and get a shallower nesting by minimising the scopes starting from the restriction in the middle of the chain and proceeding recursively in the two halves. For example $EC_3 \equiv EC'_3$ where

$$EC'_3 = \nu x_2.\big(\nu x_1.\langle\{z\}_{x_1}, \{x_1\}_{x_2}\rangle \parallel \nu x_3.\langle\{x_2\}_{x_3}, \{x_3\}_k\rangle\big)$$

for which $\mathrm{nest}_{\mathsf{v}}(EC'_3) = 2$. In general it can be proven that $\mathrm{depth}(EC_n) = \lceil\log_2(n)\rceil$. This means that if a set of processes X contains, for any $n \in \mathbb{N}$, a term having EC_m with $m \geq n$ as a subterm up to congruence, then there is no finite upper bound to the depth of the terms in X.

Lemma 9. *For every process P there exists an irreducible process $Q \equiv_{\mathsf{kn}} P$ with $\mathrm{nest}_{\mathsf{v}}(Q) = \mathrm{depth}(P)$.*

Proof. Let $k = \mathrm{depth}(P)$. By definition of depth, we know there is a $P' \equiv_{\mathsf{kn}} P$ with $\mathrm{nest}_{\mathsf{v}}(P') = k$. If P' is irreducible we are done. Otherwise, it can be made irreducible by replacing some of its active messages with some of their subterms as dictated by Definition 2. Since the replacement can be made in-place, and it does not affect restrictions, we obtain an irreducible $Q \equiv_{\mathsf{kn}} P'$ with the same nesting $\mathrm{nest}_{\mathsf{v}}(Q) = \mathrm{nest}_{\mathsf{v}}(P') = k$. □

The following easy lemma will be useful later; it states that since names can be reused in non-nested restrictions, the nesting of restrictions of a term is a bound on the number of distinct names needed to represent the term.

Lemma 10. *Every P can be α-converted into a process Q such that $|\mathrm{bn}(Q)| \leq \mathrm{nest}_{\mathsf{v}}(P)$.*

Proof. We proceed by induction on the structure of P. For the base cases, where P is $\mathbf{0}$ or a sequential process, $\mathrm{nest}_{\mathsf{v}}(P) = 0$ and indeed P does not contain restrictions, so $|\mathrm{bn}(P)| = 0$. For the induction step, if $P = \nu a.P'$, by induction hypothesis we can α-convert P' to Q' so that $|\mathrm{bn}(Q')| \leq \mathrm{nest}_{\mathsf{v}}(P')$, ensuring wlog that $a \notin \mathrm{bn}(Q')$. Then $\nu a.Q'$ is the required process. If $P = P_1 \parallel P_2$ then, by induction hypothesis for each $i \in \{1, 2\}$ we can α-convert P_i to Q_i so that $|\mathrm{bn}(Q_i)| \leq \mathrm{nest}_{\mathsf{v}}(P_i)$ and we can do so by ensuring that $\mathrm{bn}(Q_1) \subseteq \mathrm{bn}(Q_2)$ or $\mathrm{bn}(Q_2) \subseteq \mathrm{bn}(Q_1)$. So we have P is α-equivalent to $Q_1 \parallel Q_2$ and $|\mathrm{bn}(Q_1 \parallel Q_2)| = |\mathrm{bn}(Q_1) \cup \mathrm{bn}(Q_2)| = \max(|\mathrm{bn}(Q_1)|, |\mathrm{bn}(Q_2)|) \leq \max(\mathrm{nest}_{\mathsf{v}}(P_1), \mathrm{nest}_{\mathsf{v}}(P_2)) = \mathrm{nest}_{\mathsf{v}}(P)$. □

For example, the term EC'_3 from Example 5 can be α-converted to $\nu x_2.\big(\nu x_1.\langle\{z\}_{x_1}, \{x_1\}_{x_2}\rangle \parallel \nu x_1.\langle\{x_2\}_{x_1}, \{x_1\}_k\rangle\big)$.

Let $\mathbb{D}_k^X := \{P \in \mathbb{P} \mid \mathrm{depth}(P) \leq k, \mathrm{fn}(P) \subseteq X\}$ be the set of processes of depth at most $k \in \mathbb{N} \cup \{\omega\}$ and with free names in X. Since our results do not depend on the choice of X we write $\mathbb{D}_k$ to mean $\mathbb{D}_k^X$ for some finite $X \subseteq \mathcal{N}$. Note that in both $\mathbb{D}_k^X$ and $\mathbb{S}_s^X$, the non-trivial restriction is on depth and size respectively, while the condition on the free names can typically be satisfied by setting X to the free names of the initial term. Let A^* be the set of sequences of elements of A.

Definition 8. Let $s, k \in \mathbb{N} \cup \{\omega\}$. We say the process P is (s, k)*-bounded* (w.r.t. a finite set Δ of definitions) if $\mathrm{Traces}_\Delta(P) \cap \mathbb{S}_s^* \subseteq \mathbb{D}_k^*$, i.e. from P only processes of depth at most k can be reached, when only messages of size up to s are allowed.

First we note that bounding the depth and not the message size is not enough to get a decidable secrecy property thanks to the encoding $\mathcal{S}[\![\text{-}]\!]$ presented in Section III-C. Since for any 2MM $\mathcal{A}$ with $\mathcal{S}[\![\mathcal{A}]\!] = (P, \Delta)$ we have $\mathrm{Reach}_\Delta(P) \subseteq \mathbb{D}_2$, we conclude the following.

Theorem 3. *Let $k \geq 2$, then:*

- *Secrecy is undecidable for (ω, k)-bounded processes.*
- *Secrecy relative to $\mathbb{D}_k$ is undecidable.*

It is natural to ask at this point whether secrecy relative to $\mathbb{D}_k \cap \mathbb{S}_s$ is decidable, since it would rule out the reductions using both encodings $\mathcal{S}[\![\text{-}]\!]$ and $\mathcal{D}[\![\text{-}]\!]$. By means of a third, weaker, encoding, we show that this is not the case.

Theorem 4. *Secrecy relative to $\mathbb{D}_k \cap \mathbb{S}_s$ is undecidable, for any $k > 6$ and $s \geq 3$.*

Proof (*Sketch*)*.* The idea of the proof is to define an encoding of 2MM, detailed in Appendix D, such that the runs of the 2MM can all be matched by some runs of its encoding, and, crucially, these runs only involve processes of depth at most 6 and message sizes at most 3. The encoding is weak in the sense that it also yields spurious runs which do not correspond to real runs of the 2MM. However, the encoding is constructed so that these spurious traces lead invariably to processes exceeding the depth k. The halting instruction of the 2MM is translated to a leak. Therefore we obtain that by considering secrecy relative to $\mathbb{D}_k \cap \mathbb{S}_s$ we are effectively considering reachability of the halting instruction using the non-spurious traces only. $\square$

In the rest of the section we will show that secrecy becomes decidable for (s,k)-bounded processes when s and k are both finite; namely, for protocols whose messages are suitably bounded in size so that no process exceeding some depth bound is reachable, regardless of the intruder's interference.

B. Well-Structured Transition Systems (WSTS)

In this section we review the well-known theory of Well Structured Transition Systems (WSTS) [15, 16], and recall the main results that we are going to use for proving our main decidability result.

Let $(S, \sqsubseteq)$ be a quasi order (qo). An infinite sequence $s_0, s_1, \dots$ of elements of S is called *good* if there are two indexes $i < j$ such that $s_i \sqsubseteq s_j$. The sequence is called *bad* if it is not good. When the qo $(S, \sqsubseteq)$ has no bad sequences it is called a *well quasi order* (wqo).

Definition 9 (Coverability, WSTS)**.** Let $\to$ and $\sqsubseteq$ be binary relations on a set S. For a triple $(S, \to, \sqsubseteq)$, the *coverability problem* asks if, given $s, t \in S$, there exists t' such that $s \to^* t'$ and $t \sqsubseteq t'$. We call t the *coverability query*.

The triple $(S, \to, \sqsubseteq)$ is a *Well Structured Transition System* (WSTS) if

1) $(S, \sqsubseteq)$ is a wqo
2) for every $s, s', t \in S$ such that $s \to t$ and $s \sqsubseteq s'$, there is a $t' \in S$ such that $s' \to t'$ and $t \sqsubseteq t'$.

When condition 2 is met, we say $\sqsubseteq$ is a *simulation*.

It is well-known that for any subset $S' \subseteq S$ of a wqo $(S, \sqsubseteq)$, the set of minimal elements of S', $\min(S')$, is finite. A WSTS $(S, \to, \sqsubseteq)$ is called *effective* if $\sqsubseteq$ is decidable and the function $\mathrm{minpre}\colon S \to \mathscr{P}(S)$ such that

$$\mathrm{minpre}(t) = \min\{s \in S \mid \exists t' \colon s \to t', t \sqsubseteq t'\}.$$

is computable.

Theorem 5 ([15, 16])**.** *The coverability problem is decidable for effective WSTS.*

The algorithm from the proof of Theorem 5 works as follows. Define $\mathrm{pre}(X) := \{s \mid s \to t \in X\}$ and $X{\uparrow} = \{y \in S \mid \exists x \sqsubseteq y\}$ to be the upward-closure of a set X; a set is upward-closed if $X = X{\uparrow}$. Coverability of t from s can be reformulated as the question whether s is a member of $\bigcup_{n\in\mathbb{N}} \mathrm{pre}^n(\{t\}{\uparrow}){\uparrow}$. This set can be computed as the upward-closure of $\bigcup_{n\in\mathbb{N}} \mathrm{minpre}^n(t)$, which saturates after finitely many iterations due to the wqo property.

The section that follows is devoted to defining an ordering on processes, such that it forms a WSTS with the process' semantics, for which secrecy reduces to coverability.

C. Bounded Processes are Well-Structured

We now define two quasi orders on terms. The first, $\sqsubseteq_{\mathsf{ird}}$, compares irreducible standard forms. The second, $\sqsubseteq_{\mathsf{kn}}$, takes message derivability into account. The former serves as a stepping-stone to prove that the latter is a wqo over (s,k)-bounded terms, for $s, k \in \mathbb{N}$.

Definition 10 (Irreducible embedding)**.** Let $P, P' \in \mathbb{P}$. The relation $P \sqsubseteq_{\mathsf{ird}} P'$ holds if $\mathrm{isf}(P) = \nu\vec{x}.(\langle\Gamma\rangle \parallel Q)$, $\mathrm{isf}(P') = \nu\vec{x}.\nu\vec{y}.(\langle\Gamma'\rangle \parallel Q \parallel Q')$ and $\Gamma \subseteq \Gamma'$.

Definition 11 (Knowledge embedding)**.** Let $P, P' \in \mathbb{P}$. The relation $P \sqsubseteq_{\mathsf{kn}} P'$ holds if $\mathrm{sf}(P) = \nu\vec{x}.(\langle\Gamma\rangle \parallel Q)$, $\mathrm{sf}(P') = \nu\vec{x}.\nu\vec{y}.(\langle\Gamma'\rangle \parallel Q \parallel Q')$ and $\Gamma \leq_{\mathsf{kn}} \Gamma'$.

Both orderings are adequate to reduce secrecy to coverability. Let $\sqsubseteq$ be either $\sqsubseteq_{\mathsf{ird}}$ or $\sqsubseteq_{\mathsf{kn}}$. Assuming $a \in \vec{x}$, we have $Q \equiv_{\mathsf{kn}} \nu\vec{x}.(\mathsf{Secret}[\,a\,] \parallel \langle a\rangle \parallel Q')$ if and only if $\nu a.(\mathsf{Secret}[\,a\,] \parallel \langle a\rangle) \sqsubseteq Q$. In other words, embedding $\nu a.(\mathsf{Secret}[\,a\,] \parallel \langle a\rangle)$ characterises leaks.

To be able to apply Theorem 5 we need to show that one of the two embedding forms a WSTS with the semantics of a term. Unfortunately, $\sqsubseteq_{\mathsf{ird}}$ fails to be a simulation.

Fact 1. *The relation $\sqsubseteq_{\mathsf{ird}}$ is not a simulation.*

Proof. Consider, as counterexample, $P = \langle\{\{a\}_b\}_c\rangle \parallel \boldsymbol{\tau}.\langle c\rangle$ and $P' = P \parallel \langle b\rangle$. Clearly, $P \sqsubseteq_{\mathsf{ird}} P'$ and $P \to \langle\{\{a\}_b\}_c\rangle \parallel \langle c\rangle = \langle\Gamma\rangle$ and $P' \to \langle\{\{a\}_b\}_c\rangle \parallel \langle c\rangle \parallel \langle b\rangle = \langle\Gamma'\rangle$. But $\mathrm{ird}(\Gamma) = \{\{a\}_b, c\} \not\subseteq \mathrm{ird}(\Gamma') = \{a, b, c\}$. (Note however that $\Gamma \leq_{\mathsf{kn}} \Gamma'$) $\square$

Knowledge embedding, on the other hand, is easily shown to be a simulation.

Lemma 11. *The relation $\sqsubseteq_{\mathsf{kn}}$ is a simulation.*

Proof. Assume $P_1 \sqsubseteq_{\mathsf{kn}} P_2$ and $P_1 \to Q_1$ where

$$\begin{aligned} P_1 &\equiv_{\mathsf{kn}} \nu\vec{a}_1.(\langle\Gamma\rangle \parallel \mathsf{Q}[\vec{b}\,] \parallel C)\\ P_2 &\equiv_{\mathsf{kn}} \nu\vec{a}_1.\nu\vec{a}_2.(\langle\Gamma'\rangle \parallel \mathsf{Q}[\vec{b}\,] \parallel C \parallel C')\\ \mathsf{Q}[\vec{b}\,] &\triangleq \mathbf{in}(\vec{x} : N).P' + A\\ Q_1 &\equiv_{\mathsf{kn}} \nu\vec{a}_1.\nu\vec{c}.(\langle\Gamma\rangle \parallel \langle\vec{c}\rangle \parallel P'[\vec{M}/\vec{x}\,] \parallel C) \end{aligned}$$

with $\Gamma \leq_{\mathsf{kn}} \Gamma'$, as in the operational semantics rule. Then by $\Gamma \leq_{\mathsf{kn}} \Gamma'$ we obtain that the message $N[\vec{M}/\vec{x}\,]$ can be derived from $\Gamma', \vec{c}$, so $P_2 \to Q_2$ where

$$Q_2 \equiv_{\mathsf{kn}} \nu\vec{a}_1.\nu\vec{a}_2.\nu\vec{c}.(\langle\Gamma'\rangle \parallel \langle\vec{c}\rangle \parallel P'[\vec{M}/\vec{x}\,] \parallel C \parallel C').$$

Since $Q_1 \sqsubseteq_{\mathsf{kn}} Q_2$ this completes the proof. $\square$

Our goal is to show that for all $s, k \in \mathbb{N}$ and for all (s,k)-bounded processes P, $(\mathrm{Reach}_{\Delta}(P) \cap \mathbb{S}_s, \to_{\Delta}, \sqsubseteq_{\mathsf{kn}})$ is an effective WSTS. The next step is proving that $\sqsubseteq_{\mathsf{kn}}$ is a wqo over $\mathbb{D}_k \cap \mathbb{S}_s$. We do so by exploiting the following easy observation.

Lemma 12. *Let $P, Q \in \mathbb{P}$. If $P \sqsubseteq_{\mathsf{ird}} Q$ then $P \sqsubseteq_{\mathsf{kn}} Q$.*

From this fact we obtain that every bad sequence of $(\mathbb{D}_k \cap \mathbb{S}_s, \sqsubseteq_{\mathsf{kn}})$ needs to be a bad sequence of $(\mathbb{D}_k \cap \mathbb{S}_s, \sqsubseteq_{\mathsf{ird}})$. Therefore we focus on proving that the latter has no bad sequence.

Lemma 13. *For any $s, k \in \mathbb{N}$, $(\mathbb{D}_k \cap \mathbb{S}_s, \sqsubseteq_{\mathsf{ird}})$ is a wqo.*

Proof. Fix a finite set of names $Y \subseteq \mathcal{N}$. We first show how terms in $\mathbb{D}_k^Y \cap \mathbb{S}_s^Y$ can be mapped to forests with finitely many labels, so that forest embedding implies irreducible embedding. Then we can invoke Kruskal's theorem [28], which states that forest embedding is a wqo on forests with wqo labels (finite sets are wqo by equality). Formally, $\mathbb{F}_L$ is the set of forests with nodes labelled by elements of a set L, and is defined as the smallest set satisfying $\mathbb{F}_L = \mathcal{M}(L \times \mathbb{F}_L)$ where $\mathcal{M}(X)$ is the set of finite-basis multisets over X, with multiset union $\oplus$. The *forest-embedding* quasi order $\sqsubseteq_{\mathbb{F}}$ is defined so $\varphi_1 \sqsubseteq_{\mathbb{F}} \varphi_2$ holds just if there is an injective function $\iota\colon \operatorname{dom}(\varphi_1) \to \operatorname{dom}(\varphi_2)$ such that for all $t \in \operatorname{dom}(\varphi_1)$,

- $\varphi_1(t) \leq \varphi_2(\iota(t))$, and
- if $t = (\ell_1, \varphi_1')$ and $\iota(\ell_1, \varphi_1') = (\ell_2, \varphi_2')$ then ($\ell_1 = \ell_2$ and $\varphi_1' \sqsubseteq_{\mathbb{F}} \varphi_2'$), or $\{(\ell_1, \varphi_1')\} \sqsubseteq_{\mathbb{F}} \varphi_2'$.

It follows from Kruskal's theorem that $(\mathbb{F}_L, \sqsubseteq_{\mathbb{F}})$ is a wqo if L is finite. We now define a finite set L and a mapping $\mathcal{F}[\![\text{-}]\!]$ from processes to forests, such that:

(A) for all $P \in \mathbb{D}_k^Y \cap \mathbb{S}_s^Y$ there is an irreducible process Q such that $Q \equiv_{\mathsf{kn}} P$ and $\mathcal{F}[\![Q]\!] \in \mathbb{F}_L$;
(B) if $\mathcal{F}[\![Q_1]\!] \sqsubseteq_{\mathbb{F}} \mathcal{F}[\![Q_2]\!]$ then $Q_1 \sqsubseteq_{\mathsf{ird}} Q_2$, for any irreducible processes Q_1, Q_2.

Then the theorem follows from these two properties: assume there is a bad sequence $P_1, P_2, \dots$ in $(\mathbb{D}_k^Y \cap \mathbb{S}_s^Y, \sqsubseteq_{\mathsf{ird}})$, then by (A) there is a corresponding sequence of irreducible processes $Q_1, Q_2, \dots$ with $P_i \equiv_{\mathsf{kn}} Q_i$ and $\mathcal{F}[\![Q_i]\!] \in \mathbb{F}_L$. Since $\sqsubseteq_{\mathsf{ird}}$ is invariant under $\equiv_{\mathsf{kn}}$ by definition, the sequence $Q_1, Q_2, \dots$ is also bad. Hence, by (B) the sequence $\mathcal{F}[\![Q_1]\!], \mathcal{F}[\![Q_2]\!], \dots$ is also bad, which is in contradiction with Kruskal's theorem.

To conclude the proof we need to define a suitable map $\mathcal{F}[\![\text{-}]\!]$:

$$\mathcal{F}[\![P]\!] := \begin{cases} \emptyset & \text{if } P = \mathbf{0} \\ \{(P, \emptyset)\} & \text{if } P \text{ is sequential} \\ \{(a, \mathcal{F}[\![Q]\!])\} & \text{if } P = \nu a.Q \\ \mathcal{F}[\![Q_1]\!] \oplus \mathcal{F}[\![Q_1]\!] & \text{if } P = Q_1 \parallel Q_2 \end{cases}$$

It is not difficult to see that (B) holds for this definition: $\mathcal{F}[\![Q_1]\!] \sqsubseteq_{\mathbb{F}} \mathcal{F}[\![Q_2]\!]$ implies that every node in $\mathcal{F}[\![Q_1]\!]$ is mapped to a node in $\mathcal{F}[\![Q_2]\!]$ with the same label; therefore, by renaming every name to a fresh name to avoid clashes, and applying scope extrusion, we obtain that $Q_1 \equiv \nu\vec{a}.(\langle\Gamma\rangle \parallel Q)$ and $Q_2 \equiv \nu\vec{a}.\nu\vec{b}.(\langle\Gamma\rangle \parallel Q \parallel R)$ (here $\vec{a}$ and Q are the renamed labels that were matching in the two forests). Since Q_1 and Q_2 are assumed to be irreducible, this proves $Q_1 \sqsubseteq_{\mathsf{ird}} Q_2$.

Now to prove (A) we have to fix a finite set of labels L: let $X_k = \{x_1, \dots, x_n\} \subseteq \mathcal{N}$ such that $X_k \cap Y = \emptyset$ and define

$$L := X \cup O \cup S \quad X := Y \cup X_k \quad O := \{\langle M\rangle \mid M \in \mathbb{M}_s^X\}$$
$$S := \{\mathsf{Q}[\,M_1, \dots, M_n\,] \mid \mathsf{Q} \in \mathcal{Q}, \operatorname{ar}(\mathsf{Q}) = n, M_i \in \mathbb{M}_s^X\}$$

The set L is clearly finite. We can now use Lemma 9 to get an irreducible $Q \equiv_{\mathsf{kn}} P$ with $\operatorname{nest}_\nu(Q) = \operatorname{depth}(P) \leq k$. By Lemma 10 we can assume $\operatorname{bn}(Q) \subseteq X_k$, therefore $\mathcal{F}[\![Q]\!] \in \mathbb{F}_L$ as required. $\square$

Lemma 14. *For any $s, k \in \mathbb{N}$, $(\mathbb{D}_k \cap \mathbb{S}_s, \sqsubseteq_{\mathsf{kn}})$ is a wqo.*

Proof. Straightforward consequence of Lemma 13 and Lemma 12: any bad sequence of $\sqsubseteq_{\mathsf{kn}}$ is also a bad sequence of $\sqsubseteq_{\mathsf{ird}}$. $\square$

Theorem 6. *Let $s, k \in \mathbb{N}$ and P be a (s, k)-bounded process, under a set Δ of definitions. Then $(\operatorname{Reach}_\Delta(P) \cap \mathbb{S}_s, \to_\Delta, \sqsubseteq_{\mathsf{kn}})$ is an effective WSTS.*

Proof. Let $S := \operatorname{Reach}_\Delta(P) \cap \mathbb{S}_s$. From the boundedness assumption on P we have $S \subseteq \mathbb{D}_k \cap \mathbb{S}_s$, so by Lemma 14, $(S, \sqsubseteq_{\mathsf{kn}})$ is a wqo. To show that $\sqsubseteq_{\mathsf{kn}}$ is a simulation on S, we can replicate the proof of Lemma 11: for every $P_1, P_2, Q_1 \in S$ with $P_1 \sqsubseteq_{\mathsf{kn}} P_2$ and $P_1 \to Q_1$, we can find a Q_2 such that $P_2 \to Q_2$; it is easy to check that Q_2 is in S: Q_2 is clearly in $\operatorname{Reach}_\Delta(P)$, and since P_2 and Q_1 are in $\mathbb{S}_s$, so is Q_2.

We complete the proof by showing effectiveness. The relation $\sqsubseteq_{\mathsf{kn}}$ is clearly decidable. The following construction of minpre is proof of its computability. We define $\operatorname{minpre}(R) := \min_{\sqsubseteq_{\mathsf{kn}}}(\mathcal{B})$ where the finite set of processes $\mathcal{B}$ is computed as follows. Let $Y = \operatorname{bn}(R)$ and let $Z_{\mathsf{Q}} \subseteq \mathcal{N}$ be a set of fresh names with $|Z_{\mathsf{Q}}| = \operatorname{ar}(\mathsf{Q}) \cdot 2^{s-1}$ for any $\mathsf{Q} \in \mathcal{Q}$; similarly let $Z_{\vec{x}}$ be a set of fresh names with $|Z_{\vec{x}}| = |\vec{x}| \cdot 2^{s-1}$ for any tuple $\vec{x}$. We enumerate:

- all $\mathsf{Q} \in \mathcal{Q}$, and let $W = X \cup Y \cup Z_{\mathsf{Q}}$,
- all tuples $\vec{M}' = M_1', \dots, M_{\operatorname{ar}(\mathsf{Q})}' \in \mathbb{M}_s^W$,
- all $\mathbf{in}(\vec{x} : N).P'$ such that $\mathsf{Q}[\,\vec{M}'\,] \triangleq \mathbf{in}(\vec{x} : N).P' + A$
- all tuples $\vec{M} = M_1, \dots, M_{|\vec{x}|} \in \mathbb{M}_s^{W \cup Z_{\vec{x}}}$ and let $\vec{c} = \operatorname{names}(\vec{M}) \cap Z_{\vec{x}}$, and
- all $\Gamma \subseteq \mathbb{M}_s^W$.

For each, we compute the $\sqsubseteq_{\mathsf{kn}}$-smallest process C such that:

$$\begin{aligned} Q &= \nu Y.\nu Z_{\mathsf{Q}}.(\langle\Gamma\rangle \parallel \mathsf{Q}[\,\vec{M}'\,] \parallel C) \\ &\to \nu Y.\nu Z_{\mathsf{Q}}.\nu\vec{c}.(\langle\Gamma\rangle \parallel \langle\vec{c}\rangle \parallel P'[\,\vec{M}'/\vec{x}\,] \parallel C) = R' \end{aligned}$$

with $R \sqsubseteq_{\mathsf{kn}} R' \in \mathbb{D}_k^X \cap \mathbb{S}_s^X$. The resulting Q is added to $\mathcal{B}$ when $\operatorname{depth}(Q) \leq k$. The key observation is that, while the predecessors of some process greater than some Q may contain arbitrarily many new names, to compute the minimal predecessors it suffices to make up a finite number of fresh names. To see this, first observe that a message of size at most s can mention at most 2^{s-1} distinct names. Since we are considering only processes in S, and therefore in $\mathbb{S}_s^X$ for $X = \operatorname{fn}(P)$, the size of every message involved in a step is bounded by s. $\square$

As a corollary, since secrecy reduces to coverability, and coverability is decidable for effective WSTS, we prove that secrecy is decidable for bounded processes.

Theorem 7 (Decidability)**.** *Let P be a (s,k)-bounded process, under a set Δ of definitions, for some $s,k \in \mathbb{N}$. Then secrecy for P relative to $\mathbb{S}_s$ is decidable.*

Proof. If P is (s,k)-bounded then, by Theorems 5 and 6, secrecy for P relative to $\mathbb{S}_s$ is reducible to coverability from P of the query $\nu\vec{x}.(\mathsf{Secret}[\,M\,] \parallel \langle M \rangle)$ with $\vec{x} = \mathrm{fn}(M) \setminus \mathrm{fn}(P)$, in the effective WSTS $(\mathrm{Reach}_\Delta(P) \cap \mathbb{S}_s, \to, \sqsubseteq_{\mathsf{kn}})$. □

Note that this does not contradict Theorem 4: even for finite s,k, $(\mathbb{D}_k \cap \mathbb{S}_s, \to, \sqsubseteq_{\mathsf{kn}})$ is not a WSTS. This is because $\sqsubseteq_{\mathsf{kn}}$ is not a simulation over $\mathbb{D}_k \cap \mathbb{S}_s$: in the proof of Lemma 11 the depth of Q_2 is not necessarily less than k.

Example 6 (Encryption Oracle). Consider the following definition:

$$\mathsf{O}[\,k\,] := \mathbf{in}(x : x).(\langle \{x\}_k \rangle \parallel \mathsf{O}[\,k\,])$$

We call $\mathsf{O}[\,k\,]$ an *encryption oracle* since it encrypts every message it receives with a private key k. There is a common pattern in protocols: an interaction begins by a principal sending an unencrypted nonce x to the server, and expecting to receive back from the server some data encrypted together with the nonce. It is problematic to apply Theorem 7 to such a pattern: if messages of size 3 are allowed, the intruder would be able to send to the oracle messages of the form $\langle (c_0, c_1) \rangle$, $\langle (c_1, c_2) \rangle$ etc. where $c_1, c_2, \dots$ are intruder generated nonces. Consequently, and since k is never leaked, the reachable irreducible processes are not depth-bounded. Typically, this pattern is employed to encrypt nonces couple with certain data known to the oracle, which means that the pattern variable x is expected to match simple nonces, and not structured messages. The above complication is thus caused by a type-confusion interference by the intruder.

The process $\mathsf{O}[\,k\,]$ is $(2,1)$-bounded however: the oracle can only output messages of the form $\{c_i\}_k$ (which have size 2). The irreducible reachable processes have the form[1] $\left(\nu c.\langle \{c\}_k \rangle\right)^n \parallel \mathsf{O}[\,k\,]$ which have depth 1.

The example above illustrates a limitation of our decidability result. We present two ways of fixing this issue: either we require the designer of the protocol to always send *encrypted* nonces to oracles, or we restrict the size of messages so that the problematic type-confusion interference is avoided. Consider the modified oracle definition:

$$\mathsf{EO}[\,k_o, k\,] := \mathbf{in}(x : \{x\}_{k_o}).(\langle \{x\}_k \rangle \parallel \mathsf{EO}[\,k_o, k\,])$$

Now a principal needs to know the k_o key to be able to interact with the encryption oracle and the intruder cannot inject counterfeit messages in x, unless the key k_o is leaked. Since most commonly a principal and an encryption oracle share a private key with which they encrypt their further communications, the same key can be used to communicate the nonce. To apply this fix, the designer must also fence-off indirect influences of the intruder on the nonces; how this is done depends on the specific structure of each protocol. This precaution can be used to obtain a slight variant of the NSSK protocol of Example 1, where the only changes needed are in the first step of the protocol

$$\mathsf{S}_1[x, y, k_{xs}, k_{ys}] := \mathbf{in}(n_a : \{(x, y, n_x)\}_{k_{xs}}).\nu k.(\dots)$$
$$\mathsf{A}_1[x, y, k_{xs}] := \boldsymbol{\tau}.\nu n.(\langle \{(x, y, n)\}_{k_{xs}} \rangle \parallel \dots)$$

while the omitted portion of the code remains identical to Fig. 2. Thus modified, the protocol can be shown $(s,3)$-bounded for all $s \in \mathbb{N}$.

The second way to make the protocol analysable using our decidability result, is by assuming the implementation has some means to detect and fence-off type confusion attacks in which the intruder can send messages of unexpected sizes to principals. We model this assumption by means of what we call a *sizing* function $\varsigma \colon \mathcal{N} \to \mathbb{N}$ that maps names to sizes. We then require every substitution to respect ς, that is only names assigned the same size can be replaced for one another. Only names n with $\varsigma(n) = 1$ can be used in restrictions. A sizing function is naturally extended to messages by setting $\varsigma(\{M\}_N) := \varsigma((M,N)) := 1 + \max(\varsigma(M), \varsigma(N))$. We define $\mathrm{Traces}^{\varsigma}_{\Delta}(P)$ to be the set of traces respecting the sizing ς, i.e. where the each step involves only ς-respecting substitutions. We say a process is (ς,k)-bounded if $\mathrm{Traces}^{\varsigma}_{\Delta}(P) \subseteq \mathbb{D}^*_k$. Naturally, a (ς,k)-bounded process P is also (s,k)-bounded where s is the maximum value of $\varsigma(M)$ where M occurs as a subterm of P or of the body of some definition in Δ.

In Example 1 the intended type of n_x is a nonce created by a principal. So we can impose the sizing with $\varsigma(n_x) = 1$ and arbitrary size values to all the other pattern variables (all the restricted names will have sizing 1). We obtain that the program is $(\varsigma, 2)$-bounded which enables the use of Theorem 7 for verification. In the next section, we elaborate on this claim.

V. Proving Boundedness

We present in this section a general method, amenable to automation, for proving that a protocol is (s,k)-bounded for some given $s,k \in \mathbb{N}$. Although we only demonstrate it on an example, it builds on the general theory of *ideal completions*, which we discuss in Section VII-A.

We want to prove that the NSSK protocol is $(\varsigma,3)$-bounded, with $\varsigma(n_x) = 1$. To improve readability, we only consider the instance with two honest principals a, always the initiator, and b, always the responder. The general case is a straightforward extension of the proof.

The definitions are the ones of Fig. 2 but the starting point is

$$NS_0 = \mathsf{S}_1[\,\vec{s}\,] \parallel \mathsf{A}_1[\,\vec{a}\,] \parallel \mathsf{B}_1[\,\vec{b}\,] \parallel \langle a, b \rangle$$

with $\vec{s} = a, b, k_{as}, k_{bs}$, $\vec{a} = a, b, k_{as}$, and $\vec{b} = a, b, k_{bs}$.

The proof method works by checking, given $s,k \in \mathbb{N}$, the existence of a set $\mathcal{I}$ of processes such that:

(C1) $P_0 \in \mathcal{I}$;
(C2) $\mathcal{I}$ is an inductive invariant (up to size bound s): if $P \in \mathcal{I}$ and $P \to Q \in \mathbb{S}_s$ then $Q \in \mathcal{I}$;

[1]Here we write P^n for the parallel composition of n copies of P.

(C3) $\mathcal{I} \subseteq \mathbb{S}_s \cap \mathbb{D}_k$.

An invariant satisfying (C1) and (C2) necessarily satisfies $\mathrm{Reach}(P_0) \cap \mathbb{S}_s \subseteq \mathcal{I}$. Condition (C3) then ensures that P_0 is (s,k)-bounded. The scheme can be easily adapted to support a sizing function.

What makes this use of inductive invariants suitable for automation is that all the subsets of $\mathbb{S}_s \cap \mathbb{D}_k$ that may be needed to carry out such a proof, can be denoted using a simple symbolic representation.

We call *limits* the terms L formed according to the grammar:

$$L ::= \mathbf{0} \mid \langle M \rangle \mid \mathsf{Q}[\vec{x}] \mid \nu x.L \mid L \parallel L \mid L^{\omega}$$

That is, we add to the grammar of processes a case L^{ω}, called iteration. The crucial difference with processes is that these limits represent sets of processes. The denotation of L is the set $[\![L]\!]$ defined by:

$$\begin{aligned} [\![P]\!] &:= \{Q \mid Q \sqsubseteq_{\mathsf{kn}} P\} \quad \text{if } P \text{ is sequential} \\ [\![\nu x.L]\!] &:= \{Q \mid Q \equiv_{\mathsf{kn}} \nu x.P, P \in [\![L]\!]\} \\ [\![L_1 \parallel L_2]\!] &:= \{Q \mid Q \equiv_{\mathsf{kn}} (P_1 \parallel P_2), P_i \in [\![L_i]\!]\} \\ [\![L^{\omega}]\!] &:= \{Q \mid Q \equiv_{\mathsf{kn}} (P_1 \parallel \cdots \parallel P_n), P_i \in [\![L]\!], n \in \mathbb{N}\} \end{aligned}$$

We call the processes in $[\![L]\!]$ *instances* of L. Note that:

- every denotation contains $\mathbf{0}$ and more generally is *downward-closed* with respect to $\sqsubseteq_{\mathsf{kn}}$, i.e. if $P \in [\![L]\!]$ and $Q \sqsubseteq_{\mathsf{kn}} P$, then $Q \in [\![L]\!]$;
- as a consequence of downward closure, every denotation is also closed under $\equiv_{\mathsf{kn}}$;
- iterating a process $\langle M \rangle$ is superfluous: by the persistence law $[\![\langle M \rangle^{\omega}]\!] = [\![\langle M \rangle]\!]$.

Define $\mathrm{nest}_{\nu}(L)$ to be defined as nest_{ν} on processes with the addition of the case $\mathrm{nest}_{\nu}(L^{\omega}) := \mathrm{nest}_{\nu}(L)$. It is easy to check that for each $P \in [\![L]\!]$, $\mathrm{depth}(P) \leq \mathrm{nest}_{\nu}(L)$.

We now write a limit L_1, the denotation of which satisfies (C1) to (C3), for $P_0 = NS_0$, the sizing ς and $k = \mathrm{nest}_{\nu}(L_1) = 3$, allowing us to conclude that NS_0 is $(\varsigma, 3)$-bounded.

$$\begin{aligned} L_1 &= \mathsf{S}_1[\vec{s}]^{\omega} \parallel \mathsf{A}_1[\vec{a}]^{\omega} \parallel \mathsf{B}_1[\vec{b}]^{\omega} \parallel \langle a,b \rangle \parallel G \parallel L_2^{\omega} \\ G &= (\nu k.\langle M_{a,k} \rangle)^{\omega} \parallel (\nu k.\langle M_{b,k} \rangle)^{\omega} \\ M_{x,k} &= \{x, k, b, \{k,a\}_{k_{bs}}\}_{k_{as}} \\ L_2 &= \nu n.\big(\langle n \rangle \parallel \mathsf{A}_2[\vec{a}, n] \parallel L_3^{\omega}\big) \\ L_3 &= \nu k.\big(\langle M_{n,k}, \{k,a\}_{k_{bs}} \rangle \parallel \mathsf{A}_3[\vec{a}, k] \parallel L_4^{\omega}\big) \\ L_4 &= \nu s.\big(\langle \{s\}_k, \{s,s\}_k \rangle \parallel \mathsf{B}_2[\vec{b}, k, s]\big) \end{aligned}$$

Clearly, $P_0 \in [\![L_1]\!]$. It is not difficult, if tedious, to check that $[\![L_1]\!]$ is an inductive invariant up to ς.

The attentive reader may notice an interesting consequence of invariance of L_1: since no irreducible instance of L_1 embeds a message $\langle s \rangle$, the invariant proves s is not leaked (recall that this is the scenario where replay attacks are not modelled). This is no coincidence: the above scheme is an instance of forward coverability algorithms for WSTS with a post-effective ideal completion [29]. Presenting the full details is out of the scope of this paper; for a brief discussion see Section VII.

$$\begin{aligned} \mathsf{A}[\,k_{ab}\,] &:= \boldsymbol{\tau}.\nu n_a.(\langle n_a \rangle \parallel \mathsf{A}_1[k_{ab}, n_a] \parallel \mathsf{A}[\,k_{ab}\,]) \\ \mathsf{A}_1[k_{ab}, n_a] &:= \mathbf{in}(k, n_b : (\{1,k,n_a\}_{k_{ab}}, n_b)).\langle \{2,k,n_b\}_{k_{ab}} \rangle \\ \mathsf{B}[\,k_{ab}\,] &:= \mathbf{in}(x : x).\nu k, n_b.(\langle (\{1,k,x\}_{k_{ab}}, n_b) \rangle \parallel \mathsf{B}[\,k_{ab}\,]) \\ \mathsf{System} &:= \nu k_{ab}.(\mathsf{A}[\,k_{ab}\,] \parallel \mathsf{B}[\,k_{ab}\,]) \end{aligned}$$

Figure 3. An encoding of the protocol in Example 7

Since boundedness is a property of the reachable configurations, it should not come as a surprise that proving it is a hard task. Other conditions from the literature of a more syntactic nature may be easier to check but not as expressive. Isolating a more statically checkable fragment is a topic for future work.

VI. Examples of Security Protocols

We shall now discuss several examples of real-world protocols to illustrate the key differences between our approach and those of Fröschle [11] and of Chrétien et al. [12] in closely related work. One main difference between our approach and both Fröschle [11] and of Chrétien et al. [12] work is that we do not impose a restriction that the encryption key has to be atomic. The following examples lend further support to our conclusion that the class of depth-bounded protocols are incomparable to the class of protocols defined in [11, 12], both of which rely on a notion of acyclicity in the protocols, i.e., the notion of *well-founded* protocols in [11] and a notion of acyclicity in *dependency graph* between encrypted messages in [12].

Example 7. Consider the following key exchange protocol:

$$\begin{array}{lll} (1) & A \to B: & N_A \\ (2) & B \to A: & \{1, K, N_A\}_{K_{AB}}, N_B \\ (3) & A \to B: & \{2, K, N_B\}_{K_{AB}} \end{array}$$

where N_A and N_B are nonces and K is a fresh session key and K_{AB} is a long term key shared by A and B. For simplicity, we consider only two honest principals A and B, and the intruder is not a principal of the protocol. This protocol is not depth-bounded, regardless of the size restriction on messages. For every nonce N_1 generated by A, the intruder can create an encryption chain $\{1, K_1, N_1\}_{K_{AB}}$ and $\{2, K_1, X\}_{K_{AB}}$ where X is controlled by the intruder, as follows:

$$\begin{array}{lll} (1) & A \to B & : \; N_1 \\ (2) & B \to I(A) : & \{1, K_1, N_1\}_{K_{AB}}, N_2 \\ (2) & I(B) \to A : & \{1, K_1, N_1\}_{K_{AB}}, X \\ (3) & A \to I(B) : & \{2, K_1, X\}_{K_{AB}} \end{array}$$

This sequence can be repeated for each session initiated by A to form an unbounded encryption chain: $\{1, K_2, N_2\}_{K_{AB}}$, $\{2, K_2, N_1\}_{K_{AB}}$, $\{1, K_3, N_3\}_{K_{AB}}$, $\{2, K_3, N_2\}_{K_{AB}}$, etc.

Figure 3 shows an encoding of the above protocol, where 1 and 2 are distinct free names (omitted from the argument lists). The natural size bound of the protocol is 4: since $\mathrm{size}(\{(1,(k,x))\}_{k_{ab}}) = 4$, the principals can exchange all the messages needed, but the intruder cannot try to replace any structured message for x without exceeding the size bound.

We illustrate here some transitions that lead to unbounded depth, using the informal idea above to form encryption chains. Observe that for any name x, we have:

$$\begin{aligned}&\mathsf{A}[\,k_{ab}\,] \parallel \mathsf{B}[\,k_{ab}\,] \parallel \langle x\rangle \to^* \\ &\quad \nu k_1.(\langle\{2,k_1,x\}_{k_{ab}}\rangle \parallel \nu n_1.(\langle\{1,k_1,n_1\}_{k_{ab}}\rangle \parallel \\ &\qquad\qquad \mathsf{A}[\,k_{ab}\,] \parallel \mathsf{B}[\,k_{ab}\,] \parallel \langle n_1\rangle) \parallel \nu n_b.\langle n_b\rangle) \parallel \langle x\rangle\end{aligned}$$

We can apply the same transitions to the subprocess $\mathsf{A}[\,k_{ab}\,] \parallel \mathsf{B}[\,k_{ab}\,] \parallel \langle n_1\rangle$ to create:

$$\begin{aligned}&\mathsf{A}[\,k_{ab}\,] \parallel \mathsf{B}[\,k_{ab}\,] \parallel \langle x\rangle \to^* \\ &\quad \nu k_1.(\langle\{2,k_1,x\}_{k_{ab}}\rangle \parallel \nu n_1.(\langle\{1,k_1,n_1\}_{k_{ab}}\rangle \\ &\qquad\qquad \nu k_2.(\langle\{2,k_2,n_1\}_{k_{ab}}\rangle \parallel \nu n_2.(\langle\{1,k_2,n_2\}_{k_{ab}}\rangle \parallel \\ &\qquad\qquad\qquad \mathsf{A}[\,k_{ab}\,] \parallel \mathsf{B}[\,k_{ab}\,] \parallel \langle n_2\rangle)))) \parallel \ldots\end{aligned}$$

This expansion can be repeated to create a process with arbitrary depth, while never exceeding the size bound 4. Note that none of the encrypted subterm can be decrypted by the attacker, since the key k_{ab} is not known to the attacker, so the terms are irreducible. Since we can reach terms embedding encryption chains of unbounded length, the reachable terms have unbounded depth. It is easy to show that

$$\begin{aligned}\mathsf{System} \to^* \nu k_{ab},k,x,y.(&\langle\{1,k,x\}_{k_{ab}}\rangle \parallel \langle y\rangle \parallel \\ &\langle\{2,k,y\}_{k_{ab}}\rangle \parallel \mathsf{A}[\,k_{ab}\,] \parallel \mathsf{B}[\,k_{ab}\,] \parallel \langle x\rangle).\end{aligned}$$

Since $\mathsf{A}[\,k_{ab}\,] \parallel \mathsf{B}[\,k_{ab}\,] \parallel \langle x\rangle$ is not $(4,n)$-bounded, for any $n \in \mathbb{N}$, it follows that the protocol (System) is not $(4,n)$-bounded.

It can be shown that the above protocol is well-founded (see Appendix F), in the sense of Fröschle [11], hence is in the class of decidable protocols in [11]. This is essentially due to the tagging of encrypted messages (via the constants 1 and 2), so 'honest' encrypted messages created by a principal cannot be propagated indefinitely, i.e., the encrypted message in the output of a step in the protocol cannot be fed back to the input at an earlier step due to the different tags used at every step.

Example 8. Recall the Otway-Rees protocol [30]:

$$\begin{array}{lll}(1) & A \to B: & M,A,B,\{N_A,M,A,B\}_{K_{AS}} \\ (2) & B \to S: & M,A,B,\{N_A,M,A,B\}_{K_{AS}}, \\ & & \{N_B,M,A,B\}_{K_{BS}} \\ (3) & S \to B: & M,\{N_A,K_{AB}\}_{K_{AS}},\{N_B,K_{AB}\}_{K_{BS}} \\ (4) & B \to A: & M,\{N_A,K_{AB}\}_{K_{AS}}\end{array}$$

This protocol is not well-founded [11]: the offending part is a simple blind copy (no re-encryption involved) of message $\{N_A,M,A,B\}_{K_{AS}}$. Fröschle shows [11] that such simple blind copies can be removed from the protocol encoding in her framework, without affecting the termination analysis. Chrétien et al. [12] introduce tagging in the encrypted messages to the above protocol to ensure termination in their analysis, but they do not restrict the size of messages exchanged during the runs of the protocol. They noted that, without tagging, the dependency graph resulting from their model of the protocol is cyclic.

Figure 4 shows an encoding of the different roles of the protocol in our calculus. The protocol itself, assuming two fixed principals a and b, can be encoded as the process

$$OR = \mathsf{S}_1[a,b,k_{as},k_{bs}] \parallel \mathsf{A}_1[a,b,k_{as}] \parallel \mathsf{B}_1[a,b,k_{bs}]$$

We impose the size restrictions on the variables of the process OR as follows: x is of size 5, y is of size 3, and the rest of the variables are all of size 1. Note that although our analysis works on the untagged version of the protocol, we still need to restrict the message size, so in this case there is no direct comparison with the work done in [12]. Using the proof techniques in Section V, we can show that this protocol is depth-bounded. It may be worth highlighting the difference between this protocol and that in Example 7. Both have encryption oracles where nonces controlled by the attacker are encrypted by honest participants. In the case of Otway-Rees, the trick used in Example 7 to form the encryption chain does not work, since the value of nonces N_A, N_B and the key K_{AB} are not leaked. The only nonce that the attacker knows and can influence is M. It is easy to see that the longest encryption chain the attacker can form from its interactions with the protocol is of length at most 4, e.g. $\{N_A,M,\ldots\}_{K_{AS}}$, $\{N_B,M,\ldots\}_{K_{BS}}$, $\{N_B,K_{AB}\}_{K_{BS}}$, $\{N_A,K_{AB}\}_{K_{AS}}$.

Example 9. Recall the Yahalom protocol [31]:

$$\begin{array}{lll}(1) & A \to B: & A,N_A \\ (2) & B \to S: & B,\{A,N_A,N_B\}_{K_{BS}} \\ (3) & S \to A: & \{B,K_{AB},N_A,N_B\}_{K_{AS}},\{A,K_{AB}\}_{K_{BS}} \\ (4) & A \to B: & \{A,K_{AB}\}_{K_{BS}},\{N_B\}_{K_{AB}}\end{array}$$

If we assume the attacker is among the principals, then it is not depth-bounded. For example, if A is the attacker, it controls N_A, and by the end of the session, will know N_B, so it can use N_B as the nonce at the start of the next session with B, and will form an encryption chain through the message at step (2). In the setting of Chrétien et al. [12], the corresponding dependency graph is cyclic if sessions with dishonest agents are admissible, for the same reason as above. The cycles do not arise when restricted to honest sessions. Figure 5 shows an encoding of the roles in the Yahalom protocol in our calculus. The full protocol, when we fix the principals to a and b, can be encoded as

$$Y = \mathsf{S}_1[a,b,k_{as},k_{bs}] \parallel \mathsf{A}_1[a,b,k_{as}] \parallel \mathsf{B}_1[a,b,k_{bs}]$$

To prove depth-boundedness, we restrict the size of variables in Y such that x is of size 3 and all other variables are of size 1. Using the proof techniques in Section V, we can prove this protocol depth-bounded.

VII. Discussion and Future Work

A. Forward analysis

Theorem 7 proved decidability of coverability for (s,k)-bounded processes by instantiating the *backwards search*, detailed in Section IV-B. The backwards search has two main drawbacks: 1) it needs to be given a suitable depth-bound k, 2) it may be inefficient due to the fact that it starts from

$$\begin{aligned}
\mathsf{S}_1[a,b,k_{as},k_{bs}] &:= \mathbf{in}(m,n_a,n_b : m,a,b,\{n_a,m,a,b\}_{k_{as}},\{n_b,m,a,b\}_{k_{bs}}).\\
&\qquad (\nu k_{ab}.\langle m,\{n_a,k_{ab}\}_{k_{as}},\{n_b,k_{ab}\}_{k_{bs}}\rangle \parallel \mathsf{S}_1[a,b,k_{as},k_{bs}])\\
\mathsf{A}_1[a,b,k_{as}] &:= \boldsymbol{\tau}.\nu m,n_a.(\langle m,a,b,\{n_a,m,a,b\}_{k_{as}}\rangle \parallel \mathsf{A}_2[m,n_a,k_{as}] \parallel \mathsf{A}_1[a,b,k_{as}])\\
\mathsf{A}_2[m,n_a,k_{as}] &:= \mathbf{in}(k_{ab} : m,\{n_a,k_{ab}\}_{k_{as}}).\mathbf{0}\\
\mathsf{B}_1[a,b,k_{bs}] &:= \mathbf{in}(m,x : m,a,b,x).\nu n_b.(\langle m,a,b,x,\{n_b,m,a,b\}_{k_{bs}}\rangle \parallel \mathsf{B}_2[n_b,k_{bs}] \parallel \mathsf{B}_1[a,b,k_{bs}])\\
\mathsf{B}_2[n_b,k_{bs}] &:= \mathbf{in}(m,y,k_{ab} : m,y,\{n_b,k_{ab}\}_{k_{bs}}).\langle m,y\rangle
\end{aligned}$$

Figure 4. An encoding of Otway-Rees protocol

$$\begin{aligned}
\mathsf{S}_1[a,b,k_{as},k_{bs}] &:= \mathbf{in}(n_a,n_b : b,\{a,n_a,n_b\}_{k_{bs}}).\nu k_{ab}.(\langle\{b,k_{ab},n_a,n_b\}_{k_{as}},\{a,k_{ab}\}_{k_{bs}}\rangle \parallel \mathsf{S}_1[a,b,k_a,k_b])\\
\mathsf{A}_1[a,b,k_{as}] &:= \boldsymbol{\tau}.\nu n_a.(\langle a,n_a\rangle \parallel \mathsf{A}_2[n_a,a,b,k_{as}] \parallel \mathsf{A}_1[a,b,k_{as}])\\
\mathsf{A}_2[n_a,a,b,k_{as}] &:= \mathbf{in}(k_{ab},n_b,x : \{b,k_{ab},n_a,n_b\}_{k_{as}},x).\langle x,\{n_b\}_{k_{ab}}\rangle\\
\mathsf{B}_1[a,b,k_{bs}] &:= \mathbf{in}(n_a : a,n_a).\nu n_b.(\langle b,\{a,n_a,n_b\}_{k_{bs}}\rangle \parallel \mathsf{B}_2[n_b,a,b,k_{bs}] \parallel \mathsf{B}_1[a,b,k_{bs}])\\
\mathsf{B}_2[n_b,a,b,k_{bs}] &:= \mathbf{in}(k_{ab} : \{a,k_{ab}\}_{k_{bs}}).\langle\{n_b\}_{k_{ab}}\rangle
\end{aligned}$$

Figure 5. An encoding of Yahalom protocol

the coverability query: a backward step may involve many terms that are in no way reachable from the initial state. To remedy both shortcomings, one could instantiate the so-called *forward search* algorithm for WSTS with *post-effective ideal completions* [29]. As a nice byproduct, we would be able, by slight modification of the algorithm to get a decision procedure that given P, and $s, k \in \mathbb{N}$, determines if P is (s,k)-bounded. The instantiation of the ideal completions framework for cryptographic protocol is a topic of ongoing research.

B. Approximate analyses

Protocols that are bounded in depth are very expressive, a fact that we pay in terms of complexity: the only known lower-bound for coverability is non-primitive recursive for depth-bounded processes. Only extensive experimentation will reveal if typical protocols exercise this very high computational price. It is therefore natural to ask if there are further restrictions, or approximate coverability analyses, that can lower the complexity. There is indeed scope for developing both, by adapting recent work for the π-calculus.

A type-theoretic approach to reason about depth at low complexities is introduced in [32]; while adapting the types to embed message-derivability information is not obvious, we are hopeful that a similar type system can be constructed for our calculus, unlocking the cheap estimation of secrecy invariants expressed as types. On the static analysis side, acceleration schemes such as the ones presented in [33] can provide sound but approximate methods to find invariants proving secrecy.

C. Conclusions

We have presented a new expressive class of security protocols with an unbounded number of sessions and unrestricted fresh data for which secrecy is decidable. Based on the notion of depth-boundedness, this class of protocols includes many real-world symmetric key protocols. Our investigation on the relation between depth bounds and other known protocol restrictions reveals that depth boundedness cannot be seen as a generalisation of known results. At the same time, the fragment is not subsumed by any other restriction in the literature, making it a genuinely new perspective on the structure of protocols.

Acknowledgement

The third author is supported by Singapore Ministry of Education Tier 2 grant MOE2014-T2-02-076 and NTU Startup grant M4081190.020.

References

[1] D. Dolev and A. C. Yao, "On the security of public key protocols," *IEEE Trans. Information Theory*, vol. 29, no. 2, pp. 198–207, 1983.

[2] N. A. Durgin, P. D. Lincoln, J. C. Mitchell, and A. Scedrov, "Undecidability of bounded security protocols," in *Workshop on Formal Methods and Security Protocols*, 1999.

[3] R. M. Amadio, D. Lugiez, and V. Vanackère, "On the symbolic reduction of processes with cryptographic functions," *Theor. Comput. Sci.*, vol. 290, no. 1, pp. 695–740, 2003.

[4] N. Heintze and J. D. Tygar, "A model for secure protocols and their compositions," *IEEE Trans. Software Eng.*, vol. 22, no. 1, pp. 16–30, 1996.

[5] M. Rusinowitch and M. Turuani, "Protocol insecurity with finite number of sessions is NP-complete," in *CSFW'01*, 2001, pp. 174–187.

[6] H. Comon-Lundh, V. Cortier, and E. Zalinescu, "Deciding security properties for cryptographic protocols. application to key cycles," *ACM Trans. Comput. Log.*, vol. 11, no. 2, pp. 9:1–9:42, 2010.

[7] G. Lowe, "Towards a completeness result for model checking of security protocols," *Journal of Computer Security*, vol. 7, no. 1, pp. 89–146, 1999.

[8] R. Ramanujam and S. P. Suresh, "Decidability of context-explicit security protocols," *Journal of Computer Security*, vol. 13, no. 1, pp. 135–165, 2005.

[9] ——, "A decidable subclass of unbounded security protocols," in *WITS'03*, 2003, pp. 11–20.

[10] ——, "Tagging makes secrecy decidable with unbounded nonces as well," in *FSTTCS'03*, 2003, pp. 363–374.

[11] S. B. Fröschle, "Leakiness is decidable for well-founded protocols," in *POST'15*, 2015, pp. 176–195.

[12] R. Chrétien, V. Cortier, and S. Delaune, "Decidability of trace equivalence for protocols with nonces," in *CSF'15*. IEEE Computer Society, 2015, pp. 170–184.
[13] R. Milner, J. Parrow, and D. Walker, "A calculus of mobile processes, I & II," *Inf. Comput.*, vol. 100, no. 1, pp. 1–77, 1992.
[14] R. Meyer, "On boundedness in depth in the π-calculus," in *IFIP TCS*, ser. IFIP, G. Ausiello, J. Karhumäki, G. Mauri, and C.-H. L. Ong, Eds., vol. 273. Springer, 2008, pp. 477–489.
[15] A. Finkel and P. Schnoebelen, "Well-structured transition systems everywhere!" *Theor. Comput. Sci.*, vol. 256, no. 1-2, pp. 63–92, 2001.
[16] P. A. Abdulla, K. Cerans, B. Jonsson, and Y. Tsay, "Algorithmic analysis of programs with well quasi-ordered domains," *Inf. Comput.*, vol. 160, no. 1-2, pp. 109–127, 2000.
[17] A. Tiu, R. Goré, and J. E. Dawson, "A proof theoretic analysis of intruder theories," *Logical Methods in Computer Science*, vol. 6, no. 3, 2010.
[18] A. Tiu and J. E. Dawson, "Automating open bisimulation checking for the spi calculus," in *CSF'10*. IEEE Computer Society, 2010, pp. 307–321.
[19] M. Abadi and C. Fournet, "Mobile values, new names, and secure communication," in *POPL'01*. ACM Press, 2001, pp. 104–115.
[20] M. Abadi and A. D. Gordon, "A calculus for cryptographic protocols: The spi calculus," *Inf. Comp.*, vol. 148, no. 1, pp. 1–70, 99.
[21] M. D. Ryan and B. Smyth, "Applied pi-calculus," in *Formal Models and Techniques for Analyzing Security Protocols*, 2011.
[22] R. Chadha, Ştefan Ciobâcă, and S. Kremer, "Automated verification of equivalence properties of cryptographic protocols," in *ESOP'12*, ser. LNCS, vol. 7211. Springer, 2012, pp. 108–127.
[23] D. Baelde, S. Delaune, and L. Hirschi, "Partial order reduction for security protocols," in *CONCUR'15*, ser. LIPIcs, vol. 42, 2015, pp. 497–510.
[24] S. B. Fröschle, "Causality in security protocols and security APIs: Foundations and practical verification," Habilitationsschrift, University of Oldenburg, 2012.
[25] R. M. Needham and M. D. Schroeder, "Using encryption for authentication in large networks of computers," *Commun. ACM*, vol. 21, no. 12, pp. 993–999, 1978.
[26] D. E. Denning and G. M. Sacco, "Timestamps in key distribution protocols," *Commun. ACM*, vol. 24, no. 8, pp. 533–536, 1981.
[27] R. Meyer, "Structural stationarity in the π-calculus," Ph.D. dissertation, Carl von Ossietzky University of Oldenburg, 2009.
[28] J. B. Kruskal, "Well-quasi-ordering, the tree theorem, and Vazsonyi's conjecture," *Transactions of the American Mathematical Society*, vol. 95, no. 2, pp. 210–225, 1960.
[29] A. Finkel and J. Goubault-Larrecq, "Forward analysis for WSTS, part I: completions," in *STACS*, ser. LIPIcs, vol. 3, 2009, pp. 433–444.
[30] D. J. Otway and O. Rees, "Efficient and timely mutual authentication," *Operating Systems Review*, vol. 21, no. 1, pp. 8–10, 1987.
[31] M. Burrows, M. Abadi, and R. M. Needham, "A logic of authentication," in *SOSP'89*. ACM, 1989, pp. 1–13.
[32] E. D'Osualdo and C.-H. L. Ong, "On hierarchical communication topologies in the π-calculus," in *ESOP'16*, 2016.
[33] D. Zufferey, T. Wies, and T. A. Henzinger, "Ideal abstractions for well-structured transition systems," in *VMCAI'12'*, ser. LNCS, vol. 7148. Springer, 2012, pp. 445–460.

Appendix A
Proofs for Section II

A. Proof for Lemma 3

Proof. The proof of $\Gamma, X, Y \vdash M$ is constructed as follows:

$$\dfrac{\dfrac{\vdots}{\Gamma, X, Y \vdash f(X,Y)} \qquad \dfrac{\vdots}{\Gamma, f(X,Y), X, Y \vdash M}}{\Gamma, X, Y \vdash M}\ \textsc{Cut}$$

The left-premise is provable using the introduction rule for f and the Id rule. The right-premise is provable by weakening the proof of $\Gamma, f(X,Y) \vdash M$ using Lemma 2. □

B. Proof for Lemma 5

Proof. Termination is straightforward: the multiset of message sizes decreases in the well-founded multiset extension of the ordering on natural numbers. Since we have termination, to check confluence, it is enough to check for local confluence, i.e. if $\Gamma_0 \longrightarrow \Gamma_1$ and $\Gamma_0 \longrightarrow \Gamma_2$, then there exists Γ_3 such that $\Gamma_1 \longrightarrow^* \Gamma_3$ and $\Gamma_2 \longrightarrow^* \Gamma_3$. The only interesting case is when one or both reductions involve decomposing an encrypted message. Suppose $\Gamma_0 = \Gamma, \{M_1\}_{K_1}, \{M_2\}_{K_2}$ and $\Gamma_0 \vdash K_1$, $\Gamma_0 \vdash K_2$, then we have $\Gamma_0 \longrightarrow \Gamma_1$ and $\Gamma_0 \longrightarrow \Gamma_2$ with $\Gamma_1 = \Gamma, M_1, K_1, \{M_2\}_{K_2}$ and $\Gamma_2 = \Gamma, M_2, K_2, \{M_1\}_{K_1}$. Let $\Gamma_3 = \Gamma, M_1, K_1, M_2, K_2$, by Lemma 3 we have $\Gamma_1 \vdash K_2$ and $\Gamma_2 \vdash K_1$. Hence, we have $\Gamma_1 \longrightarrow \Gamma_3$ and $\Gamma_2 \longrightarrow \Gamma_3$, as required. The other cases work similarly. □

C. Proof for Lemma 6

Proof. Assume $\Gamma_1 \longrightarrow \Gamma_2$.

- Suppose $\Gamma_1 \vdash M$ for some M. Since every rewrite step is a message decomposition, it follows from Lemma 3 that $\Gamma_2 \vdash M$.
- Suppose $\Gamma_2 \vdash M$. The non-trivial case is when the rewrite step is a decomposition of an encryption, i.e.

$$\Gamma_1 = \Gamma, \{N\}_K \longrightarrow \Gamma, N, K = \Gamma_2$$

with $\Gamma_1 \vdash K$. The proof of $\Gamma_1 \vdash M$ is constructed as follows:

$$\dfrac{\dfrac{\vdots}{\Gamma, \{N\}_K \vdash K} \qquad \dfrac{\vdots}{\Gamma, \{N\}_K, N, K \vdash M}}{\Gamma, \{N\}_K \vdash M}$$

where the left premise is obtained by applying Lemma 2 to the proof of $\Gamma_2 \vdash M$. □

Appendix B
Proof of Lemma 7

We proceed by induction on the depth of the cut-free inference for $\Gamma_1 \vdash M$. In the base case, the inference is a single application of Rule Id and Γ_1 is the desired set of messages. In the induction step, we do a case analysis on the last rule used in the derivation:

Rule $\mathbf{E}_L$: by using the induction hypothesis on the left and right premises we obtain $\Gamma_l \supseteq \Gamma, \{M\}_K, K$ and $\Gamma_r \supseteq \Gamma, \{M\}_K, M, K, N$ respectively, with $\langle\Gamma_l\rangle \equiv_{\mathsf{kn}} \langle\Gamma, \{M\}_K\rangle$ and $\langle\Gamma_r\rangle \equiv_{\mathsf{kn}} \langle\Gamma, \{M\}_K, M, K\rangle$. Now consider the set $\Gamma' = \Gamma_l \cup \Gamma_r$ which contains N; by the above congruences, persistence and decryption, we have $\langle\Gamma_l, \Gamma_r\rangle \equiv_{\mathsf{kn}} \langle\Gamma_l, M, K\rangle \equiv_{\mathsf{kn}} \langle\Gamma_l, \{M\}_K, K\rangle \equiv_{\mathsf{kn}} \langle\Gamma, \{M\}_K\rangle$ as desired.

Rule $\mathbf{E}_R$: by using the induction hypothesis on the left and right premises we obtain $\Gamma_l \supseteq \Gamma, M$ and $\Gamma_r \supseteq \Gamma, K$ respectively, with $\langle\Gamma_l\rangle \equiv_{\mathsf{kn}} \langle\Gamma\rangle \equiv_{\mathsf{kn}} \langle\Gamma_r\rangle$. Now consider the set $\Gamma' = \Gamma_l \cup \Gamma_r \cup \{\{M\}_K\}$; by diffusion and persistence we have $\langle\Gamma\rangle \equiv_{\mathsf{kn}} \langle\Gamma_l, \Gamma_r\rangle \equiv_{\mathsf{kn}} \langle\Gamma_l, M, \Gamma_r, K\rangle \equiv_{\mathsf{kn}} \langle\Gamma_l, M, \Gamma_r, K, \{M\}_K\rangle \equiv_{\mathsf{kn}} \langle\Gamma_l, \Gamma_r, \{M\}_K\rangle \equiv_{\mathsf{kn}} \langle\Gamma'\rangle$ as desired.

Rule P_L: by using the induction hypothesis on the premise we obtain a $\Gamma' \supseteq \Gamma, (M,N), M, N, M'$ with $\langle \Gamma' \rangle \equiv_{\mathsf{kn}} \langle \Gamma, (M,N), M, N \rangle$. By applying diffusion and persistence we have $\langle \Gamma, (M,N), M, N \rangle \equiv_{\mathsf{kn}} \langle \Gamma, (M,N) \rangle$ so Γ' is the desired set.

Rule P_R: analogous to the case for Rule E_R.

Appendix C
Minsky Machines encodings

A. *Exploiting unbounded message size*

A first encoding exploits the fact that if no bound on the size of the messages is enforced, then the value of a counter can be encoded in the size of a message: $\mathcal{S}[\![0]\!]_a := a$, $\mathcal{S}[\![n+1]\!]_a = \{\mathcal{S}[\![n]\!]_a\}_a$. Then a counter c_i with value $n \in \mathbb{N}$ can be encoded as the message $\{\mathcal{S}[\![n]\!]_a\}_{k_i}$ for some names k_i, a. Since messages are persistent, the counter "key" k_i needs to be refreshed at every operation. Formally, a 2MM is a finite sequence of instructions $I_1, \dots, I_\ell$; instructions can be increments c_i++ (for $i \in \{0,1\}$) or decrements if not zero c_i−− **or goto** h (for $i \in \{0,1\}$ and $h \in \{1,\dots,\ell\}$). To each instruction I_j we associate a process name $\mathsf{S_j}[\,a, k_0, k_1\,]$. An instruction $I_j = c_i{+}{+}$ is encoded as

$$\begin{aligned}\mathcal{S}[\![I_j]\!] &= \big(\mathsf{S_j}[\,a,k_0,k_1\,] := \mathbf{in}(v : \{v\}_{k_i}).P_{\mathsf{inc}}\big)\\ P_{\mathsf{inc}} &= \nu k.\big(\langle\{\{v\}_a\}_k\rangle \parallel \mathsf{S_{j+1}}[\,a,k'_0,k'_1\,]\big)\end{aligned}$$

where $k'_i = k$ and $k'_{1-i} = k_{1-i}$. The encoding of an instruction $I_j = c_i{-}{-}$ **or goto** h is the definition

$$\begin{aligned}\mathcal{S}[\![I_j]\!] &= \big(\mathsf{S_j}[\,a,k_0,k_1\,] := A_{>0} + A_{=0}\big)\\ A_{>0} &= \mathbf{in}(v : \{\{v\}_a\}_{k_i}).\nu k.\big(\langle\{v\}_k\rangle \parallel \mathsf{S_{j+1}}[\,a,k'_0,k'_1\,]\big)\\ A_{=0} &= \mathbf{in}(\{a\}_{k_i}).\mathsf{S_h}[\,a,k_0,k_1\,]\end{aligned}$$

with k'_0, k'_1 as before. The initial configuration is encoded by the process $P_{\mathcal{S}} := \nu k_1, k_2.\big(\mathsf{S_1}[\,a,k_0,k_1\,] \parallel \langle\{a\}_{k_0}\rangle \parallel \langle\{a\}_{k_1}\rangle\big)$. To modify the value of a counter, it is necessary to possess the corresponding encryption key k_i. Clearly, since no key is ever output, the intruder cannot interfere with the value of a counter.

To reduce reachability of a control-state j of the 2MM to secrecy, it is sufficient to replace the definition of $\mathsf{S_j}$ with $\mathsf{S_j}[\,a,k_0,k_1\,] := \boldsymbol{\tau}.\nu x.(\mathsf{Secret}[\,x\,] \parallel \langle x\rangle)$.

B. *Exploiting unbounded encryption chains*

To each instruction I_j we associate a process name $\mathsf{D_j}[\,z,k_0,k_1\,]$. An instruction $I_j = c_i{+}{+}$ is encoded as

$$\begin{aligned}\mathcal{D}[\![I_j]\!] &= \big(\mathsf{D_j}[\,z,k_1,k_2\,] := \boldsymbol{\tau}.P'_{\mathsf{inc}}\big)\\ P'_{\mathsf{inc}} &= \nu k.\big(\langle\{k_i\}_k\rangle \parallel \mathsf{D_{j+1}}[\,z,k'_0,k'_2\,]\big)\end{aligned}$$

where $k'_i = k$ and $k'_{1-i} = k_{1-i}$. The encoding of an instruction $I_j = c_i{-}{-}$ **or goto** h is the definition

$$\begin{aligned}\mathcal{D}[\![I_j]\!] &= \big(\mathsf{D_j}[\,z,k_0,k_1\,] := A'_{>0} + A'_{=0}\big)\\ A'_{>0} &= \mathbf{in}(k : \{k\}_{k_i}).\mathsf{D_{j+1}}[\,z,k'_0,k'_1\,]\\ A'_{=0} &= \mathbf{in}(\{z\}_{k_i}).\mathsf{D_h}[\,z,k_0,k_1\,]\end{aligned}$$

with k'_0, k'_1 as before. The initial configuration is the process $P_{\mathcal{D}} := \nu k_0, k_1.\big(\mathsf{D_1}[\,z,k_0,k_1\,] \parallel \langle\{z\}_{k_0}\rangle \parallel \langle\{z\}_{k_1}\rangle\big)$.

Appendix D
Proof of Theorem 4

The idea of the proof is to define an encoding $\mathcal{W}[\![\text{-}]\!]$ of 2MM such that the runs of the 2MM can be all matched by some runs of its encoding, and, crucially, these runs only involve processes of depth at most 6 and message sizes at most 3. The encoding is weak in the sense that it also yields spurious runs which do not correspond to real runs of the 2MM. However, the encoding is constructed so that these spurious traces lead invariably to processes exceeding the depth k. The halting instruction of the 2MM is translated to a leak. Therefore we obtain that by considering secrecy relative to $\mathbb{D}_k \cap \mathbb{S}_s$ we are effectively considering reachability of the halting instruction using the non-spurious traces only.

Let us present the main ingredients of the encoding. A counter is represented by two names a, b and its value by the number of active processes $\mathsf{S}[\,a,b\,]$ in the current configuration. An increment is simply the creation of a new $\mathsf{S}[\,a,b\,]$ component. A decrement is implemented by a protocol that, when correctly followed, terminates with the effect of removing one $\mathsf{S}[\,a,b\,]$ component. Since there is no way to know if a component is *absent*, the branch of the computation accounting for the case when the counter to be decremented is zero, is selected non-deterministically, but with a subtle side-effect: two terms, one using a, the other using b, that attach to each the two names a long encryption chain. The two names a and b are then discarded and two fresh names are used by the instructions from then on. When the counter is zero, in a non-spurious run, the two names a, b are both known only to the process implementing the current instruction. When the run becomes spurious, because of a wrong guess that the counter's value is zero, there will be still some process $\mathsf{S}[\,a,b\,]$ knowing both names. This would form, together with the two long encryption chains created during the guess, an even longer chain that pushes the depth of the term beyond the depth bound respected in non-spurious runs.

The main complication in implementing this scheme is given by the persistence of output messages: we cannot consume them so we need to guard against interference of past communications the implementation of the current instruction. We do this by exploiting the fact that input actions can be "consumed".

$$\begin{aligned}\mathsf{S}[\,a,b\,] &:= \mathbf{in}(s : \{s\}_a).\nu s'.(\langle\{s'\}_s\rangle \parallel \mathsf{S}'[a,b,s'])\\ \mathsf{S}'[a,b,s'] &:= \mathbf{in}(\{s'\}_{s'}).\langle\{s',s'\}_{s'}\rangle\end{aligned}$$

Let $\vec{c} = \vec{c}$. To each instruction I_j we associate a process name $\mathsf{W_j}[\,\vec{c}\,]$. An instruction $I_j = c_i{+}{+}$ is encoded as

$$\begin{aligned}\mathcal{W}[\![I_j]\!] &= \big(\mathsf{W_j}[\,\vec{c}\,] := \boldsymbol{\tau}.P_{\mathsf{inc}}\big)\\ P_{\mathsf{inc}} &= \mathsf{S}[\,a_i,b_i\,] \parallel \mathsf{W_{j+1}}[\,\vec{c}\,]\end{aligned}$$

The encoding of an instruction $I_j = c_1{-}{-}$ **or goto** h is the following definition (the case involving c_2 is analogous):

$$\begin{aligned}\mathcal{W}[\![I_j]\!] &= \big(\mathsf{W_j}[\,\vec{c}\,] := \boldsymbol{\tau}.A_{>0} + \boldsymbol{\tau}.A_{=0}\big)\\ A_{>0} &= \nu s.(\langle\{s\}_a\rangle \parallel \mathsf{Dec_j}[\,\vec{c}, s\,])\\ \mathsf{Dec_j}[\,\vec{c}, s\,] &:= \mathbf{in}(s' : \{s'\}_s).(\langle\{s'\}_{s'}\rangle \parallel \mathsf{Dec'_j}[\,\vec{c}, s'\,])\\ \mathsf{Dec'_j}[\,\vec{c}, s'\,] &:= \mathbf{in}(\{s', s'\}_{s'}).\mathsf{W_{j+1}}[\,\vec{c}\,]\\ A_{=0} &= \nu a, b.\mathsf{W_h}[\,a, b, a_2, b_2\,] \parallel E^{2^k}_{a_1} \parallel E^{2^k}_{b_1}\end{aligned}$$

The term $A_{=0}$, here representing the effect of guessing that the counter value is zero, invokes the instruction at h refreshing the names representing the affected counter; at the same time it creates two encryption chains of length 2^k from a_1 and b_1 respectively:

$$E^m_y = \nu x_1, \dots, x_m.(\{y\}_{x_1} \parallel \{x_1\}_{x_2} \parallel \cdots \parallel \{x_m\}_{x_{m-1}})$$

This ensures that if the counter was really zero, the depth of the term would be at most k, if there is a process knowing both a_1 and b_1, a witness for a non-zero value for c_1, then the encryption chain would overall be 2^{k+1} long, which makes the overall depth $k + 1$. The rest of the proof is just careful checking of the above claims by examination of the possible transitions in the encoding.

Appendix E
Boundedness of NSSK

We now write a limit L_1, the denotation of which satisfies (C1) to (C3), for $P_0 = NS_0$, the sizing ς and $k = \mathrm{nest}_\nu(L_1) = 3$, allowing us to conclude that NS_0 is $(\varsigma, 3)$-bounded.

$$\begin{aligned}L_1 &= \mathsf{S_1}[\vec{s}\,]^\omega \parallel \mathsf{A_1}[\,\vec{a}\,]^\omega \parallel \mathsf{B_1}[\,\vec{b}\,]^\omega \parallel \langle a, b\rangle \parallel G \parallel L_2^\omega\\ G &= (\nu k.\langle M_{a,k}\rangle)^\omega \parallel (\nu k.\langle M_{b,k}\rangle)^\omega\\ M_{x,k} &= \{x, k, b, \{k, a\}_{k_{bs}}\}_{k_{as}}\\ L_2 &= \nu n.\big(\langle n\rangle \parallel \mathsf{A_2}[\,\vec{a}, n\,] \parallel L_3^\omega\big)\\ L_3 &= \nu k.\big(\langle M_{n,k}, \{k, a\}_{k_{bs}}\rangle \parallel \mathsf{A_3}[\,\vec{a}, k\,] \parallel L_4^\omega\big)\\ L_4 &= \nu s.\big(\langle\{s\}_k, \{s, s\}_k\rangle \parallel \mathsf{B_2}[\,\vec{b}, k, s\,]\big)\end{aligned}$$

Clearly, $P_0 \in [\![L_1]\!]$. It is easy, if tedious, to check that $[\![L_1]\!]$ is an inductive invariant up to ς. This is done by case analysis on the sequential processes involved in a transition. We consider all sequential processes $\mathsf{Q}[\,\vec{x}\,]$ that may appear in an instance of L_1. For each, we examine the actions available according to the definition for Q: we match each action to each possible derivable message of instances of L_1, provided it respects the sizing constraint. Applying the semantic rule will have the effect of removing $\mathsf{Q}[\,\vec{x}\,]$ from the term (which does not alter membership to $[\![L]\!]$ because $\mathbf{0} \in [\![\mathsf{Q}[\,\vec{x}\,]]\!]$), and replaces it with the action's continuation, with the appropriate substitution applied. It is then sufficient to check that said continuation will produce an instance of L_1. We say the continuation C is *covered* by some subterm L' of L_1 to mean that C is an instance of L', proving that the overall result of the transition is in $[\![L_1]\!]$.

Case $\mathsf{S_1}[\vec{s}\,]$: The pattern $n_x : (a, b, n_x)$ can only be matched by three derivable messages: (a, b, a), (a, b, b), and (a, b, n) where n stands for any instance of restriction νn in L_2 or some intruder-made name. Any attempt to substitute structured messages for n_x would result in a violation of the sizing constraint. All three messages would produce one instance of $\mathsf{S_1}[\,\vec{s}\,]$ which is clearly covered by L_1. Matching the first two messages results also in the addition of $\nu k.\langle M_{a,k}\rangle$ or $\nu k.\langle M_{b,k}\rangle$, which are covered by $G \parallel \mathsf{S_1}[\,\vec{s}\,]$). The third case (a, b, n) is covered by L_2^ω: $(\nu n, k.\langle n, M_{n,k}\rangle) \in [\![L_2]\!]$.

Case $\mathsf{A_1}[\,\vec{a}\,]$: The $\boldsymbol{\tau}$ action can always be executed, producing continuations that are all covered by $\mathsf{A_1}[\,\vec{a}\,]^\omega \parallel L_2^\omega$. Note that since a and b are active messages, the only contribution of $\langle a, b, n\rangle$ is making $\langle n\rangle$ active (as covered by L_2).

Case $\mathsf{A_2}[\,\vec{a}, n\,]$: The pattern $m, k : \{n, k, b, m\}_{k_{as}}$ can only match messages encrypted by k_{as}, which is unknown to the attacker: no irreducible instance of L_1 contains $\langle k_{as}\rangle$. The only messages encrypted with that key are the ones in instances of G and L_3^ω. Note however that none of the messages in G have the nonce n as first component (i.e. n is always different from a, b, and c). Therefore, the only matching messages are the $M_{n,k}$ in instances of L_3, which all share the n with some $\mathsf{A_2}[\,\vec{a}, n\,]$ in some instance of L_2. The resulting continuation $\langle\{k, a\}_{k_{bs}}\rangle \parallel \mathsf{A_3}[\,\vec{a}, k\,]$ is covered by L_3^ω.

Case $\mathsf{A_3}[\,\vec{a}, k\,]$: Since no k is ever leaked (it does not appear as an active message), the only messages that can match the pattern $s : \{s\}_k$ are the ones in instances of L_4, the only ones encrypted with a k shared by some occurrence of $\mathsf{A_3}[\,\vec{a}, k\,]$. The continuation $\langle\{s, s\}_k\rangle$ is covered by L_4.

Case $\mathsf{B_1}[\,\vec{b}\,]$: The pattern $k : \{k, a\}_{k_{bs}}$ can only be matched by messages encrypted with the secret key k_{bs}. The only candidates are therefore the $\{k, a\}_{k_{bs}}$ in instances of L_3. The continuations are covered by $\mathsf{B_1}[\,\vec{b}\,]^\omega$ and L_4^ω.

Case $\mathsf{B_2}[\,\vec{b}, s\,]$: Independently of the possible matches, if a transition is possible it will still trivially produce an instance of L_1.

Appendix F
Examples

A. *Proof of well-foundedness of Example 7*

The specification for the protocol above (in the notation of [11]) is a pair $P = (roles, script)$ where $roles = \{A, B\}$ and $script$ is defined as follows:

	A	B
(1)	$-A, B$	$-B, A$
(2)	$+N_A$	$-N_A$
(3)	$-\{1, K, N_A\}_{K_{AB}}$	$+\{1, K, N_A\}_{K_{AB}}, N_B$
(4)	$+\{2, K, N_B\}_{K_{AB}}$	$-\{2, K, N_B\}_{K_{AB}}$

Note that we assume that A, B are agent names, N_A, N_B and K are of atomic types. To show that the protocol P is well-founded, it is enough to show that $\prec^1_P$ is acyclic. The relation $\prec^1_P$ consists of two components $\Rightarrow_P$ and $\rightarrow_P$. The former is defined as: $(r, i) \Rightarrow_P (r', j)$ iff $r = r'$ and $i < j$. That is, it follows the order of the protocol script above. The relation $\rightarrow_P$ relates output in one step to input in another step of the protocol (possibly under different roles). In this case, there are only two possible such pairs:

- $(B,3) \to_P (A,3)$: This is justified by taking an encrypted term $m_e = \{1, K, N_A\}_{K_{AB}}$, substitutions $\sigma = \sigma' = id$ (identity), and checking that m_e is a subterm of both $\mathrm{msg}((B,3)) = \{1, K, N_A\}_{K_{AB}}, N_B$ and $\mathrm{msg}((A,3)) = \{1, K, N_A\}_{K_{AB}}$.
- $(A,4) \to_P (B,4)$: similar reasoning as above.

It is clear that $\prec^1_P = \Rightarrow_P \cup \to_P$ is acyclic, therefore the protocol is well-founded.

The key here is the tagging of messages, so, for example, output at $(A,4)$ cannot be fed into input at $(B,3)$ (which would create a cyle in $\prec^1_P$.)