Verification of three authentication protocols using BAN, SVO and Strand Spaces

F. R. Piva *A. J. Devegili* *R. Dahab*

Technical Report - IC-06-017 - Relatório Técnico

September - 2006 - Setembro

Verification of three authentication protocols using BAN, SVO and Strand Spaces

Fabio Rogerio Piva[*] Augusto Jun Devegili[†] Ricardo Dahab[‡]

Abstract

We compare three formal verification techniques for cryptographic protocols: BAN and SVO logics, and strand spaces, identifying some weaknesses of these logics and advantages of strand spaces. We also present new proofs of modified Yahalom, modified Kerberos and Woo-Lam Π.

1 Introduction

Cryptographic protocols are communication protocols that rely on cryptographic techniques to achieve their goals. Some cryptographic protocols are widely used in computer networks, e.g. SSL/TLS (Secure Socket Layer/Transport Layer Security), IPSec and Kerberos.

Designing a cryptographic protocol is not a simple task. There are various cryptographic techniques and algorithms available, such as symmetric and asymmetric ciphers, hash functions, pseudo-random number generators and message authentication codes. Although each of these may be correct in the sense that it is either impossible or unfeasible to break them separately, designing a *correct* cryptographic protocol using these algorithms and techniques is not straightforward. Several flaws have been unveiled in cryptographic protocols, some of which had been in use for quite some time.

These flaws lead to an obvious question: given a cryptographic protocol, how can one verify its correctness? Informal arguments clearly do not suffice. On the other hand, formal techniques might provide more rigorous rationale on the correctness (or flaws and attacks) of cryptographic protocols.

We may divide formal techniques for verification of cryptographic protocols in two main categories:

Theorem proving: protocol properties are represented as theorems. Proving a theorem means proving that the protocol does indeed have that property. When a goal cannot be proved and therefore the protocol is flawed, it may be difficult to find the reason of

[*]Instituto de Computação, Universidade Estadual de Campinas, 13081-970 Campinas, SP, Brazil. Partially supported by CNPq.

[†]Instituto de Computação, Universidade Estadual de Campinas, 13081-970 Campinas, SP, Brazil. Partially supported by CAPES.

[‡]Instituto de Computação, Universidade Estadual de Campinas, 13081-970 Campinas, SP, Brazil.

the attack. Theorem proving techniques include domain logics such as BAN [1] and SVO [2], automated induction in high order logic [3] and strand space theory [4];

Model checking: protocol runs are represented as a set of states. If a protocol is correct, then no state representing an attack should be reachable. As the set of possible states is usually too large, model checking techniques prune the state space, placing bounds on the number of participants or concurrent protocol sessions, and thus potentially ignoring some attacks. On the other hand, when an attack is found, it is possible to accurately trace the sequence of states that leads to the attack. Examples of model checking techniques include FDR [5], Murϕ [6], Brutus [7], and the NRL Analyzer [8].

Both categories have advantages and disadvantages. Model checking is more easily implemented as a computer program. In the general case, automated theorem proving is a hard problem. On the other hand, a theorem may represent a property valid for all possible instances of the protocol, whilst model checking may ignore some scenarios.

In this text, we consider three theorem proving techniques: BAN logic, SVO logic and strand spaces (Section 2). Using these techniques, we have analysed three authentication protocols: Yahalom modified by Burrows et al. (Section 3) and Kerberos with an extra message (Section 4) as described in [9], and the original Woo-Lam protocol [10] (Section 5). We have also provided a comparison of these formal techniques (Section 6) and some conclusions of this work (Section 7). As far as we know, some of these verifications have not been published, namely modified Kerberos in SVO and strand spaces, modified Yahalom in strand spaces, and Woo-Lam Π in BAN and SVO.

2 Formal verification techniques

In this section we briefly review BAN logic, SVO logic and strand spaces.

2.1 BAN logic

BAN logic [1], named after its creators Burrows, Abadi and Needham, was one of the first attempts of formal verification applied to cryptographic protocols. It is a many-sorted, temporal and belief logic with the following sorts of objects: principals, encryption keys and formulae (or statements). The logic considers two epochs: the past and the present. The present represents the current run of the protocol and the past is everything before it. Past beliefs are not necessarily present beliefs. Conjunction is the only propositional connective and it is represented by a comma. The notation defines a set of predicates, listed below. We use letters P or Q for principals, K for keys, and X or Y for formulae.

- $P \mid\equiv X$: P believes in X;
- $P \triangleleft X$: someone has sent P a message containing X;
- $P \mid\sim X$: P sent a message containing X and, by the time P sent it, P believed in it;
- $P \mapsto X$: P has jurisdiction over X, e.g. a trusted server generating good keys;

- $\sharp$(X): X has not been sent any time before the current run of the protocol. Usually, X is a nonce;
- $P \overset{K}{\longleftrightarrow} Q$: K is a good key for P and Q to communicate;
- $\overset{+P}{\mapsto} K$: K is P's public key;
- $P \overset{X}{\rightleftharpoons} Q$: X is a shared secret known only to P and Q;
- $\{X\}_K$: X encrypted with key K. Note that encryption is idealised in the sense that algebraic properties of different cryptosystems are not considered;
- $\langle X\rangle_Y$: X combined with Y so that Y's presence proves the identity of whoever uttered X.

Reasoning in BAN involves a set of postulates. Postulates have different versions according to the use of shared keys, public keys or shared secrets. Here we list the postulates in the shared key version.

Message meaning (MM): $\dfrac{P \models P \overset{K}{\longleftrightarrow} Q, P \triangleleft \{X\}_K}{P \models Q \mid\sim X}$

Nonce verification (NV): $\dfrac{P \models \sharp(X), P \models Q \mid\sim X}{P \models Q \models X}$

Jurisdiction (J): $\dfrac{P \models Q \Rightarrow X, P \models Q \models X}{P \models X}$

Message separation (MS): $\dfrac{P \triangleleft (X,Y)}{P \triangleleft X}$

Message decryption (MD): $\dfrac{P \models P \overset{K}{\longleftrightarrow} Q, P \triangleleft \{X\}_K}{P \triangleleft X}$

Message freshness (MF): $\dfrac{P \models \sharp(X)}{P \models \sharp(X,Y)}$

Before analysing a protocol using BAN, it is necessary to write an idealised version of the protocol. The idealisation process removes cleartext messages (they are useless in proving properties because they can be easily forged) and rewrites messages so that they follow the appropriate notation.

After the protocol is idealised, both initial assumptions and protocol goals are written down as formulae. Using the initial assumptions and the protocol as premises, the protocol verifier uses the postulates and tries to prove every protocol goal. If every goal is proved, the protocol is deemed correct.

2.2 SVO logic

SVO logic [2], named after its creators Syverson and van Oorschot, is derived from BAN and other BAN-like logics. Unlike BAN, SVO has a model-theoretic semantics and is proved to be sound with respect to that semantics. Its language has two subsets, the language of messages and the language of formulae. Typical propositional connectives may be used in formulae, such as $\neg$ for negation, $\wedge$ for conjunction and $\supset$ for material implication. SVO is an axiomatic modal logic: there are two inference rules, namely modus ponens and necessitation, and axioms are those of propositional calculus and axiom schemata similar to the postulates in BAN logic.

Let $\mathcal{T}$ be the set of primitive terms, i.e., the constant symbols that denote principals, keys, numbers etc. Then the language of messages, $\mathcal{M}_{\mathcal{T}}$, is the smallest language over $\mathcal{T}$ satisfying:

- X is a message if $X \in \mathcal{T}$;
- $F(X_1, \ldots, X_n)$ is a message if $X_1, \ldots, X_n$ are messages and F is any function;
- φ is a message if φ is a formula.

The language of formulae, $\mathcal{F}_{\mathcal{T}}$, is the smallest language satisfying:

- $P \overset{K}{\longleftrightarrow} Q, \mathrm{PK}_\psi(P, K), \mathrm{PK}_\sigma(P, K)$ and $\mathrm{PK}_\delta(P, K)$ are formulae when P and Q are principals and K is a key. $\mathrm{PK}_\psi(\cdot, \cdot), \mathrm{PK}_\sigma(\cdot, \cdot)$ and $\mathrm{PK}_\delta(\cdot, \cdot)$ denote encryption, signature and key agreement keys;
- $\mathrm{SV}(X, K, Y)$ is a formula when X and Y are messages and K is a key. $\mathrm{SV}(\cdot, \cdot, \cdot)$ denotes signature verification;
- $P \triangleleft X, P$ *received* X, P *says* $X, P \mid\!\sim X$, and $\sharp(X)$ are formulae when P is a principal and X is a message;
- $\langle X \rangle_{*P}$ is a formula when X is a formula and P is a principal. This is read as X according to P, meaning that P does not understand the contents of X, but P does recognise X if he sees X again;
- $\neg\varphi$ and $\varphi \wedge \psi$ are formulae if φ and ψ are formulae (other connectives may be definable in the same manner);
- $P \mid\!\equiv \varphi$ and $P \mapsto\!\!\!\Rightarrow \varphi$ are formulae when φ is a formula and P is a principal.

There are two inference rules:

Modus Ponens (MP): from φ and $\varphi \supset \psi$, infer ψ;

Necessitation: from $\vdash \varphi$, infer $P \mid\!\equiv \varphi$.

And the axiom schemata are:

Believing (B): for any principal P and formulae φ and ψ,

B1: $P \models \varphi \wedge P \models (\varphi \supset \psi) \supset P \models \psi$

B2: $P \models \varphi \supset P \models P \models \varphi$

Source association (SAA): keys are used to deduce the identity of the sender of a message

SAA1: $(P \overset{K}{\longleftrightarrow} Q \wedge R \textit{ received } \{X \textit{ from } Q\}_K) \supset (Q \mid\!\sim X \wedge Q \lhd K)$

SAA2: $(\mathrm{PK}_\sigma(Q, K) \wedge R \textit{ received } X \wedge \mathrm{SV}(X, K, Y)) \supset Q \mid\!\sim Y$

Key agreement (KA): session keys that are the result of good key-agreement keys are good

KA1: $(\mathrm{PK}_\delta(P, K_p) \wedge \mathrm{PK}_\delta(Q, K_q)) \supset P \xleftrightarrow{F_0(K_p,K_q)} Q$

KA2: $\varphi \equiv \varphi[F_0(K, K')/F_0(K', K)]$

Receiving (R): a principal receives each component of messages that he has received, including encrypted messages provided he has the corresponding decryption key

R1: $P \textit{ received } (X_1, \ldots, X_n) \supset P \textit{ received } X_i$

R2: $(P \textit{ received } \{X\}_K \wedge P \lhd K^{-1}) \supset P \textit{ received } X$

R3: $P \textit{ received } [X]_K \supset P \textit{ received } X$

Seeing (S): a principal sees everything he receives

S1: $P \textit{ received } X \supset P \lhd X$

S2: $P \lhd (X_1, \ldots, X_N) \supset P \lhd X_i$

S3: $(P \lhd X_1 \wedge \ldots \wedge P \lhd X_N) \supset P \lhd F(X_1, \ldots, X_n)$

Comprehending (C): if a principal comprehends a message and sees a function of it, then he understands that this is what he is seeing

C1: $P \models (P \lhd F(X)) \supset P \models (P \lhd X)$

Saying (SA): a principal sees what he says

SA1: $P \mid\!\sim (X_1, \ldots, X_n) \supset (P \mid\!\sim X_i \wedge P \lhd X_i)$

SA2: $P \textit{ says } (X_1, \ldots, X_n) \supset (P \mid\!\sim (X_1, \ldots, X_n) \wedge P \textit{ says } X_i)$

Jurisdiction (JA): P's word is law for some φ

J1: $(P \Rightarrow \varphi \wedge P \textit{ says } \varphi) \supset \varphi$

Freshness (FA): messages containing some fresh message are also fresh

F1: $\sharp(X_i) \supset \sharp(X_1, \ldots, X_n)$

F2: $\sharp(X_1, \ldots, X_n) \supset \sharp((F(X_1, \ldots, X_n))$

Nonce verification (NVA): fresh messages are current messages

NV1: $(\sharp(X) \wedge P \mid\!\sim X) \supset P \textit{ says } X$

Symmetric goodness of shared keys (SGSK):

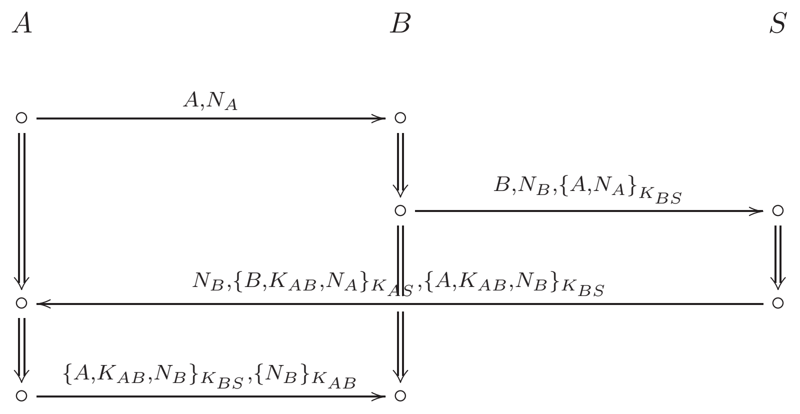
SGSK1: $P \overset{K}{\longleftrightarrow} Q \equiv Q \overset{K}{\longleftrightarrow} P$

Reasoning in SVO is similar to reasoning in BAN: premises are written down based on initial assumptions and protocol steps. Protocol goals are theorems that must be proved in order to consider the protocol correct: goals should be formulae derived from premises and axioms using the inference rules.

2.3 Strand spaces

In strand space theory [4], a cryptographic protocol is modelled as a directed acyclic graph and some conclusions are based on this representation. Proving a protocol property in strand spaces means proving some theorem on the structure of the corresponding graph.

A *strand* is a sequence of events engaged in by a principal during a protocol run. A *strand space* Σ is a set of strands. Figure 1 is an example of a strand space with three strands, one for each principal in the protocol (A, B and S). These are *regular strands.* A *penetrator strand* is a strand that models the capabilities of a penetrator. A *bundle* is a subspace of some strand space.



$$\begin{tikzcd}
A & B & S \\
\circ \arrow[r, "{A,N_A}"] \arrow[ddd, Rightarrow] & \circ \arrow[d, Rightarrow] & \\
 & \circ \arrow[r, "{B,N_B,\{A,N_A\}_{K_{BS}}}"] \arrow[dd, Rightarrow] & \circ \arrow[d, Rightarrow] \\
 & & \circ \arrow[ll, "{N_B,\{B,K_{AB},N_A\}_{K_{AS}},\{A,K_{AB},N_B\}_{K_{BS}}}"'] \\
\circ \arrow[d, Rightarrow] & & \\
\circ \arrow[r, "{\{A,K_{AB},N_B\}_{K_{BS}},\{N_B\}_{K_{AB}}}"] & \circ &
\end{tikzcd}$$

Figure 1: Example of a strand space

The set of messages is the free term algebra A generated by the concatenation operation on a set A of terms. The *subterm relation* on A, denoted by $t_0 \sqsubset t_1$, indicates that t_0 is a subterm of t_1. A *signed term* identifies whether a term is being sent $(+t)$ or received $(-t)$.

A strand space over A is a set Σ with a *trace* mapping $tr : \Sigma \to (\pm\mathsf{A})^*$, where $(\pm\mathsf{A})^*$ is a sequence of signed terms. The *length* of a trace is the number of terms in the sequence and is denoted by $length(\cdot)$. A *node* is a pair $\langle s, i\rangle$ with $s \in \Sigma$ a strand and i an integer with $1 \leqslant i \leqslant length(tr(s))$. The set of nodes is $\mathcal{N}$. If $n = \langle s, i\rangle \in \mathcal{N}$ is a node, then $index(n) = i$, $strand(n) = s$, $term(n) = (tr(s))_i$ and $uns\_term(n)$ is the unsigned term of $term(n)$.

There is an edge $n_1 \to n_2$ iff $term(n_1) = +a$ and $term(n_2) = -a$, indicating that term a is being sent from n_1 to n_2. If $n_1 = \langle s, i\rangle$ and $n_2 = \langle s, i+1\rangle$, then there is an edge $n_1 \Rightarrow n_2$, meaning that n_1 is an immediate predecessor of n_2. Generally, $n \Rightarrow^+ n'$ if n precedes n' in the same strand.

Let I be a set of unsigned terms. Then a node $n \in \mathcal{N}$ is an *entry point* for I iff $term(n) = +t$ for some $t \in I$, and for all n' such that $n' \Rightarrow^+ n$, $term(n') \notin I$. An unsigned term t *originates* on $n \in \mathcal{N}$ iff n is an entry point for the set $I = \{t' : t \sqsubset t'\}$. An unsigned term t is *uniquely originating* iff t originates on a unique $n \in \mathcal{N}$.

A *bundle* is a finite subgraph of the graph representing the protocol. Let $\to_{\mathcal{C}} \subset \to, \Rightarrow_{\mathcal{C}} \subset \Rightarrow$ and let $\mathcal{C} = \langle \mathcal{N}_{\mathcal{C}}, (\to_{\mathcal{C}} \cup \Rightarrow_{\mathcal{C}})\rangle$ be a subgraph of $\langle \mathcal{N}, (\to \cup \Rightarrow)\rangle$. $\mathcal{C}$ is a bundle if:

1. $\mathcal{C}$ is finite;
2. if $n_2 \in \mathcal{N}_{\mathcal{C}}$ and $term(n_2)$ is negative, then there is a unique n_1 such that $n_1 \to_{\mathcal{C}} n_2$;
3. if $n_2 \in \mathcal{N}_{\mathcal{C}}$ and $n_1 \Rightarrow n_2$, then $n_1 \Rightarrow_{\mathcal{C}} n_2$;
4. $\mathcal{C}$ is acyclic.

The $\mathcal{C}$-height of a strand s in a bundle $\mathcal{C}$ is the largest i such that $\langle s, i\rangle \in \mathcal{C}$. Also, $\mathcal{C}\text{-}trace(s) = \langle (tr(s))_1, \ldots, (tr(s))_m\rangle$, where $m = \mathcal{C}\text{-height}(s)$.

Let $\mathcal{S} \subset \to \cup \Rightarrow$ be a set of edges. Then $\prec_{\mathcal{S}}$ is the transitive closure of $\mathcal{S}$, and $\preceq_{\mathcal{S}}$ is the reflexive, transitive closure of $\mathcal{S}$. Both $\prec_{\mathcal{S}}$ and $\preceq_{\mathcal{S}}$ are subsets of $\mathcal{N}_{\mathcal{S}} \times \mathcal{N}_{\mathcal{S}}$, where $\mathcal{N}_{\mathcal{S}}$ is the set of nodes incident to some edge in $\mathcal{S}$.

We now state a few lemmas that are useful in proofs in strand space theory. Lemma 2.1 states that every set of nodes in a bundle $\mathcal{C}$ has a $\preceq_{\mathcal{C}}$ -minimal node n. Lemma 2.2 states that such a node n is positive. If the set of nodes is constructed via the subterm relation, then Lemma 2.3 states that n is an originating node.

Lemma 2.1. *Suppose $\mathcal{C}$ is a bundle. Then $\preceq_{\mathcal{C}}$ is a partial order, i.e., a reflexive, anti-symmetric, transitive relation. Every non-empty subset of the nodes in $\mathcal{C}$ has $\preceq_{\mathcal{C}}$ -minimal members.*

Lemma 2.2. *Suppose $\mathcal{C}$ is a bundle, and $\mathcal{S} \subseteq \mathcal{C}$ is a set of nodes such that*

$$\forall m, m' \ uns\_term(m) = uns\_term(m') \textit{ implies } (m \in \mathcal{S} \textit{ iff } m' \in \mathcal{S}).$$

If n is a $\preceq_{\mathcal{C}}$ -minimal member of $\mathcal{S}$, then the sign of n is positive.

Proof. Suppose $term(n)$ were negative. Then, by the definition of bundle, $n' \rightarrow n$ for some $n' \in \mathcal{C}$ and $uns\_term(n) = uns\_term(n')$. Hence $n' \in \mathcal{S}$, contradicting the minimality of n. □

Lemma 2.3. *Suppose $\mathcal{C}$ is a bundle, $t \in \mathsf{A}$ and $n \in \mathcal{C}$ is a $\preceq_{\mathcal{C}}$ -minimal element of $\{m \in \mathcal{C} : t \sqsubset term(m)\}$. Then node n is an originating occurrence for t.*

Proof. Obviously $t \sqsubset term(n)$. By Lemma 2.2, the sign of n is positive. If $n' \Rightarrow^{+} n$, then $n' \in \mathcal{C}$. Because of the minimality of n, $t \not\sqsubset term(n')$. Thus n is originating for t. □

Let $\mathsf{T} \subseteq \mathsf{A}$ be a set of atomic messages and $\mathsf{K} \subseteq \mathsf{A}$ be a set of cryptographic keys disjoint from T. We define an unary operator $inv : \mathsf{K} \rightarrow \mathsf{K}$ and two binary operators: $encr : \mathsf{K} \times \mathsf{A} \rightarrow \mathsf{A}$ and $join : \mathsf{A} \times \mathsf{A} \rightarrow \mathsf{A}$. Another representation for these operators is K^{-1} for $inv(K)$, $\{m\}_K$ for $encr(K, M)$ and $a\ b$ for $join(a, b)$.

The original strand space theory makes some freeness assumptions on the free term algebra A:

Axiom 2.4. *For $m, m' \in \mathsf{A}$ and $K, K' \in \mathsf{K}$, $\{m\}_K = \{m'\}_{K'} \Rightarrow m = m' \wedge K = K'$.*

Axiom 2.5. *For $m_0, m_0', m_1, m_1' \in \mathsf{A}$ and $K, K' \in \mathsf{K}$,*

1. $m_0 m_1 = m_0' m_1' \Rightarrow m_0 = m_0' \wedge m_1 = m_1'$
2. $m_0 m_1 \neq \{m_0'\}_{K'}$
3. $m_0 m_1 \notin \mathsf{K} \cup \mathsf{T}$
4. $\{m_0\}_K \notin \mathsf{K} \cup \mathsf{T}$.

Given Axiom 2.5, we define the width of terms. We also define precisely the subterm relation.

Definition 2.6. *If $m \in \mathsf{K} \cup \mathsf{T}$ or if $m = \{m_0\}_K$, then $width(m) = 1$. If $m = m_0 m_1$, then $width(m) = width(m_0) + width(m_1)$.*

Definition 2.7. *The subterm relation $\sqsubset$ is defined inductively as the smallest relation such that:*

1. $a \sqsubset a$
2. $a \sqsubset \{g\}_K$ *if* $a \sqsubset g$
3. $a \sqsubset gh$ *if* $a \sqsubset g$ *or* $a \sqsubset h$.

The penetrator model in strand spaces contains a set of keys known to the penetrator, $\mathsf{K_P}$, and a set of *penetrator strands* that represent possible penetrator behaviours. Each penetrator strand may contain the following *penetrator traces*:

Text message (M): $\langle +t \rangle$ where $t \in \mathsf{T}$

Flushing (F): $\langle -g \rangle$

Tee (T): $\langle -g, +g, +g \rangle$

Concatenation (C): $\langle -g, -h, +gh \rangle$

Separation (S): $\langle -gh, +g, +h \rangle$

Key (K): $\langle +K \rangle$ where $K \in \mathsf{K_P}$

Encryption (E): $\langle -K, -h, +\{h\}_K \rangle$

Decryption (D): $\langle -K^{-1}, -\{h\}_K, +h \rangle$.

We now define an infiltrated strand space.

Definition 2.8. *An* infiltrated strand space *is a pair* $(\Sigma, \mathcal{P})$ *with* Σ *a strand space and* $\mathcal{P} \subseteq \Sigma$ *such that* $tr(p)$ *is a penetrator trace for every* $p \in \mathcal{P}$.

A strand $s \in \Sigma$ *is a* penetrator strand *if it belongs to* $\mathcal{P}$*, and a node is a* penetrator node *if the strand it lies on is a penetrator strand.*

A node n *is an* ***M, F*** *etc. node if* n *lies on a penetrator strand with a trace of kind* ***M, F*** *etc.*

From the definition of penetrator, penetrator strands and penetrator traces, it is possible to state the following general proposition:

Proposition 2.9. *Let* $\mathcal{C}$ *be a bundle, and let* $K \in \mathsf{K} \setminus \mathsf{K_P}$.

If K *never originates on a regular node, then* $K \not\sqsubset term(n)$ *for any node* $n \in \mathcal{C}$*. In particular, for any penetrator node* $p \in \mathcal{C}, K \not\sqsubset term(p)$.

If $\mathsf{k} \subseteq \mathsf{K}$, then a k-ideal of A is a subset I of A such that for all $h \in I, g \in \mathsf{A}$ and $K \in \mathsf{k}$, $hg, gh \in I$ and $\{h\}_K \in I$. The smallest k-ideal containing h is denoted $I_\mathsf{k}[h]$. From this definition, we state the following set of propositions, theorems and corollaries. Proofs are available in [4].

Proposition 2.10. *If* $S \subseteq \mathsf{A}$*, then* $I_\mathsf{k}[S] = \bigcup_{x \in S} I_\mathsf{k}[x]$.

Proposition 2.11. *Suppose* $K \in \mathsf{K}, S \subseteq \mathsf{A}$*, and for every* $s \in S$*,* s *is simple and is not of the form* $\{g\}_K$*. If* $\{h\}_K \in I_\mathsf{k}[S]$ *for* $K \in \mathsf{K}$*, then* $K \in \mathsf{k}$.

Proposition 2.12. *Suppose* $S \subseteq \mathsf{A}$*, and every* $s \in S$ *is simple. If* $gh \in I_\mathsf{k}[S]$*, then either* $g \in I_\mathsf{k}[S]$ *or* $h \in I_\mathsf{k}[S]$.

Theorem 2.13. *Suppose* $\mathcal{C}$ *is a bundle over* $\mathsf{A}, S \subseteq \mathsf{T} \cup \mathsf{K}, \mathsf{k} \subseteq \mathsf{K}$*, and* $\mathsf{K} \subseteq S \cup \mathsf{k}^{-1}$*. Then* $I_\mathsf{k}[S]$ *is honest.*

Corollary 2.14. *Suppose* $\mathcal{C}$ *is a bundle,* $\mathsf{K} = S \cup \mathsf{k}^{-1}$ *and* $S \cap \mathsf{K_P} = \emptyset$*. If there exists a node* $m \in \mathcal{C}$ *such that* $term(m) = I_\mathsf{k}[S]$*, then there exists a regular node* $n \in \mathcal{C}$ *such that* n *is an entry point for* $I_\mathsf{k}[S]$.

Corollary 2.15. *Suppose* $\mathcal{C}$ *is a bundle,* $\mathsf{K} = S \cup \mathsf{k}^{-1}; S \cap \mathsf{K_P} = \emptyset$*, and no regular node in* $\mathcal{C}$ *is an entry point for* $I_\mathsf{k}[S]$*. Then any term of the form* $\{g\}_K$ *for* $K \in S$ *does not originate on a penetrator strand.*

3 The Yahalom protocol modified by Burrows et al.

The Yahalom protocol [9] is a server-based key establishment protocol that provides key authentication for both principals engaged in the protocol. Burrows et al. suggested a modification of the Yahalom protocol, and Syverson showed that this modification was flawed [11]. Figure 2 describes this modified version of the Yahalom protocol, which we call *modified Yahalom* throughout this text.

1. $A \rightarrow B : A, N_A$
2. $B \rightarrow S : B, N_B, \{A, N_A\}_{K_{BS}}$
3. $S \rightarrow A : N_B, \{B, K_{AB}, N_A\}_{K_{AS}}, \{A, K_{AB}, N_B\}_{K_{BS}}$
4. $A \rightarrow B : \{A, K_{AB}, N_B\}_{K_{BS}}, \{N_B\}_{K_{AB}}$

Figure 2: Modified Yahalom: the Yahalom protocol modified by Burrows et al.

Note that the notation $P \rightarrow Q : X$ shown in figure 2 means that principal P transmits message X to principal Q. This notation is well known, and is used throughout this text to describe cryptographic protocols.

In modified Yahalom the server works as a trusted key generation center. It is firstly contacted by the responder, who requests a new session with the initiator. The server then generates a new session key and encrypts it in two components (one using the initiator's long term key and the other using the responder's long term key). Those components are delivered to the initiator, who must keep one for himself and transmit the other to the responder.

In this section we prove that the modified Yahalom fails to authenticate the responder to the initiator.

3.1 BAN verification of modified Yahalom

3.1.1 Idealisation

1. $A \rightarrow B :$
2. $B \rightarrow S : \{N_A\}_{K_{BS}}$
3. $S \rightarrow A : \left\{A \xleftrightarrow{K_{AB}} B, \sharp(A \xleftrightarrow{K_{AB}} B), N_A\right\}_{K_{AS}}, \left\{A \xleftrightarrow{K_{AB}} B, \sharp(A \xleftrightarrow{K_{AB}} B), N_B\right\}_{K_{BS}}$
4. $A \rightarrow B : \left\{A \xleftrightarrow{K_{AB}} B, \sharp(A \xleftrightarrow{K_{AB}} B), N_B\right\}_{K_{BS}}, \left\{A \xleftrightarrow{K_{AB}} B, N_B\right\}_{K_{AB}}$

3.1.2 Assumptions

A1. $A \mid\equiv \sharp(N_A)$

A2. $B \models \sharp(N_B)$

A3. $A \models A \overset{K_{AS}}{\longleftrightarrow} S$

A4. $B \models B \overset{K_{BS}}{\longleftrightarrow} S$

A5. $S \models A \overset{K_{AS}}{\longleftrightarrow} S$

A6. $S \models B \overset{K_{BS}}{\longleftrightarrow} S$

A7. $S \models S \Rrightarrow A \overset{K_{AB}}{\longleftrightarrow} B$

A8. $A \models S \Rrightarrow A \overset{K_{AB}}{\longleftrightarrow} B$

A9. $B \models S \Rrightarrow A \overset{K_{AB}}{\longleftrightarrow} B$

A10. $S \models S \Rrightarrow \sharp(A \overset{K_{AB}}{\longleftrightarrow} B)$

A11. $A \models S \Rrightarrow \sharp(A \overset{K_{AB}}{\longleftrightarrow} B)$

A12. $B \models S \Rrightarrow \sharp(A \overset{K_{AB}}{\longleftrightarrow} B)$

3.1.3 Analysis

D1. $$\frac{S \triangleleft \{N_A\}_{K_{BS}}, S \models B \overset{K_{BS}}{\longleftrightarrow} S}{S \models B \mid\sim (N_A)} \text{ (M2, A6 by MM)}$$

D2. $$\frac{A \triangleleft \left\{A \overset{K_{AB}}{\longleftrightarrow} B, \sharp(A \overset{K_{AB}}{\longleftrightarrow} B), N_A\right\}_{K_{AS}}, A \models A \overset{K_{AS}}{\longleftrightarrow} S}{A \models S \mid\sim (A \overset{K_{AB}}{\longleftrightarrow} B, \sharp(A \overset{K_{AB}}{\longleftrightarrow} B), N_A)} \text{ (M3, A3 by MM)}$$

D3. $$\frac{A \models S \mid\sim (A \overset{K_{AB}}{\longleftrightarrow} B, \sharp(A \overset{K_{AB}}{\longleftrightarrow} B), N_A), A \models \sharp(N_A)}{A \models S \models (A \overset{K_{AB}}{\longleftrightarrow} B, \sharp(A \overset{K_{AB}}{\longleftrightarrow} B), N_A)} \text{ (D2, A1 by NV)}$$

D4. $$\frac{A \models S \models (A \overset{K_{AB}}{\longleftrightarrow} B), A \models S \Rrightarrow (A \overset{K_{AB}}{\longleftrightarrow} B)}{A \models (A \overset{K_{AB}}{\longleftrightarrow} B)} \text{ (D3, A8 by J)}$$

D5. $$\frac{A \models S \models \sharp(A \overset{K_{AB}}{\longleftrightarrow} B), A \models S \Rrightarrow \sharp(A \overset{K_{AB}}{\longleftrightarrow} B)}{A \models \sharp(A \overset{K_{AB}}{\longleftrightarrow} B)} \text{ (D3, A11 by J)}$$

From the sentences above, we may conclude that the protocol achieves key establishment for entity A. A believes that not only the key is good for communication with B, but also that it was recently generated.

D6. $$\frac{B \triangleleft \left\{A \xleftrightarrow{K_{AB}} B, \sharp(A \xleftrightarrow{K_{AB}} B), N_B\right\}_{K_{BS}}, B \models B \xleftrightarrow{K_{BS}} S}{B \models S \mid\!\sim (A \xleftrightarrow{K_{AB}} B, \sharp(A \xleftrightarrow{K_{AB}} B), N_B), B \triangleleft A \xleftrightarrow{K_{AB}} B} \text{ (M4, A4 by MM)}$$

D7. $$\frac{B \models S \mid\!\sim (A \xleftrightarrow{K_{AB}} B, \sharp(A \xleftrightarrow{K_{AB}} B), N_B), B \models \sharp(N_B)}{B \models S \models (A \xleftrightarrow{K_{AB}} B, \sharp(A \xleftrightarrow{K_{AB}} B), N_B)} \text{ (D7, A2 by NV)}$$

D8. $$\frac{B \models S \models (A \xleftrightarrow{K_{AB}} B), B \models S \Rightarrow (A \xleftrightarrow{K_{AB}} B)}{B \models A \xleftrightarrow{K_{AB}} B} \text{ (D7, A9 by J)}$$

D9. $$\frac{B \models S \models \sharp(A \xleftrightarrow{K_{AB}} B), B \models S \Rightarrow \sharp(A \xleftrightarrow{K_{AB}} B)}{B \models \sharp(A \xleftrightarrow{K_{AB}} B)} \text{ (D7, A10 by J)}$$

From the sentences above, we may conclude that the protocol achieves key establishment for entity B. B believes that not only the key is good for communication with A, but also that it was recently generated.

D10. $$\frac{B \models A \xleftrightarrow{K_{AB}} B, B \triangleleft \{N_b\}_{K_{AB}}}{B \models A \mid\!\sim N_B, A \xleftrightarrow{K_{AB}} B} \text{ (M4, D8 by MM)}$$

D11. $$\frac{B \models A \mid\!\sim N_B, A \xleftrightarrow{K_{AB}} B, B \models \sharp(N_B, A \xleftrightarrow{K_{AB}} B)}{B \models A \models N_B, A \xleftrightarrow{K_{AB}} B} \text{ (D10, A2 by NV)}$$

This derivation leads us to conclude that the protocol achieves liveness of A to B. It also shows us that B is able to conclude that A believes the session key is good.

3.2 SVO verification of modified Yahalom

3.2.1 Assumptions

A1. $A \models \sharp(N_A)$

A2. $B \models \sharp(N_B)$

A3. $A \models A \xleftrightarrow{K_{AS}} S$

A4. $B \models B \xleftrightarrow{K_{BS}} S$

A5. $S \models A \xleftrightarrow{K_{AS}} S$

A6. $S \models B \xleftrightarrow{K_{BS}} S$

A7. $S \models S \Rightarrow A \xleftrightarrow{K_{AB}} B$

A8. $A \models S \Rightarrow A \xleftrightarrow{K_{AB}} B$

A9. $B \models S \Rrightarrow A \xleftrightarrow{K_{AB}} B$

A10. $S \models S \Rrightarrow \sharp(A \xleftrightarrow{K_{AB}} B)$

A11. $A \models S \Rrightarrow \sharp(A \xleftrightarrow{K_{AB}} B)$

A12. $B \models S \Rrightarrow \sharp(A \xleftrightarrow{K_{AB}} B)$

3.2.2 Annotation

N1. $B \lhd (A, N_A)$

N2. $S \lhd (B, N_B), \{A, N_A\}_{K_{BS}}$

N3. $A \lhd (N_B), \{B, K_{AB}, N_A\}_{K_{AS}}, \{A, K_{AB}, N_B\}_{K_{BS}}$

N4. $B \lhd \{A, K_{AB}, N_B\}_{K_{BS}}, \{N_B\}_{K_{AB}}$

3.2.3 Comprehension

C1. $B \models B \lhd (A, \langle N_A \rangle_{*B})$

C2. $S \models S \lhd (B, \langle N_B \rangle_{*S}), \{A, \langle N_A \rangle_{*S}\}_{K_{BS}}$

C3. $A \models A \lhd \langle N_B \rangle_{*A}, \{B, \langle K_{AB} \rangle_{*A}, N_A\}_{K_{AS}}, \langle \{A, K_{AB}, N_B\}_{K_{BS}} \rangle_{*A}$

C4. $B \models B \lhd \{A, \langle K_{AB} \rangle_{*B}, N_B\}_{K_{BS}}, \{N_B\}_{\langle K_{AB} \rangle_{*B}}$

3.2.4 Interpretation

I1.

I2. $B \models B \lhd (A, \langle N_A \rangle_{*B}) \rightarrow S \models S \lhd (B, \langle N_B \rangle_{*S}), \{A, \langle N_A \rangle_{*S}\}_{K_{BS}}$

I3. $A \models A \lhd \langle N_B \rangle_{*A}, \{B, \langle K_{AB} \rangle_{*A}, N_A\}_{K_{AS}}, \langle \{A, K_{AB}, N_B\}_{K_{BS}} \rangle_{*A} \rightarrow$
$A \models A \lhd \left\{B, A \xleftrightarrow{\langle K_{AB} \rangle_{*A}} B, \sharp(A \xleftrightarrow{\langle K_{AB} \rangle_{*A}} B), N_A\right\}_{K_{AS}}, \langle \{A, K_{AB}, N_B\}_{K_{BS}} \rangle_{*A}$

I4. $B \models B \lhd \{A, \langle K_{AB} \rangle_{*B}, N_B\}_{K_{BS}}, \{N_B\}_{\langle K_{AB} \rangle_{*B}} \rightarrow$
$B \models B \lhd \left\{A, A \xleftrightarrow{\langle K_{AB} \rangle_{*B}} B, \sharp(A \xleftrightarrow{\langle K_{AB} \rangle_{*B}} B), N_B\right\}_{K_{BS}}$

I5. $B \models B \lhd \{A, |K_{AB}|_{*B}, N_B\}_{K_{BS}}, \{A, \langle K_{AB} \rangle_{*B}, N_B\}_{K_{BS}}, \{N_B\}_{\langle K_{AB} \rangle_{*B}} \rightarrow$
$B \models \left\{N_B, A \xleftrightarrow{\langle K_{AB} \rangle_{*B}} B\right\}_{\langle K_{AB} \rangle_{*B}}$

3.2.5 Derivation

D1. $A \models A \triangleleft \left\{ B, A \xleftrightarrow{\langle K_{AB} \rangle_{*A}} B, \sharp(A \xleftrightarrow{\langle K_{AB} \rangle_{*A}} B,), N_A \right\}_{K_{AS}}, \langle \{A, K_{AB}, N_B\}_{K_{BS}} \rangle_{*A}$ (I3, C3 by MP)

D2. $D2a : A \models S \mid\!\sim (B, A \xleftrightarrow{\langle K_{AB} \rangle_{*A}} B, \sharp(A \xleftrightarrow{\langle K_{AB} \rangle_{*A}} B), N_A)$ (D1, A3 by SAA)

$D2b : A \models S \ has \ (B, A \xleftrightarrow{\langle K_{AB} \rangle_{*A}} B, \sharp(A \xleftrightarrow{\langle K_{AB} \rangle_{*A}} B), N_A)$ (D1, A3 by SA)

D3. $A \models S \ says \ (B, A \xleftrightarrow{\langle K_{AB} \rangle_{*A}} B, \sharp(A \xleftrightarrow{\langle K_{AB} \rangle_{*A}} B), N_A)$ (D2a, by FA, NVA)

D4. D4a: $A \models A \xleftrightarrow{\langle K_{AB} \rangle_{*A}} B$ (D3, A8 by SA, JA)

D4b: $A \models \sharp(A \xleftrightarrow{\langle K_{AB} \rangle_{*A}} B)$ (D3, A8 by SA, JA)

Derivations D4a and D4b show that A can conclude that the key was recently generated and good for communication with B.

D5. $B \models B \triangleleft \left\{ A, A \xleftrightarrow{\langle K_{AB} \rangle_{*B}} B, \sharp(A \xleftrightarrow{\langle K_{AB} \rangle_{*B}} B), N_B \right\}_{K_{BS}}$ (I4, C4 by MP)

D6. D6a: $B \models S \mid\!\sim (A, A \xleftrightarrow{\langle K_{AB} \rangle_{*B}} B, \sharp(A \xleftrightarrow{\langle K_{AB} \rangle_{*B}} B), N_B)$ (D5, A4 by SAA)

D6b: $B \models S \ has \ (A, A \xleftrightarrow{\langle K_{AB} \rangle_{*B}} B, \sharp(A \xleftrightarrow{\langle K_{AB} \rangle_{*B}} B), N_B)$ (D5, A4 by SAA)

D7. $B \models S \ says \ (A \xleftrightarrow{\langle K_{AB} \rangle_{*B}} B, \sharp(A \xleftrightarrow{\langle K_{AB} \rangle_{*B}} B),)$ (D6a, A2 by NVA)

D8. D8a: $B \models A \xleftrightarrow{\langle K_{AB} \rangle_{*B}} B$ (D7, A9, A12 by JA)

D8b: $B \models \sharp(A \xleftrightarrow{\langle K_{AB} \rangle_{*B}} B)$ (D7, A9, A12 by JA)

Derivations D8a and D8b show that B can conclude that the key was recently generated and good for communication with A. These derivations, along with D4a and D4b, prove that the protocol provides key establishment between A and B.

D9. $B \models A \mid\!\sim (N_B, A \xleftrightarrow{\langle K_{AB} \rangle_{*B}} B)$ (I5, C4, D8a by SAA)

D10. $B \models A \ says \ (N_B, A \xleftrightarrow{|K_{AB}|_{*B}} B)$ (D9, A2 by NVA)

D10 alone is sufficient to provide liveness of A to B. It means that A has just answered B's challenge, so A must be alive and running the same protocol run as B.

3.3 Strand spaces verification of modified Yahalom

Figure 3 shows the strand spaces representation of the modified Yahalom protocol.

We specialise the term algebra A: let $\mathsf{T}_{\text{name}} \subset \mathsf{T}$ be the set of principal names.

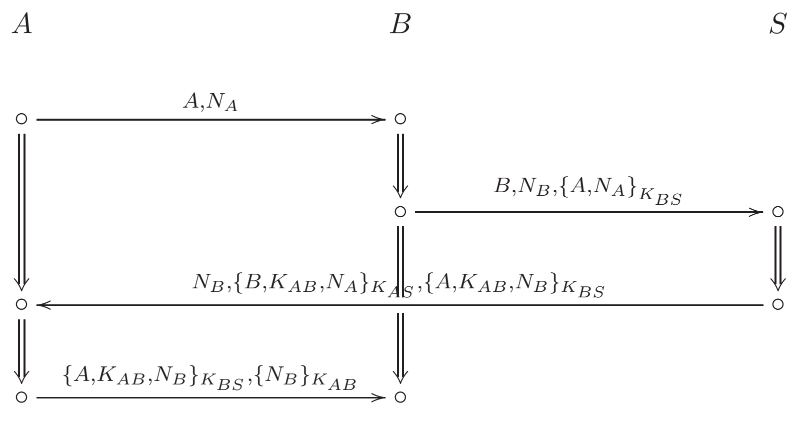

A
B
S
A, N_A
$B, N_B, \{A, N_A\}_{K_{BS}}$
$N_B, \{B, K_{AB}, N_A\}_{K_{AS}}, \{A, K_{AB}, N_B\}_{K_{BS}}$
$\{A, K_{AB}, N_B\}_{K_{BS}}, \{N_B\}_{K_{AB}}$

Figure 3: Modified Yahalom

3.3.1 Regular strands definition

There are three types of regular strands in this protocol:

- A strand $s \in \text{Init}[X, Y, N, N', K, H]$ iff s has trace of the form
$$\langle +X\ N,\ \ -N'\ \{Y\ K\ N\}_{K_{XS}}\ H,\ \ +H\ \{N'\}_K\ \rangle,$$
where $X, Y \in \mathsf{T}_{\text{name}}$, $N \notin \mathsf{T}_{\text{name}}$, $N \in \mathsf{T}$ and $K \in \mathsf{K}$. The principal associated with a strand $s \in \text{Init}[X, Y, N, N', K, H]$ is A.

- A strand $s \in \text{Resp}[X, Y, N, N', K]$ iff s has trace of the form
$$\langle -X\ N,\ \ +Y\ N'\ \{X\ N\}_{K_{YS}},\ \ -\{X\ K\ N'\}_{K_{YS}}\ \{N'\}_K\rangle,$$
where $X, Y \in \mathsf{T}_{\text{name}}$, $N' \notin \mathsf{T}_{\text{name}}$, $N' \in \mathsf{T}$ and $K \in \mathsf{K}$. The principal associated with a strand $s \in \text{Resp}[X, Y, N, N', K]$ is B.

- A strand $s \in \text{Serv}[X, Y, N, N', K]$ iff s has trace of the form
$$\langle -Y\ N'\ \{X\ N\}_{K_{YS}},\ \ +N'\ \{Y\ K\ N\}_{K_{XS}}\ \{X\ K\ N'\}_{K_{YS}}\rangle,$$
where $X, Y \in \mathsf{T}_{\text{name}}$, $N, N' \notin \mathsf{T}_{\text{name}}$, $N, N' \in \mathsf{T}$, $K \in \mathsf{K}$ and $K \notin \{K_{XS} : X \in \mathsf{T}_{\text{name}}\}$. The principal associated with a strand $s \in \text{Serv}[X, Y, N, N', K]$ is S.

The sets Serv, Resp and Init are pairwise disjoint.

3.3.2 Secrecy of K_{AB}

Theorem 3.1. *Let $\mathcal{C}$ be a bundle in the strand space Σ in which K is uniquely generated, $K_{AS}, K_{BS} \notin \mathsf{K_P}$; and $s_{serv} \in Serv[A, B, N, N', K]$. Let $S = \{K_{AS}, K_{BS}, K\}$ and $\mathsf{k} = \mathsf{K} \setminus S$. Then for every node $m \in \mathcal{C}$, $term(m) \notin I_{\mathsf{k}}[S]$.*

Proof. Assume there is a regular node $n \in \mathcal{C}$ that is an entry point for $I_{\mathsf{k}}[S]$; thus, $term(n) \in I_{\mathsf{k}}[S]$ and n is positive. By definition of $I_{\mathsf{k}}[S]$ and proposition 2.10, $K_A \sqsubset term(n)$, $K_B \sqsubset term(n)$ or $K \sqsubset term(n)$ holds. No regular node can contain a K_{XS} key as subterm, where $X \in \mathsf{T}_{\text{name}}$. Hence, $K \sqsubset term(n)$ (note that $K \notin \{K_{XS} : X \in \mathsf{T}_{\text{name}}\}$). Then, for s the strand containing n, one of the following holds:

(i) $s \in \text{Serv}, n = \langle s, 1\rangle$ and K is the session key, or

(ii) $s \in \text{Init}[*, *, *, *, K, H]$ and $n = \langle s, 2\rangle$.

Since $K \sqsubset term(\langle s, 1\rangle)$, n cannot be an entry point and therefore (ii) does not hold. K is uniquely generated, thus $s = s_{serv}$ and $term(n) = \{B\ K\ N\}_{K_{AS}}, \{A\ K\ N'\}_{K_{BS}}$. By Prop. 2.12,

(iii) either $\{B\ K\ N\}_{K_{AS}} \in I_{\mathsf{k}}[S]$,

(iv) or $\{A\ K\ N'\}_{K_{BS}} \in I_{\mathsf{k}}[S]$.

However, $K_{AS} \notin \mathsf{k}$, $K_{BS} \notin \mathsf{k}$ and every member of S is simple and not encrypted. By Prop. 2.11, neither (iii) nor (iv) hold, hence $term(n)$ is not an entry point for $I_{\mathsf{k}}[S]$. $\square$

Corollary 3.2. *Let $\mathcal{C}$ be a bundle in the strand space Σ in which K is uniquely generated; $K_{AS}, K_{BS} \notin \mathsf{K_P}$; and $s_{serv} \in Serv[A, B, N, N', K]$. Let $S = \{K_{AS}, K_{BS}, K\}$ and $\mathsf{k} = \mathsf{K} \setminus S$. Then for every node $m \in \mathcal{C}$, $term(m) \notin I_{\mathsf{k}}[K]$ and K is secret.*

3.3.3 Authentication

To verify the authentication property of modified Yahalom, two propositions are necessary:

Proposition 3.3. *Let $\mathcal{C}$ be a bundle in Σ. No term of the form $\{g\}_{K_{XS}}$, with $X \in \mathsf{T}_{name}$, S being a server with a strand $s \in Serv$, $g \in T$ and $K_{XS} \notin \mathsf{K_P}$ can be originated in a penetrator node.*

Proof. Let $U = \{K_{XS}\}$ and $\mathsf{k} = \mathsf{K}$. We can safely state that K_{XS} is not originated in any regular node in $\mathcal{C}$ (that means that no regular node in $\mathcal{C}$ is an entry point for $I_{\mathsf{k}}[U]$), since Serv, Init and Resp traces do not contain any node with K_{XS} as subterm (only the session key K appears as subterm, and by definition of Serv, K is not a long term key). So we may apply Corollary 2.15. $\square$

Proposition 3.4. *Let $\mathcal{C}$ be a bundle in Σ. K is the session key generated by the server, $g \in T$ and $K \notin \mathsf{K_P}$. Then, if K remains secret during the whole protocol run, no term of the form $\{g\}_K$ can originate in a penetrator node.*

Proof. Let us analyse all possible penetrator strands:

F, T, S, D : By Lemmas 2.1 and 2.2, none of these strands can contain a node that is an originating occurrence of a term in the form $\{g\}_K$.

C : Any term of the form $\{g\}_K$ is atomic by definition, and so it cannot be a concatenation of other terms. So, no strand of this type can contain an originating instance of a term of this form.

K : As $\{g\}_K \notin \mathsf{K_P}$, this case does not apply.

E : If $K \notin \mathsf{K_P}$ and if K remains secret during the whole protocol run, then no strand of this type can contain a node which has received the key K. So, a strand of this type can never contain an originating instance of a term in the form $\{g\}_K$, with K secret during the whole protocol run.

M : The penetrator could generate a random atomic message that would coincide with $\{g\}_K$. As this is highly unlikely, we assume it will not happen.

Thus, no node that originates a term of the form $\{g\}_K, K \in \mathsf{K_P}$ with K secret during the whole protocol execution can be in a penetrator strand. $\square$

3.3.4 Initiator's guarantee

Theorem 3.5. *Let $\mathcal{C}$ be a bundle in Σ, N_a uniquely originating in $\mathcal{C}$, $K_{AS}, K_{BS}, K_{AB} \notin \mathsf{K_P}$, and let $s \in Init[A, B, N_a, *, K_{AB}, *]$. If s has $\mathcal{C}$-height ≥ 2, then bundle $\mathcal{C}$ contains:*

- *A strand $s_{serv} \in Serv[X, Y, *, *, K_{AB}]$ with $\mathcal{C}$-height $= 2$, and*
- *a strand $s_{resp} \in Resp[X, Y, *, *, *]$ with $\mathcal{C}$-height ≥ 2, where $X, Y \in \{A, B\}$.*

Proof. If $s \in$ Init and s has $\mathcal{C}$-height ≥ 2, then there is a node $n_{s,1} \in s$ which is the second node in strand s and $term(n_{s,1}) = -N_b \ \{B\ K_{AB}\ N_a\}_{K_{AS}}\ H$. Clearly, $\{B\ K_{AB}\ N_a\}_{K_{AS}} \sqsubset term(n_{s,1})$. By Lemmas 2.1 and 2.2, we know that there is a node $m \in \mathcal{C}$ with $term(m) = +N_b\ \{B\ K_{AB}\ N_a\}_{K_{AS}}\ H$ which generated $N_b\ \{B\ K_{AB}\ N_a\}_{K_{AS}}\ H$ (m is a minimal member with respect to $N_b\ \{B\ K_{AB}\ N_a\}_{K_{AS}}\ H$).

According to Prop. 3.3, $\{B\ K_{AB}\ N_a\}_{K_{AS}}$ was originated in a regular node in $\mathcal{C}$. By the structure[1] of $\{B\ K_{AB}\ N_a\}_{K_{AS}}$, it can only originate in a node $n_{serv,1}$ of a strand $s_{serv} \in$ Serv. There are two possibilities for the nature of $\{B\ K_{AB}\ N_a\}_{K_{AS}}$:

- $\{B\ K_{AB}\ N_a\}_{K_{AS}}$ is the first encrypted term in $n_{serv,1}$, $s_{serv} \in$ Serv$[A, B, N_a, *, K_{AB}]$ and s_{serv} has $\mathcal{C}$-height $= 2$; or
- $\{B\ K_{AB}\ N_a\}_{K_{AS}}$ is the last encrypted term in $n_{serv,1}$, $s_{serv} \in Serv[B, A, *, N_a, K_{AB}]$ and s_{serv} has $\mathcal{C}$-height $= 2$.

If the first hypothesis is true, then there is a node $n_{serv,0} \in s_{serv}$ in the bundle that has $\{A\ N_a\}_{K_{BS}}$ as its subterm. By Prop. 3.3, this term originates in a regular node which, by the structure of $\{A\ N_a\}_{K_{BS}}$, belongs to a strand $s_1 \in$ Resp$[A, B, N_a, *, *]$.

[1]The structure of a term is related essentially to the following two factors: its width, which is defined by the number of atomic subterms of that term; and the domain of its fields, which restrains the nature of each of these subterms.

If the second hypothesis is true, then there is a node $n_{serv,0} \in s_{serv}$ in the bundle which has $\{B\ *\}_{K_{AS}}$ as its subterm. By proposition 3.3, this term originates in a regular node which, by the structure of $\{B\ *\}_{K_{AS}}$, belongs to a strand $s_2 \in \text{Resp}[B, A, *, *, *]$.

None of these two hypothesis is a contradiction, and so both are possible. The only guarantees we can obtain for the Initiator regarding the Responder and Server are:

- There is a strand $s_{serv} \in \text{Serv}[X, Y, *, *, K_{AB}]$ in $\mathcal{C}$ with $\mathcal{C}$-height $= 2$, and
- there is a strand $s_{resp} \in \text{Resp}[X, Y, *, *, *]$ in $\mathcal{C}$ with $\mathcal{C}$-height ≥ 2.

□

Although the Initiator concludes that a Responder in fact exists in the bundle, there is no indication that this Responder is running the same protocol run as the Initiator, neither that they are peer entities. Actually, this Responder can be the Initiator itself playing a responder part in a parallel protocol run. This is exactly the weakness shown by the Syverson attack on modified Yahalom.

As for the Server, the Initiator can conclude that it in fact exists. However, that Server does not "know" whether A is the Initiator or the Responder in this protocol run. In the last case, the Server is not involved in the same protocol run in which A is Initiator (we refer to this run as the original one). Note that in this last case, A is the protocol Responder, and S also sees it that way. So, both A and S are running a parallel protocol run besides the original one. This is exactly the attack observed by Syverson.

The only valid guarantee that the Initiator can achieve is that the session key K_{AB} remains secret and is generated by the Server S, which completes the original protocol run. Besides that execution, if we consider the second hypothesis (the one which characterises the Syverson attack), the Server also completes at least one parallel protocol run. This fact can also be seen in the mentioned attack.

3.3.5 Responder's guarantee

Let $\mathcal{C}$ be a bundle in Σ, N_b uniquely originating in $\mathcal{C}$ and $K_{AS}, K_{BS}, K_{AB} \notin \mathsf{K_P}$, and $s \in \text{Resp}[A, B, N_a, N_b, K_{AB}]$.

Theorem 3.6. *If s has $\mathcal{C}$-height $= 3$, then there is a strand $s_{serv} \in Serv[A, B, N_a, N_b, K_{AB}]$ in bundle $\mathcal{C}$ with $\mathcal{C}$-height $= 2$.*

Proof. If $s \in$ Resp and s has $\mathcal{C}$-height $= 3$, then there is a node $n_{s,2} \in s$ which is the last node in the strand s and $\{A\ K_{AB}\ N_b\}_{K_{BS}} \sqsubset term(n_{s,2})$. By Lemmas 2.1 and 2.2, we know that there is a node $m \in \mathcal{C}$ with $term(m) = +uns\_term(n_{s,2})$.

According to Prop. 3.3, $\{A\ K_{AB}\ N_b\}_{K_{BS}}$ originates in a regular node in $\mathcal{C}$. By the structure of $\{A\ K_{AB}\ N_b\}_{K_{BS}}$, it can only originate in a node $n_{init,2}$ (third node of an Initiator strand) or in a node $n_{serv,1}$ (second node in a Server strand). By precedence, the uniquely originating node of $\{A\ K_{AB}\ N_b\}_{K_{BS}}$ is the last one. By the definition of bundle, if there is a node $n_{serv,1}$ in $\mathcal{C}$, then there is also a node $n_{serv,0}$ in $\mathcal{C}$. So, there is a strand $s_{serv} \in Serv$ and s_{serv} has $\mathcal{C}$-height $= 2$.

Now we must decide over which values the Responder and the Server agree. Note that $\{A\ K_{AB}\ N_b\}_{K_{BS}}$ can be both the first or the last encrypted component of $term(n_{serv,1})$.

- $\{A\ K_{AB}\ N_b\}_{K_{BS}}$ is the first encrypted component of $n_{serv,1}$, $s_{serv} \in Serv[B, A, N_b, *, K_{AB}]$, or
- $\{A\ K_{AB}\ N_b\}_{K_{BS}}$ is the last encrypted component of $n_{serv,1}$, $s_{serv} \in Serv[A, B, *, N_b, K_{AB}]$.

If the first hypothesis is true, then there is a node $n_{serv,0} \in s_{serv}$ in the bundle which has $\{B\ N_b\}_{K_{AS}}$ as its subterm. By Prop. 3.3, this term uniquely originates in a regular node which, by the structure of $\{B\ N_b\}_{K_{AS}}$, belongs to a strand $s_1 \in Resp[B, A, N_b, *, *]$. In that case, N_b is uniquely originating in s_1. But N_b is uniquely originating in s_{serv} in the bundle, and so $s_1 = s_{serv}$ and $B = A$. However, we assume B and A as being distinct principals, a contradiction.

If the second hypothesis is true, then there is a node $n_{serv,0} \in s_{serv}$ in the bundle which has $\{A\ *\}_{K_{BS}}$ as its subterm. By Prop. 3.3, this term is uniquely originating in a regular node which, by the structure of $\{A\ *\}_{K_{BS}}$, belongs to a strand $s_2 \in \text{Resp}[A, B, *, *, *]$. So, we conclude that $s_{serv} \in \text{Serv}[A, B, *, N_b, K_{AB}]$. □

This proof shows that not only the Responder has the guarantee that there is in fact a valid Server, but also that this Server agrees with it about the values of A, B, N_b and K_{AB}. This agreement provides entity authentication to the Responder regarding the Server, and guarantees that the session key received by the Responder is the same one generated by the Server. The Responder can also conclude that the Server executed a complete protocol run.

Theorem 3.7. *If s has $\mathcal{C}$-height $= 3$, then there is a strand $s_{init} \in Init[A, B, N_a, N_b, K_{AB}, H]$ in the bundle $\mathcal{C}$ with $\mathcal{C}$-height $= 3$, where $H = \{A\ K_{AB}\ N_b\}_{K_{BS}}$.*

Proof. If $s \in Resp$ and s has $\mathcal{C}$-height $= 3$, then there is a node $n_{s,2} \in s$ which is the last node in the strand s and $term(n_{s,2}) = -\{A\ K_{AB}\ N_b\}_{K_{BS}}\ \{N_b\}_{K_{AB}}$. By Lemmas 2.1 and 2.2, we know that there is a node $m \in \mathcal{C}$ with $term(m) = +\{A\ K_{AB}\ N_b\}_{K_{BS}}\ \{N_b\}_{K_{AB}}$ which generated this message (m is a minimal member regarding to $+\{A\ K_{AB}\ N_b\}_{K_{BS}}\ \{N_b\}_{K_{AB}}$).

According to Prop. 3.4, $\{N_b\}_{K_{AB}}$ was originated in a regular node in $\mathcal{C}$. By the structure of $+\{N_b\}_{K_{AB}}$, it can only be originated in an initiator strand, in its third node $n_{init,2}$. So, $\{N_b\}_{K_{AB}}$ is uniquely originating in $\mathcal{C}$ by node $n_{init,2}$, what leads to the conclusion that $m = n_{init,2}$. Thus there is a strand $s_{init} \in \text{Init}[A, *, *, N_b, K_{AB}, H']$ in $\mathcal{C}$ with $m \in s_{init}$. By the definition of bundle, if there is a node $n_{init,2}$ in $\mathcal{C}$, then there are also nodes $n_{init,0}$ and $n_{init,1}$ in $\mathcal{C}$. Thus, s_{init} has $\mathcal{C}$-height $= 3$.

It's then possible to derive the last agreements between Responder and Initiator from the regularity of the Initiator. For the second Initiator node to be consistent to the trace of this strand type, it is necessary that the first component of it's term is equal to N_b and that the second has the form $\{B\ K_{AB}\ *\}_{K_{AS}}$. Furthermore, $H' = \{A\ K_{AB}\ N_b\}_{K_{BS}} = H$. That way we can conclude that $s_{init} \in \text{Init}[A, B, *, N_b, K_{AB}, H]$.

We then have that $s_{init} \in \text{Init}[A, B, *, N_b, K_{AB}, H]$ and s_{init} has $\mathcal{C}$-height $= 2$. □

This demonstration shows that not only the Responder has the guarantee that there is in fact a valid Initiator, but also that this Initiator agrees with him about the values of A, B, N_b, K_{AB} and H. This agreement grants entity authentication to the Responder regarding the Initiator, and key establishment. Finally, as the Responder's challenge N_b can be used as time identification and was correctly responded by the Initiator, the first one has assurance of the last one.

We conclude that modified Yahalom fails to provide the guarantees it purportedly offers to the Initiator, but it succeeds in the case of the Responder.

4 The Kerberos protocol with an extra message

The Kerberos protocol [9] is a server-based key establishment protocol that provides entity authentication of the principal that started the protocol (the initiator) to the other principal (the responder), but not the converse. It is possible to modify Kerberos to provide mutual authentication by adding an extra message, as shown in Figure 4. Throughout this text, we shall refer to this version as *modified Kerberos*. There are no published attacks to neither version of the Kerberos protocol.

1. $A \rightarrow S : A, B, N_A$
2. $S \rightarrow A : \{K_{AB}, B, L, N_A, \ldots\}_{K_{AS}}, \{K_{AB}, A, L, \ldots\}_{K_{BS}}$
3. $A \rightarrow B : \{A, T_A\}_{K_{BS}}, \{K_{AB}, A, L, \ldots\}_{K_{BS}}$
4. $B \rightarrow A : \{T_A, \ldots\}_{K_{AB}}$

Figure 4: Modified Kerberos: the Kerberos protocol with an extra message to fulfill mutual authentication

Two main steps compose a modified Kerberos run. In the first (messages one 1 and 2) the initiator contacts the server and requests a new session with the responder. The server then generates a new session key for that session and delivers it to the initiator. In the second step (messages 3 and 4) the initiator contacts the responder and sends him the recently generated by the server key. Note that the responder never interacts directly with the server; the encrypted message which contains the session key is delivered to the responder through the initiator – although the initiator is unable to read its content. The last message is intended to authenticate the responder to the initiator.

In this section we prove that mutual authentication is in fact achieved by modified Kerberos, and that the session key is never compromised by the protocol.

4.1 BAN verification of modified Kerberos

4.1.1 Idealisation

1. $A \rightarrow S :$

2. $S \rightarrow A : \left\{A \xleftrightarrow{K_{AB}} B, L, \sharp(L), N_a\right\}_{K_{AS}}, \left\{A \xleftrightarrow{K_{AB}} B, L, \sharp(L)\right\}_{K_{BS}}$
3. $A \rightarrow B : \left\{T_A, \sharp(T_A), A \xleftrightarrow{K_{AB}} B\right\}_{K_{AB}}, \left\{A \xleftrightarrow{K_{AB}} B, L, \sharp(L)\right\}_{K_{BS}}$
4. $B \rightarrow A : \left\{T_A, A \xleftrightarrow{K_{AB}} B\right\}_{K_{AB}}$

4.1.2 Assumptions

A1. $A \mid\equiv \sharp(N_A)$

A2. $B \mid\equiv \sharp(N_B)$

A3. $A \mid\equiv A \xleftrightarrow{K_{AS}} S$

A4. $B \mid\equiv B \xleftrightarrow{K_{BS}} S$

A5. $S \mid\equiv A \xleftrightarrow{K_{AS}} S$

A6. $S \mid\equiv B \xleftrightarrow{K_{BS}} S$

A7. $S \mid\equiv S \Rightarrow A \xleftrightarrow{K_{AB}} B$

A8. $A \mid\equiv S \Rightarrow A \xleftrightarrow{K_{AB}} B$

A9. $B \mid\equiv S \Rightarrow A \xleftrightarrow{K_{AB}} B$

A10. $S \mid\equiv S \Rightarrow \sharp(A \xleftrightarrow{K_{AB}} B)$

A11. $A \mid\equiv S \Rightarrow \sharp(A \xleftrightarrow{K_{AB}} B)$

A12. $B \mid\equiv S \Rightarrow \sharp(A \xleftrightarrow{K_{AB}} B)$

A13. $A \mid\equiv \sharp(T_A)$

A14. $B \mid\equiv A \Rightarrow T_A$

A15. $B \mid\equiv A \Rightarrow \sharp(T_A)$

A16. $S \mid\equiv A \Rightarrow T_A$

A17. $S \mid\equiv A \Rightarrow \sharp(T_A)$

A18. $S \mid\equiv \sharp(L)$

A19. $B \mid\equiv S \Rightarrow L$

A20. $B \mid\equiv S \Rightarrow \sharp(L)$

A21. $A \models S \Rrightarrow L$

A22. $A \models S \Rrightarrow \sharp(L)$

A23. $B \models \sharp(L)^*$

4.1.3 Analysis

D1. $$\frac{A \triangleleft \left\{A \overset{K_{AB}}{\longleftrightarrow} B, L, \sharp(L), N_A\right\}_{K_{AS}}, A \models A \overset{K_{AS}}{\longleftrightarrow} S}{A \models S \mid\sim (A \overset{K_{AB}}{\longleftrightarrow} B, L, \sharp(L), N_A)} \text{ (M2, A3, by MM)}$$

D2. $$\frac{A \models S \mid\sim (A \overset{K_{AB}}{\longleftrightarrow} B, L, \sharp(L), N_A), A \models \sharp(A \overset{K_{AB}}{\longleftrightarrow} B, L, \sharp(L), N_A)}{A \models S \models (A \overset{K_{AB}}{\longleftrightarrow} B, L, \sharp(L), N_A)} \text{ (D1, A1, by NV)}$$

D3. $$\frac{A \models S \models A \overset{K_{AB}}{\longleftrightarrow} B, A \models S \Rrightarrow A \overset{K_{AB}}{\longleftrightarrow} B}{A \models A \overset{K_{AB}}{\longleftrightarrow} B} \text{ (D2, A8, by J)}$$

D4. $$\frac{A \models S \models \sharp(A \overset{K_{AB}}{\longleftrightarrow} B), A \models S \Rrightarrow \sharp(A \overset{K_{AB}}{\longleftrightarrow} B)}{A \models \sharp(A \overset{K_{AB}}{\longleftrightarrow} B)} \text{ (D2, A11, by J)}$$

From the sentences above, we may conclude that the protocol achieves key establishment for entity A. A believes that not only the key is good for communication with B, but also that it was recently generated.

D5. $$\frac{\left\{T_A, A \overset{K_{AB}}{\longleftrightarrow} B\right\}_{K_{AB}}, A \models A \overset{K_{AB}}{\longleftrightarrow} B}{A \models B \mid\sim (T_A, A \overset{K_{AB}}{\longleftrightarrow} B)} \text{ (M4, D4, by MM)}$$

D6. $$\frac{A \models B \mid\sim (T_A, A \overset{K_{AB}}{\longleftrightarrow} B), A \models \sharp(T_A, A \overset{K_{AB}}{\longleftrightarrow} B)}{A \models B \models (T_A, A \overset{K_{AB}}{\longleftrightarrow} B)} \text{ (D5, A13, by NV)}$$

The sentence above shows that at the end of the protocol run, A is able to recognise B as its peer entity. Furthermore, A gets the assurance that B believes in the goodness of the session key.

D7. $$\frac{A \models S \models L, A \models S \Rrightarrow L}{A \models L} \text{ (D2, A19, by J)}$$

D8. $$\frac{A \models S \models \sharp(L), A \models S \Rrightarrow \sharp(L)}{A \models \sharp(L)} \text{ (D2, A18, A2, by J)}$$

These sentences show that the protocol provides liveness of S to A at the end of a protocol run.

D9. $\dfrac{B \triangleleft \left\{A \xleftrightarrow{K_{AB}} B, L, \sharp(L)\right\}_{K_{BS}}, B \models B \xleftrightarrow{K_{BS}} S}{B \models S \mid\sim (A \xleftrightarrow{K_{AB}} B, L, \sharp(L))}$ (M4, A4, by MM)

It is impossible to go any further without Assumption 23. We then use that assumption to derive the belief from S in the statement:

D10. $\dfrac{B \models S \mid\sim (A \xleftrightarrow{K_{AB}} B, L, \sharp(L)), B \models \sharp(A \xleftrightarrow{K_{AB}} B, L, \sharp(L))}{B \models S \models (A \xleftrightarrow{K_{AB}} B, L, \sharp(L))}$ (D9, A23, by NV)

D11. $\dfrac{B \models S \models A \xleftrightarrow{K_{AB}} B, B \models S \Rightarrow A \xleftrightarrow{K_{AB}} B}{B \models A \xleftrightarrow{K_{AB}} B}$ (D10, A9, by J)

D12. $\dfrac{B \models S \models \sharp(A \xleftrightarrow{K_{AB}} B), B \models S \Rightarrow \sharp(A \xleftrightarrow{K_{AB}} B)}{B \models \sharp(A \xleftrightarrow{K_{AB}} B)}$ (D10, A12, by J)

From the above sentences, we may conclude that the protocol achieves key establishment for entity B. B believes that not only the key is good for communication with A, but also that it was recently generated.

D13. $\dfrac{B \models S \models L, B \models S \Rightarrow L}{B \models L}$ (D10, A19, by J)

This sentence shows that the protocol provides liveness of S to A at the end of a protocol run.

D14. $\dfrac{B \triangleleft (T_A, \sharp(T_A), \left\{A \xleftrightarrow{K_{AB}} B\right\}_{K_{BS}}), B \models A \xleftrightarrow{K_{AB}} B}{B \models A \mid\sim (T_A, \sharp(T_A), A \xleftrightarrow{K_{AB}} B)}$ (M3, D11, by MM)

D15. $\dfrac{B \models A \mid\sim (T_A, \sharp(T_A), A \xleftrightarrow{K_{AB}} B), B \models \sharp(T_A, \sharp(T_A), A \xleftrightarrow{K_{AB}} B)}{B \models A \models (T_A, \sharp(T_A), A \xleftrightarrow{K_{AB}} B)}$ (D14, D12, by NV)

D16. $\dfrac{B \models A \models (T_A, \sharp(T_A)), B \models A \Rightarrow (T_A, \sharp(T_A))}{B \models (T_A, \sharp(T_A))}$ (D15, A14, A15, by J)

The sentence above shows that at the end of the protocol run, B is able to recognise A as his peer entity. Furthermore, B gets the assurance that A believes in the goodness of the session key.

4.2 SVO verification of modified Kerberos

4.2.1 Assumptions

A1. $A \models \sharp(N_A)$

A2. $B \mid\equiv \sharp(N_B)$

A3. $A \mid\equiv A \xleftrightarrow{K_{AS}} S$

A4. $B \mid\equiv B \xleftrightarrow{K_{BS}} S$

A5. $S \mid\equiv A \xleftrightarrow{K_{AS}} S$

A6. $S \mid\equiv B \xleftrightarrow{K_{BS}} S$

A7. $S \mid\equiv S \Mapsto A \xleftrightarrow{K_{AB}} B$

A8. $A \mid\equiv S \Mapsto A \xleftrightarrow{K_{AB}} B$

A9. $B \mid\equiv S \Mapsto A \xleftrightarrow{K_{AB}} B$

A10. $S \mid\equiv S \Mapsto \sharp(A \xleftrightarrow{K_{AB}} B)$

A11. $A \mid\equiv S \Mapsto \sharp(A \xleftrightarrow{K_{AB}} B)$

A12. $B \mid\equiv S \Mapsto \sharp(A \xleftrightarrow{K_{AB}} B)$

A13. $A \mid\equiv S \Mapsto L$

A14. $A \mid\equiv S \Mapsto \sharp(L)$

A15. $B \mid\equiv S \Mapsto L$

A16. $B \mid\equiv S \Mapsto \sharp(L)$

A17. $B \mid\equiv S \Mapsto T_A$

A18. $B \mid\equiv S \Mapsto \sharp(T_A)$

A19. $A \mid\equiv \sharp(T_A)$

A20. $* * * B \mid\equiv \sharp(L)$

4.2.2 Annotation

N1. $S \triangleleft (A, B, N_A)$

N2. $A \triangleleft \{K_{AB}, B, L, N_a\}_{K_{AS}}, \{K_{AB}, A, L\}_{K_{AS}}$

N3. $B \triangleleft \{A, T_A\}_{K_{AB}}, \{A, K_{AB}, N_B\}_{K_{BS}}$

N4. $A \triangleleft \{T_A\}_{K_{AB}}$

4.2.3 Comprehension

C1. $S \models S \triangleleft (A, B, \langle N_A \rangle_{*S})$

C2. $A \models A \triangleleft \{\langle K_{AB} \rangle_{*A}, B, \langle L \rangle_{*A}, N_a\}_{K_{AS}}, \langle \{K_{AB}, A, L\}_{K_{AS}} \rangle_{*A}$

C3. $B \models B \triangleleft \langle \{A, T_A\}_{K_{AB}} \rangle_{*B}, \{\langle K_{AB} \rangle_{*B}, A, \langle L \rangle_{*B}\}_{K_{BS}}$

C4. $A \models A \triangleleft \langle \{T_A\}_{K_{AB}} \rangle_{*A}$

4.2.4 Interpretation

I1.

I2. $A \models A \triangleleft \{\langle K_{AB} \rangle_{*A}, B, \langle L \rangle_{*A}, N_a\}_{K_{AS}}, \langle \{K_{AB}, A, L\}_{K_{AS}} \rangle_{*A} \rightarrow$
$A \models A \triangleleft \left\{A \xleftrightarrow{\langle K_{AB} \rangle_{*A}} B, \sharp(A \xleftrightarrow{\langle K_{AB} \rangle_{*A}} B), B, \langle L \rangle_{*A}, \sharp(\langle L \rangle_{*A}), N_A\right\}_{K_{AS}},$
$\langle \left\{A \xleftrightarrow{K_{AB}} B, \sharp(A \xleftrightarrow{K_{AB}} B), A, L, \sharp(L)\right\}_{K_{BS}} \rangle_{*A}$

I3a. $B \models B \triangleleft \langle \{A, T_A\}_{K_{AB}} \rangle_{*B}, \{\langle K_{AB} \rangle_{*B}, A, \langle L \rangle_{*B}\}_{K_{BS}} \rightarrow$
$B \models B \triangleleft \left\{A \xleftrightarrow{\langle K_{AB} \rangle_{*B}} B, \sharp(A \xleftrightarrow{\langle K_{AB} \rangle_{*B}} B), A, \langle L \rangle_{*B}, \sharp(\langle L \rangle_{*B}),\right\}_{K_{BS}}$

I3b. $B \models B \triangleleft \langle \{A, T_A\}_{K_{AB}} \rangle_{*B}, \{\langle K_{AB} \rangle_{*B}, A, \langle L \rangle_{*B}\}_{K_{BS}} \wedge B \models A \xleftrightarrow{\langle K_{AB} \rangle_{*B}} B \rightarrow B \models B \triangleleft \left\{A, \langle T_A \rangle_{*B}, A \xleftrightarrow{\langle K_{AB} \rangle_{*B}} B\right\}_{\langle K_{AB} \rangle_{*B}}$

I4. $A \models A \triangleleft \langle \{T_A\}_{K_{AB}} \rangle_{*A} \wedge A \models A \xleftrightarrow{\langle K_{AB} \rangle_{*A}} B \rightarrow A \models A \triangleleft \{T_A\}_{\langle K_{AB} \rangle_{*A}}$

4.2.5 Derivation

D1. $A \models A \triangleleft \left\{A \xleftrightarrow{\langle K_{AB} \rangle_{*A}} B, \sharp(A \xleftrightarrow{\langle K_{AB} \rangle_{*A}} B), B, \langle L \rangle_{*A}, \sharp(\langle L \rangle_{*A}), N_A\right\}_{K_{AS}},$
$\langle \left\{A \xleftrightarrow{K_{AB}} B, \sharp(A \xleftrightarrow{K_{AB}} B), A, L, \sharp(L)\right\}_{K_{BS}} \rangle_{*A}$ (I2, C2, by MP)

D2a. $A \models S \mid\!\sim (A \xleftrightarrow{\langle K_{AB} \rangle_{*A}} B, \sharp(A \xleftrightarrow{\langle K_{AB} \rangle_{*A}} B), B, \langle L \rangle_{*A}, \sharp(\langle L \rangle_{*A}), N_A)$ (D1, A3, by SA)

D2b. $A \models S$ *has* $(A \xleftrightarrow{\langle K_{AB} \rangle_{*A}} B, \sharp(A \xleftrightarrow{\langle K_{AB} \rangle_{*A}} B), B, \langle L \rangle_{*A}, \sharp(\langle L \rangle_{*A}), N_A)$ (D1, A3, by SAA)

D3. $A \models S$ *says* $(A \xleftrightarrow{\langle K_{AB} \rangle_{*A}} B, \sharp(A \xleftrightarrow{\langle K_{AB} \rangle_{*A}} B), B, \langle L \rangle_{*A}, \sharp(\langle L \rangle_{*A}), N_A)$ (D2a, A1, by FA, NVA)

D4a. $A \models A \xleftrightarrow{\langle K_{AB} \rangle_{*A}} B$, (D3, A8, A11, by SA, JA)

D4b. $A \models \sharp(A \xleftrightarrow{\langle K_{AB} \rangle_{*A}} B)$, (D3, A8, A11, by SA, JA)

Derivations D4a and D4b show that A can conclude that the key was recently generated and good for communication with B.

D5. $A \mid\equiv A \triangleleft \{T_A\}_{\langle K_{AB}\rangle_{*A}}$ (I4, D4a, by MP)

D6. $A \mid\equiv B \mid\sim (T_A)$ (D5, D4a, by SAA)

D7. $A \mid\equiv B$ *says* (T_A) (D6, A10, by NVA)

D7 alone is sufficient to provide to A liveness of B. It means that B has just answered A's challenge, so B must be alive and running the same protocol run as A.

D8a. $A \mid\equiv \langle L\rangle_{*A}$ (D3, A13, by JA)

D8b. $A \mid\equiv \sharp(\langle L\rangle_{*A})$ (D3, A13, by JA)

From the derivation above, A may conclude that server S is alive, and that ticket L is fresh. This prevents A from being forced to accept an old session key.

D9. $B \mid\equiv B \triangleleft \left\{A \xleftrightarrow{\langle K_{AB}\rangle_{*B}} B, \sharp(A \xleftrightarrow{\langle K_{AB}\rangle_{*B}} B), A, \langle L\rangle_{*B}, \sharp(\langle L\rangle_{*B})\right\}_{K_{BS}}$ (I3a, C3, by MP)

D10. $B \mid\equiv B \mid\sim (A \xleftrightarrow{\langle K_{AB}\rangle_{*B}} B, \sharp(A \xleftrightarrow{\langle K_{AB}\rangle_{*B}} B), A, \langle L\rangle_{*B}, \sharp(\langle L\rangle_{*B}))$ (D9, A4, by SAA)

Before we can go any further, we need to conclude that B believes that S says $(A \xleftrightarrow{\langle K_{AB}\rangle_{*B}} B, \sharp(A \xleftrightarrow{\langle K_{AB}\rangle_{*B}} B), A, \langle L\rangle_{*B}, \sharp(\langle L\rangle_{*B}))$. This is only possible if we give B the assurance that one of the subterms from this message is recent. The most reasonable possibility is the ticket L, so we create Assumption A20.

D11. $B \mid\equiv B$ *says* $(A \xleftrightarrow{\langle K_{AB}\rangle_{*B}} B, \sharp(A \xleftrightarrow{\langle K_{AB}\rangle_{*B}} B), A, \langle L\rangle_{*B}, \sharp(\langle L\rangle_{*B}))$ (D10, A20, by NVA)

D12a. $B \mid\equiv A \xleftrightarrow{\langle K_{AB}\rangle_{*B}} B$ (D11, A9, A12, by JA)

D12b. $B \mid\equiv \sharp(A \xleftrightarrow{\langle K_{AB}\rangle_{*B}} B)$ (D11, A9, A12, by JA)

Derivations D12a and D12b show that B can conclude that the key was recently generated and good for communication with A.

- $B \mid\equiv L$ (D11, A15, by JA)

This derivation is merely illustrative, as Assumption A20 already states that B believes that every ticket it receives is fresh.

D14. $B \mid\equiv A \mid\sim (A, \langle T_A\rangle_{*B}, A \xleftrightarrow{K_{AB}} B)$ (I3b, C3, D12a by MP, SAA)

D15. $B \mid\equiv A$ *says* $(A, \langle T_A\rangle_{*B}, A \xleftrightarrow{K_{AB}} B)$ (D14, D12b, by NVA)

D16a. $B \mid\equiv \langle T_A\rangle_{*B}$ (D15, A17, A18, by JA)

D16b; $B \mid\equiv \sharp(\langle T_A\rangle_{*B})$ (D15, A17, A18, by JA)

Finally, D16a and D16b show that B can conclude that A is alive and has been his peer entity on this protocol run.

4.3 Strand space verification of modified Kerberos

4.3.1 Strand space representation

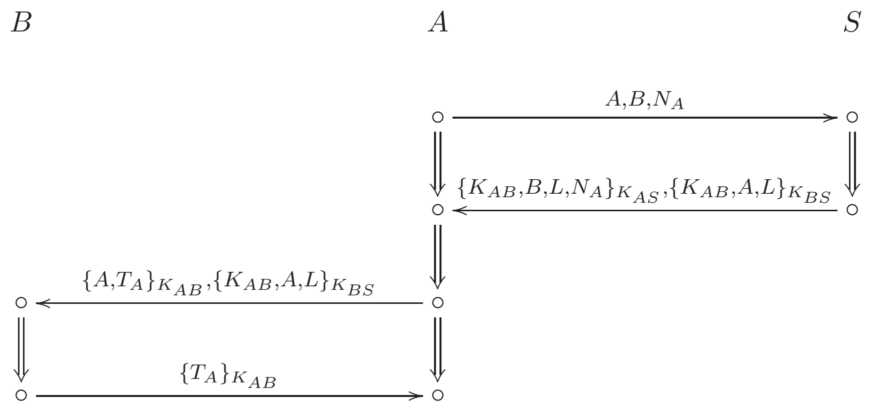

B
A
S
A,B,N_A
$\{K_{AB},B,L,N_A\}_{K_{AS}},\{K_{AB},A,L\}_{K_{BS}}$
$\{A,T_A\}_{K_{AB}},\{K_{AB},A,L\}_{K_{BS}}$
$\{T_A\}_{K_{AB}}$

Figure 5: Modified Kerberos

4.3.2 Regular strands definition

1. A strand $s \in \text{Init}[X, Y, N, T, L, K, H]$ iff s has trace of the form

$$\langle +X\ Y\ N,\ \ -\{K\ Y\ L\ N\}_{K_{XS}}H,\ \ +\{X\ T\}_K\ H,\ \ -\{T\}_K\rangle,$$

   where $X, Y \in \mathsf{T}_{\text{name}}$, $N \notin \mathsf{T}_{\text{name}}$ and $K \in \mathsf{K}$. The principal associated with a strand $s \in \text{Init}[X, Y, N, T, L, K, H]$ is A.

2. A strand $s \in \text{Resp}[X, Y, T, L, K]$ iff s has trace of the form

$$\langle -\{X\ T\}_K\ \{K\ X\ L\}_{K_{YS}},\ \ +\{T\}_K\rangle,$$

   where $X, Y \in \mathsf{T}_{\text{name}}$ and $K \in \mathsf{K}$. The principal associated with a strand $s \in Resp[X, Y, T, L, K]$ is B.

3. A strand $s \in \text{Serv}[X, Y, N, L, K]$ iff s has trace of the form

$$\langle -X\ Y\ N,\ \ +\{K\ Y\ L\ N\}_{K_{XS}}\{K\ X\ L\}_{K_{YS}}\rangle,$$

   where $X, Y \in \mathsf{T}_{\text{name}}$, $N \notin \mathsf{T}_{\text{name}}$, $K \in \mathsf{K}$ and $K \notin \{K_{XS} : X \in \mathsf{T}_{\text{name}}\}$. The principal associated with a strand $s \in textServ[X, Y, N, L, K]$ is S.

The sets Serv, Resp and Init are pairwise disjoint.

4.3.3 Secrecy of K_{AB}

Theorem 4.1. *Suppose $\mathcal{C}$ is a bundle in strand space Σ where K is uniquely generated, $K_{AS}, K_{BS} \notin \mathsf{K_P}$, and $s_{serv} \in Serv[A, B, N, L, K]$. Let $S = \{K_{AS},\ K_{BS},\ K\}$ and $\mathsf{k} = \mathsf{K} \setminus S$. Then for every node $m \in \mathcal{C}$, $term(m) \notin I_{\mathsf{k}}[K]$.*

Proof. This proof can be obtained through the following steps: Using Prop. 2.10, we can prove this theorem by proving the stronger statement $\forall m \in \mathcal{C}, term(m) \notin I_{\mathsf{k}}[S]$. By Corollary 2.14, if there is a node m in the bundle with $term(m) \in I_{\mathsf{k}}[S]$, then there is a regular node n in the bundle that is an entry point for $I_{\mathsf{k}}[S]$.

$$(\exists m \in \mathcal{C}, term(m) \in I_{\mathsf{k}}[S]) \Rightarrow (\exists n \in \mathcal{C}, n \text{ regular} \wedge n \text{ is an entry point for } I_{\mathsf{k}}[S]).$$

By the contrapositive of the above implication, we obtain:

$$(\forall n \in \mathcal{C}, n \text{ is not regular} \vee n \text{ is not an entry point for } I_{\mathsf{k}}[S]) \Rightarrow (\forall m \in \mathcal{C}, term(m) \notin I_{\mathsf{k}}[S]).$$

So, by proving that there is no regular node in $\mathcal{C}$ which is an entry point for $I_{\mathsf{k}}[S]$, we manage to show that for every node $m \in \mathcal{C}$, $term(m) \notin I_{\mathsf{k}}[S]$ and that, by Prop. 2.10, $term(m) \notin I_{\mathsf{k}}[K]$ for every node $m \in \mathcal{C}$.

Assume there is a regular node $n \in \mathcal{C}$ that is an entry point for $I_{\mathsf{k}}[S]$. By the definition of entry point, $term(n) \in I_{\mathsf{k}}[S]$. By the definition of $I_{\mathsf{k}}[S]$ and Prop. 2.10, we have that $K_A \sqsubset term(n)$, $K_B \sqsubset term(n)$ or $K \sqsubset term(n)$. No regular node can contain a K_{XS} key as subterm, with $X \in \mathsf{T}_{\text{name}}$. So, $K \sqsubset term(n)$ (note that $K \notin \{K_{XS} : X \in \mathsf{T}_{\text{name}}\}$). Let s be the strand containing n. By the definition of entry point, n is positive. We have the following possible situations:

- $s \in \text{Serv}, n = \langle s, 1\rangle$ and K is the session key; or
- $s \in \text{Init}[*, *, *, *, *, K, H]$ and $n = \langle s, 2\rangle$. But $K \sqsubset term(\langle s, 1\rangle)$, then n can't be an entry point; or
- $s \in \text{Resp},\ n = \langle s, 1\rangle$ and $K \sqsubset T$. But $T \sqsubset term(\langle s, 0\rangle)$, so n cannot be an entry point; or
- $s \in \text{Init}$ and $n = \langle s, 0\rangle$ and $K \sqsubset N$. As $K \not\sqsubset N$, we can disregard this hypothesis.

Thus, consider the first hypothesis. Since K is uniquely generated, we have $s = s_{serv}$; then $term(n) = \{K\ B\ L\ N\}_{K_{AS}}, \{K\ A\ L\}_{K_{BS}}$. By Prop. 2.12,:

- $\{K\ B\ L\ N\}_{K_{AS}} \in I_{\mathsf{k}}[S]$, or
- $\{K\ A\ L\}_{K_{BS}} \in I_{\mathsf{k}}[S]$.

But $K_{AS} \notin \mathsf{k}$ and $K_{BS} \notin \mathsf{k}$, and every member of S is simple and not encrypted. Then, by Prop. 2.11, both hypotheses are false, a contradiction.

We can then state that $term(n)$ is not an entry point for $I_{\mathsf{k}}[S]$, and so we conclude that K is secret. □

4.3.4 Authentication

To verify the authentication property of modified Kerberos, two propositions are necessary:

Proposition 4.2. *Let $\mathcal{C}$ be a bundle in Σ. No term of the form $\{g\}_{K_{XS}}$, with $X \in \mathsf{T}_{name}$, S being a server with a strand $s \in Serv$, $g \in T$ and $K_{XS} \notin \mathsf{K_P}$, can be originated in a penetrator node.*

Proof. Let $U = \{K_{XS}\}$ and $\mathsf{k} = \mathsf{K}$. We can safely state that K_{XS} is not originated in any regular node in $\mathcal{C}$ (this means that no regular node in $\mathcal{C}$ is an entry point for $I_{\mathsf{k}}[U]$), since the Serv, Init and Resp traces do not contain any node in which a term has K_{XS} as subterm (only the session key K appears as subterm, and by definition of Serv$K$ it is not a long term key). So we may apply Corollary 2.15. $\square$

Proposition 4.3. *Let $\mathcal{C}$ be a bundle in Σ and K be the session key generated by the server, $g \in \mathsf{T}$ and $K \notin \mathsf{K_P}$. Then, if K remains secret during the whole protocol run, no term of the form $\{g\}_K$ can originate in a penetrator node.*

Proof. Let us analyse all possible penetrator strands:

F, T, S, D: By Lemmas 2.1 and 2.2, none of these strands can contain a node that is an originating occurrence of a term in the form $\{g\}_K$.

C: Any term of the form $\{g\}_K$ is atomic by definition, and so it cannot be a concatenation of other terms. So, no strand of this type can contain an originating instance of a term of this form.

K: As $\{g\}_K \notin \mathsf{K_P}$, this case does not apply.

E: If $K \notin \mathsf{K_P}$, and if K remains secret during the whole protocol run, then no strand of this type can contain a node which has received the key K. So, a strand of this type can never contain an originating instance of a term in the form $\{g\}_K$, with K secret during the whole protocol run.

M: The penetrator could generate a random atomic message that would coincide with $\{g\}_K$. As this is highly unlikely, we assume it will not happen.

So no node that originates a term of the form $\{g\}_K$, $K \in \mathsf{K_P}$ with K secret during the whole protocol execution can be in a penetrator strand. $\square$

4.3.5 Initiator's Guarantee

Let $\mathcal{C}$ be a bundle in Σ, N_a uniquely originating in $\mathcal{C}$ and $K_{AS}, K_{BS}, K_{AB} \notin \mathsf{K_P}$; and let $s \in \text{Init}[A, B, N_a, T_a, L, K_{AB}, H]$.

1. If s has $\mathcal{C}$-height ≥ 2 then there is a strand $s_{serv} \in \text{Serv}[A, B, N_a, L, K_{AB}]$ in bundle $\mathcal{C}$ with $\mathcal{C}$-height $= 2$.

Proof. If $s \in \mathit{Init}$ and s has $\mathcal{C}$-height ≥ 2, then there is a node $n_{s,1} \in s$ which is the second node in strand s and $\mathit{term}(n_{s,1}) = -\{K_{AB}\ B\ L\ N_a\}_{K_{AS}} H$. By Lemmas 2.1 and 2.2, we know that there is a node $m \in \mathcal{C}$ with $\mathit{term}(m) = +v_1 H$ which generated $\{v_1 H\}$ (m is a minimal member regarding to $v_1 H$).

According to Prop. 4.2, v_1 was originated in a regular node in $\mathcal{C}$. By the structure[2] of v_1, it can only originate in a node $n_{serv,1}$ of a strand $s_{serv} \in \mathit{Serv}$. Besides, only this node has a term with structure $+v_1 H$. So, v_1 is uniquely originating in $\mathcal{C}$ by node $n_{serv,1}$, what leads to the conclusion that $m = n_{serv,1}$. Then there is a strand $s_{serv} \in \mathit{Serv}[A, B, N_a, L, K_{AB}]$ in $\mathcal{C}$ with $m \in s_{serv}$. By the efinition of bundle, if there is a node $n_{serv,1}$ in $\mathcal{C}$, then there is also a node $n_{serv,0}$ in $\mathcal{C}$. So, s_{serv} has $\mathcal{C}$-height $= 2$. □

This proof shows, without using the assumption $K_{AS}, K_{AB} \notin \mathsf{K_P}$, that not only the Initiator has the guarantee that there is in fact a valid Server, but also that this Server agrees with him about the values of A, B, N_a and K_{AB}. This agreement provides entity authentication to the Initiator regarding the Server, and that the session key received by the Initiator is the same as the generated by the Server. The Responder can also conclude that the Server executed a complete protocol run. Finally, as N_a was uniquely originated by the Initiator in this protocol run, the fact that Initiator and Server both agree over N_a provides liveness of the latter to the Initiator.

2. If s has $\mathcal{C}$-height $= 4$, then there is a strand $s_{resp} \in \mathrm{Resp}[A, B, T_a, L, K_{AB}]$ in bundle $\mathcal{C}$ with $\mathcal{C}$-height $= 2$.

Proof. If $s \in \mathrm{Init}$ and s has $\mathcal{C}$-height $= 4$, then there is a node $n_{s,3} \in s$ which is the fourth node in strand s and $\mathit{term}(n_{s,3}) = -\{T_a\}_{K_{AB}}$. By Lemmas 2.1 and 2.2, we know that there is a node $m \in \mathcal{C}$ with $\mathit{term}(m) = +\{T_a\}_{K_{AB}}$ which generated the message $\{T_a\}_{K_{AB}}$ (m is a minimal member regarding $\{T_a\}_{K_{AB}}$).

According to Prop. 4.3, $\{T_a\}_{K_{AB}}$ was originated in a regular node in $\mathcal{C}$. By the structure of $\{T_a\}_{K_{AB}}$, it can only originate in a Responder strand, in its last node $n_{resp,1}$. So, $\{T_a\}_{K_{AB}}$ is uniquely originating in $\mathcal{C}$ by node $n_{resp,1}$, what leads to the conclusion that $m = n_{resp,1}$. Then there is in $\mathcal{C}$ a strand $s_{resp} \in \mathit{Resp}[*, *, *, *, *]$ with $m \in s_{resp}$. By the definition of bundle, if there is a node $n_{resp,1}$ in $\mathcal{C}$, then there is also node $n_{resp,0}$ in $\mathcal{C}$. So, s_{resp} has $\mathcal{C}$-height $= 2$.

Let us now derive the values which both Initiator and Responder agree:

- K_{AB}: clearly, if $\{T_a\}_{K_{AB}}$ can be decrypted with key K_{AB}, both Initiator and Responder agree over the session key, and key authentication was obtained by the Initiator.

[2]The structure of a term is related essentially to the following two factors: its width, which is defined by the number of atomic subterms of that term; and the domain of its fields, which restrains each of these subterms's nature.

- T_a: if the text decrypted by the Initiator is equal to T_a sent in $n_{s,2}$, agreement over T_a is established.
- A, L, B: if the Initiator receives $\{T_a\}_{K_{AB}}$, then he can suppose that H received on $n_{s,1}$ is a valid message of the form $\{K_{AB}\ A\ L\}_{K_{BS}}$, where B is the Responder's identity. Moreover, he can suppose that the second field of the message contains his own identity (otherwise the Responder would notice that this field and the first field of $\{A\ T_a\}_{K_{AB}}$ are different and would abort the protocol run). The only way we can obtain agreement of the Responder and Initiator over the value L is by using the Initiator's authentication regarding the Server. As Initiator and Responder agree over K_{AB}, the Initiator can suppose that message H not only was valid, but also recent and came from the Server. So, the third field of H can only be L.

We then have that $s_{resp} \in \text{Resp}[A, B, T_a, L, K_{AB}]$ and s_{resp} has $\mathcal{C}$-height $= 2$. □

This proof shows, without using the assumption $K_{AS} \notin \mathsf{K_P}$, that not only the Initiator has the guarantee that there is in fact a valid Responder, but also that this Responder agrees with him about the values of A, B, T_a, L and K_{AB}. This agreement provides entity authentication to the Initiator regarding the Responder, and key establishment. The Initiator can also conclude that the Responder executed a complete protocol run. Finally, if T_a can be used as time identification, the fact that Initiator and Responder both agree over T_a provides liveness of the latter to the Initiator.

4.3.6 Responder's guarantee

Let $\mathcal{C}$ be a bundle in Σ, N_b uniquely originating in $\mathcal{C}$ and $K_{AS}, K_{BS}, K_{AB} \notin \mathsf{K_P}$; and let $s \in \text{Resp}[A, B, T_a, L, K_{AB}]$.

1. If s has $\mathcal{C}$-height ≥ 1, then there is a strand $s_{serv} \in \mathit{Serv}[A, B, *, L, K_{AB}]$ in bundle $\mathcal{C}$ with $\mathcal{C}$-height $= 2$.

   Proof. If $s \in$ Resp and s has $\mathcal{C}$-height ≥ 1, then there is a node $n_{s,0} \in s$ which is the last node in strand s and $\mathit{term}(n_{s,0}) = -v_3v_2$. By Lemmas 2.1 and 2.2, we know that there is a node $m \in \mathcal{C}$ with $\mathit{term}(m) = +v_3v_2$ that generated message v_3v_2 (m is a minimal member regarding to v_3v_2).

   According to Prop. 4.2, v_2 was originated in a regular node in $\mathcal{C}$. By the structure of v_2, it can only originate in one of the following nodes: a node $n_{init,2}$ (third node in a Initiator strand) or a node $n_{serv,1}$ (second node in a Server strand). By precedence, the node which uniquely originates v_2 is the last one. By Definition 1.2, if there is a node $n_{serv,1}$ in $\mathcal{C}$, then there is also the node $n_{serv,0}$ in $\mathcal{C}$. So, there is a strand $s_{serv} \in \text{Serv}[A, B, *, L, K_{AB}]$ and s_{serv} has $\mathcal{C}$-height $= 2$. □

   This shows that not only the Responder has the guarantee that there is in fact a valid Server, but also that this Server agrees on values A, B, L and K_{AB}. This agreement provides entity authentication to the Responder regarding the Server, and that the session key received by the Responder is the same generated by the Server. The Responder can also conclude that the Server executed a complete protocol run. Finally, if L can be used as time identification, the fact that Responder and Server both agree over L provides liveness of the latter to the Responder.

2. If s has $\mathcal{C}$-height ≥ 1, then there is a strand $s_{init} \in \text{Init}[A, B, *, T_a, L, K_{AB}, H]$ in the bundle $\mathcal{C}$ with $\mathcal{C}$-height $= 3$, where $H = \{K_{AB}\ A\ L\}_{K_{BS}}$.

   Proof. If $s \in \mathit{Resp}$ and s has $\mathcal{C}$-height ≥ 1, then there is a node $n_{s,0} \in s$ which is the last node in strand s and $\mathit{term}(n_{s,0}) = -v_3v_2$. By Lemmas 2.1 and 2.2, we know that there is a node $m \in \mathcal{C}$ with $\mathit{term}(m) = +v_3v_2$ which generated this message (m is a minimal member regarding to $+v_3v_2$).

   According to Prop. 4.3, v_3 was originated in a regular node in $\mathcal{C}$. By the structure of $+v_3$, it can only be originated in an initiator strand, in its third node $n_{init,2}$. So, v_3 is uniquely originating in $\mathcal{C}$ by node $n_{init,2}$, what leads to the conclusion that $m = n_{init,2}$. There is a strand $s_{init} \in \text{Init}[A, *, *, T_a, *, *, H']$ in bundle $\mathcal{C}$ with $m \in s_{init}$. By Definition 1.2, if there is a node $n_{init,2}$ in $\mathcal{C}$, then there are also nodes $n_{init,0}$ and $n_{init,1}$ in $\mathcal{C}$. So, s_{init} has $\mathcal{C}$-height $= 3$.

   Using an argument similar to that in the discussion of the Initiator's guarantee, the Responder can be sure that $H' = H$. So, it is possible to derive the last agreements between Responder and Initiator, and we can conclude that $s_{init} \in \text{Init}[A, B, *, T_a, L, K_{AB}, H]$.

   We then have that $s_{init} \in \text{Init}[A, B, T_a, L, K_{AB}]$ and s_{init} has $\mathcal{C}$-height $= 2$. □

   This shows that not only the Responder has the guarantee that there is in fact a valid Initiator, but also that this Initiator agrees on the values of A, B, T_a, L, K_{AB} and H. This agreement grants entity authentication to the Responder regarding the Initiator, and key establishment. Finally, as the Responder's challenge T_a can be used as time identification, the fact of both Responder and Initiator both agree over T_a provides liveness of the latter to the Responder.

We conclude that modified Kerberos satisfies both mutual authentication and key establishment requirements. For that to occur, some special considerations are necessary, such as the link between Server authentication and the other principal authentication in both cases.

As for the method, it is clear that it is much easier to produce proofs in strand spaces for protocols in which messages cannot be mistaken for others because of their width.

5 The Woo-Lam Π Protocol

The Woo-Lam Π authentication protocol intendeds to authenticate the initiator to the responder, using an indirect challenge-response mechanism which is assisted by a key translation center (that is, the server). This protocol is known to have different attacks. Woo and Lam discuss the design ideas and the attacks on this protocol in [10]. Figure 6 describes this protocol.

1. $A \to B : A$
2. $B \to A : N_B$
3. $A \to B : \{N_B\}_{K_{AS}}$
4. $B \to S : \{A, \{N_B\}_{K_{AS}}\}_{K_{BS}}$
5. $S \to B : \{N_B\}_{K_{BS}}$

Figure 6: The Woo-Lam Π protocol

Note that the responder is unable to read the nonce from inside message 3, as he does not possess the long term key K_{AS}. The responder then forwards this message to the server, who deciphers and re-encrypts it with another key (the responder's long term key).

In this section we show that the Woo-Lam Π fails to provide unilateral entity authentication. We show two security failures: One leads to the attack shown in figure 6, published in [12], and the other is explored by Abadi's attack to the protocol [10].

5.1 BAN verification of Woo-Lam Π

5.1.1 Idealisation

1. $A \to B :$
2. $B \to A :$
3. $A \to B : \{N_B\}_{K_{AS}}$
4. $B \to S : \left\{A, \{N_B\}_{K_{AS}}\right\}_{K_{BS}}$
5. $S \to B : \{N_B\}_{K_{BS}}$

5.1.2 Assumptions

A1. $B \mid\equiv \sharp(N_B)$

A2. $A \mid\equiv A \xleftrightarrow{K_{AS}} S$

A3. $B \mid\equiv B \xleftrightarrow{K_{BS}} S$

A4. $S \mid\equiv A \xleftrightarrow{K_{AS}} S$

A5. $S \mid\equiv B \xleftrightarrow{K_{BS}} S$

5.1.3 Analysis

D1. $\dfrac{B \lhd \{N_B\}_{K_{AS}}, B \models B \overset{K_{BS}}{\longleftrightarrow} S}{B \models S \mid\!\sim N_B}$ (M5, A3 by MM)

D2. $\dfrac{B \models S \mid\!\sim N_B, B \models \sharp(N_B)}{B \models S \models N_B}$ (D1, A1 by NV)

It is not possible to derive any belief of B with respect to A. The only conclusion B can arrive to is that there is an active Server S (sentence D2).

5.2 SVO verification of Woo-Lam Π

5.2.1 Assumptions

A1. $B \models \sharp(N_B)$

A2. $A \models A \overset{K_{AS}}{\longleftrightarrow} S$

A3. $B \models B \overset{K_{BS}}{\longleftrightarrow} S$

A4. $S \models A \overset{K_{AS}}{\longleftrightarrow} S$

A5. $S \models B \overset{K_{BS}}{\longleftrightarrow} S$

5.2.2 Annotation

N1. $B \lhd (A)$

N2. $A \lhd (N_B)$

N3. $B \lhd \{N_B\}_{K_{AS}}$

N4. $S \lhd \{A, \{N_B\}_{K_{AS}}\}_{K_{BS}}$

N5. $B \lhd \{N_B\}_{K_{BS}}$

5.2.3 Comprehension

C1. $B \models B \lhd (A)$

C2. $A \models A \lhd (\langle N_B \rangle_{*A})$

C3. $B \models B \lhd \langle \{N_B\}_{K_{AS}} \rangle_{*B}$

C4. $S \models S \lhd \left\{A, \{N_B\}_{K_{AS}}\right\}_{K_{BS}}$

C5. $B \models B \lhd (\{N_B\}_{K_{BS}})$

5.2.4 Interpretation

I1.

I2.

I3. $B \models B \lhd \langle\{N_B\}_{K_{AS}}\rangle_{*B} \rightarrow B \models B \lhd (\langle\{N_B, \sharp(N_B)\}_{K_{AS}}\rangle_{*B})$

I4. $S \models S \lhd \left\{A, \{N_B\}_{K_{AS}}\right\}_{K_{BS}} \rightarrow S \models S \lhd \left\{A, \{N_B, \sharp(N_B)\}_{K_{AS}}\right\}_{K_{BS}}$

I5. $B \models B \lhd (\{N_B\}_{K_{BS}}) \rightarrow B \models B \lhd (\{N_B, \sharp(N_B)\}_{K_{BS}})$

5.2.5 Derivation

D1. $B \models B \lhd \{N_B, \sharp(N_B)\}_{K_{BS}}$ (I5, C5 by MP)

D2. $D2a : B \models S \mid\sim N_B$ (D1, A3 by SAA)

D3. $B \models S$ *says* N_B (D2, A1 by FA, NVA)

It is not possible to derive any belief of B with respect to A. The only conclusion B can arrive to is that there is an active Server S (sentence D3).

5.3 Strand space verification of Woo-Lam Π

Figure 7 shows the strand space representation of the Woo-Lam Π protocol.

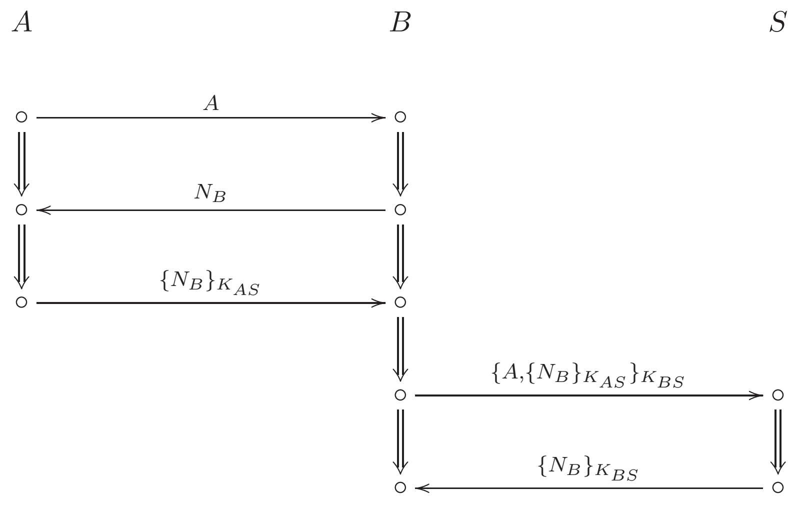


Figure 7: Woo-Lam Π protocol

We specialise the term algebra A: let $\mathsf{T}_{\text{name}} \subset \mathsf{T}$ be the set of principal names.

5.3.1 Regular strands definition

There are three types of regular strands in this protocol:

- A strand $s \in \text{Init}[X, N]$ iff s has trace of the form

$$\langle +X, \;\; -N, \;\; +\{N\}_{K_{XS}} \rangle,$$

  where $X \in \mathsf{T}_{\text{name}}$ and $N \notin \mathsf{T}_{\text{name}}$. The principal associated with a strand $s \in \text{Init}[X, N]$ is A.

- A strand $s \in \text{Resp}[X, Y, N, H]$ iff s has trace of the form

$$\langle -X, \;\; +N, \;\; -H, \;\; +\{X\ H\}_{K_{YS}}, \;\; -\{N\}_{K_{YS}} \rangle,$$

  where $X, Y \in T_{name}$. The principal associated with a strand $s \in Resp[X, Y, N, H]$ is B.

- A strand $s \in \text{Serv}[X, Y, N]$ iff s has trace of the form

$$\langle -\{X\ \{N\}_{K_{XS}}\}_{K_{YS}}, \;\; +\{N\}_{K_{XS}} \rangle,$$

  where $X, Y \in \mathsf{T}_{\text{name}}$ and $N \notin \mathsf{T}_{\text{name}}$. The principal associated with a strand $s \in textServ[X, Y, N, L, K]$ is S.

The sets Serv, Resp and Init are pairwise disjoint.

5.3.2 Authentication

To verify the authentication property of the Woo-Lam Π protocol, one proposition is necessary:

Proposition 5.1. *Let $\mathcal{C}$ be a bundle in Σ. No term of the form $\{g\}_{K_{XS}}$, with $X \in \mathsf{T}_{name}$, S a server with a strand $s \in Serv$, $g \in T$ and $K_{XS} \notin \mathsf{K}_{\mathsf{P}}$, can be originated in a penetrator node.*

Proof. Let $U = \{K_{XS}\}$ and $\mathsf{k} = \mathsf{K}$. We can safely state that K_{XS} does not originate in any regular node in $\mathcal{C}$ (this means that no regular node in $\mathcal{C}$ is an entry point for $I_{\mathsf{k}}[U]$), since the traces of Serv, Init and Resp do not contain any node whose term has K_{XS} as subterm (only the session key K appears as subterm, and by the definition of Serv, K is not a long term key). So we may apply Corollary 1.2. $\square$

5.3.3 Responder's guarantee

Woo-Lam Π does not provide to the Responder the existence of an Initiator or a Server in the same protocol run.

Let $\mathcal{C}$ be a bundle in Σ, N_b uniquely originating in $\mathcal{C}$ and $K_{AS}, K_{BS} \notin \mathsf{K}_{\mathsf{P}}$, and $s \in \text{Resp}[A, B, N_b]$.

Theorem 5.2. *If s has $\mathcal{C}$-height $= 5$, then there is a strand $s_{serv} \in Serv[A, B, N_b]$ in bundle $\mathcal{C}$ with $\mathcal{C}$-height $= 2$.*

Proof. If $s \in$ Resp and s has $\mathcal{C}$-height $= 5$, then there is a node $n_{s,4} \in s$ which is the last node in strand s and $term(n_{s,4}) = \{N_b\}_{K_{BS}}$. By Lemmas 2.1 and 2.2, we know that there is a node $m \in \mathcal{C}$ with $term(m) = +uns\_term(n_{s,4})$.

According to Prop. 5.1, $\{N_b\}_{K_{BS}}$ originates in a regular node in $\mathcal{C}$. By the structure of $\{N_b\}_{K_{BS}}$, it can only originate in node $n_{init,2}$ (third node of an Initiator strand) or in node $n_{serv,1}$ (second node in a Server strand).

In the first case, term $\{N_b\}_{K_{BS}}$ would have originated in a strand $s_{init} \in \text{Init}[B, N_b]$. This would be possible if B were participating in a parallel execution of the protocol as an Initiator and had received its very nonce N_b. This flaw is described in Figure 8 and was published in [12].

1. $I_A \rightarrow B : A$
2. $B \rightarrow I_A : N_B$

1'. $B \rightarrow I_C : B$

2'. $I_C \rightarrow B : N_B$

3'. $B \rightarrow I_C : \{N_B\}_{K_{BS}}$

3. $I_A \rightarrow B : H$
4. $B \rightarrow I_S : \{A, H\}_{K_{BS}}$
5. $I_S \rightarrow B : \{N_B\}_{K_{BS}}$

Figure 8: An attack on Woo-Lam Π

In the second case, term $\{N_b\}_{K_{BS}}$ would have originated in a strand $s_{serv} \in \text{Serv}[*, B, N_b]$. There is nothing that guarantees that the Initiator that B sees is the same that S authenticates. This fact is used in Abadi's attack to the protocol [10]. □

This shows that the Responder cannot obtain any guarantee with respect to the existence of an Initiator or a Server in the same protocol run. The attack based on the first case does not even require the existence of an active server – the penetrator can impersonate a Responder and a Server by forcing B to initiate a protocol run and use it as an oracle. On the other hand, Abadi's attack needs an active server.

6 Comparison of the verification techniques

Both BAN and SVO logics aim at verifying authentication properties such as liveness and freshness, but they cannot verify secrecy. Although these logics support the verification of key establishment, it is not possible to check whether such a key is secret throughout the entire protocol run. Furthermore, both BAN and SVO deal with only two epochs: past, which is the period of time before the current protocol run, and present, which represents the entire current protocol run. Therefore, to conclude that a session key recently established is good means proving that this key was good in some moment after the beginning of the protocol run. If the key is compromised *during* the protocol run, it is not possible to detect this flaw.

On the other hand, the strand space technique has finer time granularity than BAN and SVO. Vertices in different “horizontal lines” represent events that happen in different time moments, and therefore it is possible to interpret that an event happened before another in the same protocol run. This allows for detecting the compromise of information during a protocol run, as well as establishing a causal relation among events. Furthermore, it is possible to express and prove secrecy properties, and hence the technique may be applied to a broader class of protocols. It is worth noting that secrecy is expressed via ideals in the term algebra. Such specializations provide means for expressing and proving properties beyond the original scope of the technique.

Another difference between strand spaces and BAN/SVO is that the latter does not have an explicit intruder model. In strand spaces, every possible intruder action is modelled as a penetrator strand. Yet another difference of strand spaces is its pictorial representation of protocols. According to [12], strand space pictures provide information on protocol attacks, correctness theorems of the protocol and crucial steps in the proofs of these theorems.

Although strand spaces seem to be more flexible than BAN and SVO, none of these techniques is able to detect typing attacks, where a penetrator thwarts the original structure of messages in order to force a regular entity to misinterpret information. For example, regular entity A might be expecting message $m = X \ \{Y \ Z\}_K$ and the penetrator sends $m' = \{X' \ Y' \ Z'\}_K$. BAN, SVO and strand spaces assume that every entity is able to detect the structure of messages, and the protocol run should be aborted if messages do not respect the structure defined in the protocol specification. Although there are countermeasures such as explicit typing in order to avoid typing attacks [13, 14], not every protocol was/is designed with such defences, either because of design flaws or performance/bandwidth issues. Consequently, it is possible that a protocol proved correct by one of these techniques is actually vulnerable to typing attacks.

Table 1 summarises this comparison of BAN, SVO and strand spaces.

	BAN	SVO	Strand Spaces
Freshness	✓	✓	✓
Secrecy	—	—	✓
Time scope	past/present	past/present	each event
Soundness	—	✓	—
Intruder model	—	—	explicit
Pictorial repr.	—	—	✓
Typing attacks	—	—	—

Table 1: Comparison of BAN, SVO and strand spaces

7 Conclusions and Final Remarks

Formal verification methods by theorem proving like BAN, SVO and strand spaces allow protocol modelling, as well as proving a set of cryptographic properties, without the need to explore a potentially large state space. Although some insight might be necessary to identify the theorems that must be proved in order to deem the protocol correct, protocol verification tends to be quite straightforward. On the other hand, computer automation becomes harder due to the reasoning level demanded by the process.

In BAN and SVO, analysis is particularly straightforward. Syntaxes are quite simple and proofs can be done quite fast. Both methods can only prove authentication properties, though, and they are not meant for secrecy analysis. Another weakness is its vulnerability to typing attacks, as mentioned by Syverson in [15]. In the same text, Syverson also discusses the possible meaning of an entity's belief in a nonce. In our work we have presented this expression as liveness: if an entity B believes that another entity S believes any value that B knows it is recent, than B should be able to conclude that S is or was recently alive. Otherwise, if S had not been alive at the time of the nonce generation, how could S believe it? Despite this interpretion of liveness as a belief of a belief, the attack on the Woo-Lam Π protocol presented in this text shows that this liveness expression can be obtained with both logics for that protocol without S even being alive during or immediately before that protocol run. Therefore, this attack on Woo-Lam Π shows that it might not be possible to always interpret liveness as a belief-belief predicate. This attack, mapped in the strand spaces framework, is a replay attack.

Verification with BAN and BAN-derived logics, as well as strand spaces, have some computer support. J. Schumann used SETHEO, a theorem prover, to implement an automated proof program for BAN [16]. S. Brackin extended GNY to a Higher Order Logic theory where automated theorem proving is implemented in HOL90/Standard ML of new Jersey [17] and HOL98/Moscow ML [18, 19, 20]. A. H. Dekker defined a SVO-style logic whose set of rules is implemented in the Isabelle theorem prover [21]. He also proved the soundness of his logic using Kripke possible world semantics. B. Jacobs defined a BAN-style logic with strand space semantics [22], which is represented in the higher order logic of the theorem prover PVS. Song et al. extended the strand space model and defined a logic based on strand spaces. They devised a proof technique for this logic which is implemented in Athena, a tool written in Standard ML of New Jersey [23]. These two last examples are interesting implementations of computer-supported verification using both logic and strand spaces.

One pertinent fact related to the strand spaces technique is that it was used to prove that explicit tagging prevents typing attacks [14]. This shows that the technique is comprehensive enough so that general statements about cryptographic protocols can be proved.

We therefore conclude that verification in strand spaces, although more complex and less intuitive than in BAN and SVO logics, is the most complete and general of the three techniques.

References

[1] Michael Burrows, Martín Abadi, and Roger Needham. A logic of authentication. *ACM Transactions on Computer Systems*, 8(1):18–36, 1990.

[2] Paul F. Syverson and Paul C. van Oorschot. A unified cryptographic protocol logic. Technical Report NRL Publication 5540-227, Naval Research Lab, 1996.

[3] Lawrence C. Paulson. Proving properties of security protocols by induction. In *Proceedings of the 10th Computer Security Foundations Workshop (CSFW '97)*, page 70, Rockport, MA, USA, June 1997. IEEE Computer Society Press.

[4] F. Javier Thayer, Jonathan C. Herzog, and Joshua D. Guttman. Strand spaces: Proving security protocols correct. *Journal of Computer Security*, 7(2–3):191–230, 1999.

[5] Gavin Lowe. Breaking and fixing the Needham-Schroeder public-key protocol using FDR. In *Proceedings of the Second International Workshop on Tools and Algorithms for Construction and Analysis of Systems*, pages 147–166. Springer-Verlag, 1996.

[6] John C. Mitchell, Mark Mitchell, and Ulrich Stern. Automated analysis of cryptographic protocols using Murϕ. In *Proceedings of the 1997 IEEE Symposium on Security and Privacy*, page 141. IEEE Computer Society Press, May 1997.

[7] Edmund M. Clarke, Somesh Jha, and Will Marrero. Verifying security protocols with Brutus. *ACM Trans. Softw. Eng. Methodol.*, 9(4):443–487, 2000.

[8] Catherine Meadows. The NRL protocol analyzer: An overview. *Journal of Logic Programming*, 26(2):113–131, February 1996.

[9] Colin Boyd and Anish Mathuria. *Protocols for Authentication and Key Establishment*. Springer, 2003.

[10] Thomas Y. C. Woo and Simon S. Lam. A lesson on authentication protocol design. *SIGOPS Oper. Syst. Rev.*, 28(3):24–37, 1994.

[11] Paul F. Syverson. A taxonomy of replay attacks. In *Proceedings of Computer Security Foundations Workshop VI, 14–16 June 1994*, pages 187–191. IEEE, 1994.

[12] F. Javier Thayer Fábrega, Jonathan C. Herzog, and Joshua D. Guttman. Strand space pictures. In *Electronic Proceedings of Workshop on Formal Methods and Security Protocols*, 1998.

[13] Tuomas Aura. Strategies against replay attacks. In *Proceedings of the 10th Computer Security Foundations Workshop (CSFW '97)*, pages 59–68. IEEE Computer Society Press, 1997.

[14] James Heather, Gavin Lowe, and Steve Schneider. How to prevent type flaw attacks on security protocols. *Journal of Computer Security*, 11(2):217–144, 2003.

[15] Paul F. Syverson and Iliano Cervesato. The logic of authentication protocols. In Riccardo Focardi and Roberto Gorrieri, editors, *Foundations of Security Analysis and Design (FOSAD 2000), Tutorial Lectures*, volume LNCS 2171 of *Lecture Notes in Computer Science*, pages 63–136. Springer-Verlag, September 2000.

[16] Johann Schumann. Automatic verification of cryptographic protocols with SETHEO. In *Proceedings of the 14th International Conference on Automated Deduction, CADE-14*, volume LNAI 1249 of *Lecture Notes in Artificial Intelligence*, pages 87–100. Springer-Verlag, 1997.

[17] Stephen H. Brackin. A HOL extension of GNY for automatically analyzing cryptographic protocols. In *Proceedings of the Ninth IEEE Computer Security Foundations Workshop*, page 62. IEEE Computer Society Press, March 1996.

[18] Stephen H. Brackin. Using checkable types in automatic protocol analysis. In *Proceedings of the 15th Annual Computer Security Applications Conference*, page 99, Phoenix, AZ, USA, December 1999. IEEE Computer Society Press.

[19] Stephen H. Brackin. Implementing effective automatic cryptographic protocol analysis. In *Proceedings of the 14th IEEE International Conference on Automated Software Engineering*, page 319, Cocoa Beach, FL, USA, October 1999. IEEE Computer Society Press.

[20] Stephen H. Brackin. Automatically detecting most vulnerabilities in cryptographic protocols. In *In Proceedings of DARPA Information Survivability Conference & Exposition*, volume 1, page 222, Hilton Head, South Carolina, USA, January 2000. IEEE Computer Society Press.

[21] Anthony H. Dekker. C3PO: A tool for automatic sound cryptographic protocol analysis. In *Proceedings of the 13th IEEE Computer Security Foundations Workshop (CSFW'00)*, page 77. IEEE Computer Society Press, July 2000.

[22] Bart Jacobs. Semantics and logic for security protocols. September 2004.

[23] Dawn Xiaodong Song, Sergey Berezin, and Adrian Perrig. Athena: a novel approach to efficient automatic security protocol analysis. *Journal of Computer Security*, 9(1/2):47–74, 2001.