A BOUND ON ATTACKS ON AUTHENTICATION PROTOCOLS

Scott D. Stoller*
Computer Science Dept., SUNY at Stony Brook, Stony Brook, NY 11794-4400 USA

Abstract Authentication protocols are designed to work correctly in the presence of an adversary that can prompt honest principals to engage in an unbounded number of concurrent executions of the protocol. This paper establishes a bound on the number of protocol executions that could be useful in attacks. The bound applies to a large class of protocols, which contains versions of some well-known authentication protocols, including the Yahalom, Otway-Rees, and Needham-Schroeder-Lowe protocols.

1. Introduction

Many protocols are designed to work correctly in the presence of an adversary—hereafter called a penetrator—that can prompt honest principals to engage in an unbounded number of concurrent executions of the protocol. This paper focuses on authentication (including key establishment). Authentication protocols should satisfy at least two kinds of correctness requirements: *secrecy*, which states that certain values are not obtained by the penetrator, and *agreement*, which states, *e.g.*, that a principal's conclusion about the identity of a principal with whom it is communicating is never incorrect. Authentication protocols are short and look deceptively simple, but numerous flawed or weak protocols have been published. This attests to the importance of rigorous verification.

Allowing an unbounded number of concurrent protocol executions makes the number of reachable states unbounded, so automated verification using state-space exploration is not directly applicable. State-space exploration is feasible when small upper bounds are imposed on the size of messages and the number of protocol executions. Therefore, reduction theorems are needed, which show

*The author gratefully acknowledges the support of NSF under Grant CCR-9876058 and the support of ONR under Grants N00014-99-1-0358 and N00014-01-1-0109. This work was started while the author was at Indiana University in Bloomington. Email: stoller@cs.sunysb.edu

that if a protocol is correct in a system with certain bounds on these parameters, then the protocol is correct in the unbounded system as well.

Our reduction is formulated in the strand space model [13] but is relatively model-independent. A regular strand can be regarded as a thread that runs the program corresponding to one role (*e.g.*, initiator or responder) of the protocol and then terminates; thus, a regular strand corresponds to one execution of one role. Our reduction imposes three significant restrictions on protocols.

Shallow ciphertext restriction: the protocol does not use nested ciphertexts. This is easily checked by static analysis of the program, so we call it a *static restriction*. (This restriction can be relaxed; see Section 5.)

Bounded Support Restriction (BSR): in every history (*i.e.*, every possible behavior) of the system, each regular strand depends on at most a given number of regular strands. Correct authentication protocols are designed to involve only a small number of participants and hence typically satisfy BSR.

Revealed Genval Restriction (RGR): every genval revealed to the penetrator is revealed "directly", *i.e.*, the penetrator needs to perform at most one decryption on an intercepted message to obtain each genval.

The notion of dependence underlying BSR is a variant of Lamport's happened-before relation [5], modified to treat nonces and session keys—collectively called *generated values*, or *genvals* for short—appropriately. For example, if a genval g generated on strand s_1 appears in messages received by strand s_2 but only in contexts in which it could be replaced with a value generated by the penetrator, then g's presence in those messages does not cause s_2 to depend on s_1.

It seems difficult to develop static analyses to check BSR and RGR, so we call them *dynamic restrictions* and propose to check them during state-space exploration. Thus, we need reductions for them as well as for the correctness requirements. We prove: if a protocol satisfies the dynamic restrictions and correctness requirements when appropriate bounds are imposed on the number of regular strands in a history, then the protocol also satisfies the dynamic restrictions and correctness requirements without those bounds.

2. Related Work

Most existing techniques for automated verification of systems with unbounded numbers of processes, such as [3], are not applicable to authentication protocols, because they assume the set of values (equivalently, the set of local states of each process) is independent of the number of processes, whereas authentication protocols generate fresh nonces and session keys, so the set of values grows as the number of processes (strands) increases.

Roscoe and Broadfoot use data independence to bound the number of nonces that could be useful in attacks [10], assuming each honest principal participates in at most a given number of protocol executions at a time. Our reduction does not require such assumptions.

Lowe's reduction for authentication protocols [7] does not handle agreement requirements or known-key attacks and does not apply to the Otway-Rees [8], Yahalom [1], and Needham-Schroeder-Lowe (abbreviated NSL) [6] protocols, due to various restrictions.

Our reduction handles secrecy and agreement requirements, allows known-key attacks, and applies to some well-known protocols, including the Otway-Rees, Yahalom, and NSL protocols, after the Otway-Rees and Yahalom protocols have been modified slightly (in an obviously correctness-preserving way) to eliminate forwarding of ciphertexts, as in [7, 10].

Heather and Schneider's method [4] can efficiently (compared to state-space exploration) verify protocols for which a rank function exists. Currently, our method, unlike theirs, can verify secrecy properties for protocols that use temporary secrets, while their method, unlike ours, accommodates forwarded ciphertexts. In the absence of completeness results, it is unclear whether requiring BSR or requiring existence of a rank function is more restrictive.

The reduction in [12] is more general in some ways than this one, but it does not handle session keys, so it does not apply to most authentication protocols, especially if session keys are used to encrypt protocol messages, as in the Needham-Schroeder shared-key [1], Yahalom, and Kerberos protocols.

3. Model of Authentication Protocols

We adopt the strand space model [13], with minor modifications. We introduce simple languages for authentication protocols and correctness requirements, similar to the languages in [2] and [14], respectively.

3.1. Term, Directed Term, and Trace

The set of *primitive terms* is the union of the following five disjoint sets. (1) *Text* is a set of arbitrary non-cryptographic values, with a distinguished subset *Name* containing names of principals. (2) *Nonce* is a set of nonces. (3) Key_{sess} is a set of session keys. (4) $Key_{sym} = \{key(x, y) \mid x, y \in Name\}$ is a set of long-term symmetric keys; informally, $key(x, y)$ is intended to be shared by x and y. (5) $Key_{asym} = \{pubkey(x) \mid x \in Name\} \cup \{pvtkey(x) \mid x \in Name\}$ is a set of long-term asymmetric keys; $pubkey(x)$ and $pvtkey(x)$ represent x's public and private keys, respectively.

The set *Term* of terms is defined inductively as follows, where $Key = Key_{sym} \cup Key_{asym} \cup Key_{sess}$. (1) All primitive terms are terms. (2) If t and t' are terms and $k \in Key$, then $encr(t, k)$ (encryption of t with k, usually written $\{t\}_k$) and $pair(t, t')$ (pairing of t and t', usually written $t{\cdot}t'$) are terms.

The function $\text{inv} \in Key \to Key$ maps each key to its inverse: decrypting $\{t\}_k$ with $\text{inv}(k)$ yields t. For a symmetric key k, $\text{inv}(k) = k$. We usually write $\text{inv}(k)$ as k^{-1}. We assume perfect encryption.

Elements of $Nonce \cup Key_{sess}$ are called *generated values*, or *genvals* for short. Let $\text{genvals}(t)$ be the set of genvals that occur in a term t. For $S \subseteq Term$, let $\text{genvals}(S) = \bigcup_{t \in S} \text{genvals}(t)$.

A *ciphertext* is a term whose outermost operator is *encr*. A term t' *occurs in the clear* in a term t if there is an occurrence of t' in t that is not in the scope of *encr*.

Let $|S|$ denote the size of a set S. Let $\mathrm{dom}(f)$ denote the domain of a function f. A sequence is a function from a finite prefix of the natural numbers to elements. Let $\mathrm{len}(\sigma)$ denote the length of a sequence σ. $\langle\!\langle a, b, \ldots \rangle\!\rangle$ denotes a sequence σ with $\sigma(0) = a$, $\sigma(1) = b$, and so on.

A *directed term* is $+t$ or $-t$, where t is a term. Positive and negative terms represent sending and receiving messages, respectively. Let $\pm Term$ denote the set of directed terms. For a directed term t, the *absolute value* of t, denoted $\mathrm{abs}(t)$, is t without its direction; for example, $\mathrm{abs}(-A) = A$. For $S \subseteq \pm Term$, let $\mathrm{abs}(S) = \{\mathrm{abs}(t) \mid t \in S\}$. We often refer to directed terms as terms.

A *trace* is a finite sequence of directed terms. Let $(\pm Term)^*$ denote the set of traces.

3.2. Strand Space

A *strand space* is a function $tr \in \mathrm{dom}(tr) \to (\pm Term)^*$, where $\mathrm{dom}(tr)$ is an arbitrary set whose elements are called *strands* (think of them as "strand identifiers").

A *node* of tr is a pair $\langle s, i \rangle$ with $s \in \mathrm{dom}(tr)$ and $0 \leq i < \mathrm{len}(tr(s))$. Let $\mathcal{N}_{tr}$ denote the set of nodes of tr. We say that node $\langle s, i \rangle$ is on strand s. Let $\mathrm{nodes}_{tr}(s)$ denote the set of nodes on strand s in tr. Let $\mathrm{strand}(\langle s, i \rangle) = s$, $\mathrm{index}(\langle s, i \rangle) = i$, and $\mathrm{term}_{tr}(\langle s, i \rangle) = tr(s)(i)$. For $S \subseteq \mathcal{N}_{tr}$, let $\mathrm{strand}(S) = \{\mathrm{strand}(n) \mid n \in S\}$ and $\mathrm{term}_{tr}(S) = \{\mathrm{term}_{tr}(n) \mid n \in S\}$. If $\mathrm{term}_{tr}(n)$ is positive (or negative), we say that n is positive (or negative).

The local dependence relation on nodes is defined by: $n_1 \overset{lcl}{\to} n_2$ iff $\mathrm{strand}(n_1) = \mathrm{strand}(n_2)$ and $\mathrm{index}(n_2) = \mathrm{index}(n_1) + 1$.

A term t *originates* from a node $\langle s, i \rangle$ in tr iff $\langle s, i \rangle$ is positive, t is a subterm of $\mathrm{term}_{tr}(\langle s, i \rangle)$, and t is not a subterm of $\mathrm{term}_{tr}(\langle s, 0 \rangle), \mathrm{term}_{tr}(\langle s, 1 \rangle), \ldots$, or $\mathrm{term}_{tr}(\langle s, i - 1 \rangle)$.

A term t *uniquely originates* from a node n in tr iff t originates from n in tr and not from any other node in tr. This is the strand space way of expressing freshness of genvals.

For symbols subscripted by a strand space, we elide the subscript when the strand space is evident from context.

3.3. Role and Protocol

Let *Param* be a set of parameters. The set of *parameterized terms* is defined like *Term* except with parameters as an additional base case.

A *role* r is a sequence of directed parameterized terms, with a type—*i.e.*, a set of allowed values—associated with each parameter, and with a subset of the parameters designated as uniquely-originated. Informally, parameters that represent genvals generated by r (and hence that first occur in r in a positive term) are so designated, to indicate that values of those parameters

must be uniquely-originated. In examples, uniquely-originated parameters are underlined in the parameter list. Roles must also satisfy some well-formedness conditions, detailed in [11], notably that types may not contain ciphertexts. Let $r.x$ denote parameter x of role r. For example, the roles for the initiator and responder in the NSL protocol [6] are

$$\begin{array}{ll}
\mathrm{Init}_{NSL}(i : \mathit{Name} \setminus \{P\},\ r : \mathit{Name}, & \mathrm{Resp}_{NSL}(i : \mathit{Name},\ r : \mathit{Name} \setminus \{P\}, \\
\qquad \underline{ni} : \mathit{Nonce},\ nr : \mathit{Nonce}) = & \qquad ni : \mathit{Nonce},\ \underline{nr} : \mathit{Nonce}) = \\
\langle\!\langle +\{ni \cdot i\}_{pubkey(r)}, & \langle\!\langle -\{ni \cdot i\}_{pubkey(r)}, \\
\quad -\{ni \cdot nr \cdot r\}_{pubkey(i)}, & \quad +\{ni \cdot nr \cdot r\}_{pubkey(i)}, \\
\quad +\{nr\}_{pubkey(r)} \rangle\!\rangle & \quad -\{nr\}_{pubkey(r)} \rangle\!\rangle.
\end{array}$$

In both roles, parameters i and r hold the names of the initiator and responder, respectively. We exclude P from the type of $\mathrm{Init}_{NSL}.i$ and $\mathrm{Resp}_{NSL}.r$, because we interpret P as the name of a dishonest principal (the penetrator), and we interpret $\mathrm{Init}_{NSL}.i$ and $\mathrm{Resp}_{NSL}.r$ as the name of the principal executing the role, and all actions of the penetrator are represented by traces for penetrator roles, described in Section 3.4. $\mathrm{Init}_{NSL}.ni$ and $\mathrm{Resp}_{NSL}.nr$ are uniquely-originated, because they represent nonces generated by their respective roles.

A *genval parameter* is a parameter with type Key_{sess} or *Nonce*.

A *trace for role* r is a prefix of a trace obtained by substituting for each parameter x of r a term in the type of x. For example, a trace for Init_{NSL} is $\sigma_0 = \langle\!\langle +\{ni_0 \cdot B\}_{pubkey(A)} \rangle\!\rangle$.

A role r and a trace σ for r uniquely determine a mapping, denoted $\mathit{args}(r, \sigma)$, from the parameters of r that appear in $r(0), r(1), \ldots, r(\mathrm{len}(\sigma) - 1)$ to *Term*. For example, $\mathrm{dom}(\mathit{args}(\mathrm{Init}_{NSL}, \sigma_0)) = \{i, r, ni\}$ and $\mathit{args}(\mathrm{Init}_{NSL}, \sigma_0)(i) = B$.

A *protocol* is a set of roles. For example, the NSL protocol is $\Pi_{NSL} = \{\mathrm{Init}_{NSL}, \mathrm{Resp}_{NSL}\}$.

3.4. Penetrator

The penetrator model is parameterized by a set $\mathit{pik} \subseteq \mathit{Term}$, called the *penetrator's initial knowledge*. Typically, we assume there is a single dishonest principal, named P, and take $\mathit{pik} \supseteq \mathit{pik}_0$, where $\mathit{pik}_0 = \{\mathit{pvtkey}(P)\} \cup \{\mathit{pubkey}(x) \mid x \in \mathit{Name}\} \cup \{\mathit{key}(P, x), \mathit{key}(x, P) \mid x \in \mathit{Name} \setminus \{P\}\}$.

Known-key attacks are modeled by including in *pik* the absolute values of terms appearing in some executions of the protocol and the genvals generated during those executions.

$\Pi_P(\mathit{pik})$, the set of *penetrator roles* for initial knowledge *pik*, contains

$$\begin{array}{ll}
\mathrm{Msg}(x : \mathit{Text} \cup \mathit{Nonce} \cup \mathit{Key}_{sess} \cup \mathit{pik}) = \langle\!\langle +x \rangle\!\rangle & \\
\mathrm{Pair}(x_1 : \mathit{Term}, x_2 : \mathit{Term}) = \langle\!\langle -x_1,\ -x_2,\ +x_1 \cdot x_2 \rangle\!\rangle & \\
\mathrm{Enc}(k : \mathit{Key}, x : \mathit{Term}) = \langle\!\langle -k,\ -x,\ +\{x\}_k \rangle\!\rangle & \\
\mathrm{Sep}_i(x_1 : \mathit{Term}, x_2 : \mathit{Term}) = \langle\!\langle -x_1 \cdot x_2,\ +x_i \rangle\!\rangle & \text{for } i \in \{1, 2\} \\
\mathrm{Dec}(k : \mathit{Key}, x : \mathit{Term}) = \langle\!\langle -k^{-1},\ -\{x\}_k,\ +x \rangle\!\rangle &
\end{array}$$

A trace σ for a role r is *compromised* if it is running the protocol with the penetrator as a partner, specifically, if $\mathit{args}(r, \sigma)(x) = P$ for some parameter x of r with type *Name*.

3.5. System and History

A *system* is a pair $\langle \Pi, pik \rangle$. For example, $\mathcal{M}_{NSL} = \langle \Pi_{NSL}, pik_{NSL} \rangle$, where pik_{NSL} is a superset of pik_0 that also contains terms and genvals from one execution of Π_{NSL}.

A *history* of a system $\langle \Pi, pik \rangle$ is a tuple $h = \langle tr, \overset{msg}{\rightarrow}, role \rangle$, where tr is a strand space, $\overset{msg}{\rightarrow}$ is a binary relation on $\mathcal{N}_{tr}$ (read $n_1 \overset{msg}{\rightarrow} n_2$ as "n_1 is the sending of a message received at n_2"), and *role* is a function from strands to roles (*i.e.*, $role \in \mathrm{dom}(tr) \rightarrow (\Pi \cup \Pi_P(pik))$) such that: (1) for each negative node n_2, there exists a unique positive node n_1 such that $n_1 \overset{msg}{\rightarrow} n_2$ and $\mathrm{abs}(\mathrm{term}(n_1)) = \mathrm{abs}(\mathrm{term}(n_2))$; (2) the happened-before [5] (also called causal dependence) relation $\preceq_h$, defined to be the reflexive and transitive closure of $\overset{msg}{\rightarrow} \cup \overset{lcl}{\rightarrow}$, is well-founded and acyclic; (3) for all $s \in \mathrm{dom}(tr)$, $tr(s)$ is a trace for $role(s)$; (4) for all $s \in \mathrm{dom}(tr)$, for all $x \in \mathrm{dom}(args(role(s), tr(s)))$, if parameter x is uniquely-originated and $tr(s)$ is uncompromised, then $args(role(s), tr(s))(x)$ is not in $\mathrm{genvals}(pik)$ and uniquely originates from $\langle s, i \rangle$, where i is the index of the first term in r that contains x.

If $role(s) = r$, then s is called a *strand for* r. If $role(s) \in \Pi$, then s is called a *regular strand*; otherwise, s is called a *penetrator strand*. Nodes on regular and penetrator strands are called *regular nodes* and *penetrator nodes*, respectively.

A system *satisfies* a predicate ϕ on histories iff all of its histories satisfy ϕ.

We sometimes use a history instead of a strand space as a subscript. For example, if $h = \langle tr, \overset{msg}{\rightarrow}, role \rangle$, we sometimes write $\mathcal{N}_h$ instead of $\mathcal{N}_{tr}$.

The set of predecessors of a node n in a history h is $\mathrm{preds}_h(n) = \{n' \in \mathcal{N}_h \mid n' \preceq_h n \ \wedge \ n' \neq n\}$.

A set S of nodes is *backwards-closed* with respect to a binary relation R iff, for all nodes n_1 and n_2, if $n_2 \in S$ and $n_1 \ R \ n_2$, then $n_1 \in S$. Given a history $h = \langle tr, \overset{msg}{\rightarrow}, role \rangle$ of a system $\mathcal{M}$, a set S of nodes that is backward-closed with respect to $\preceq_h$ can be regarded as a history of $\mathcal{M}$, denoted $\mathrm{nodesToHist}_h^{\mathcal{M}}(S)$, in a natural way.

3.6. Derivability

A term t is derivable (by the penetrator) from a set S of nodes of a history h of a system $\mathcal{M} = \langle \Pi, pik \rangle$, denoted $S \vdash_h^{\mathcal{M}} t$, if the penetrator can compute t from $\mathrm{term}_h(S) \cup pik$, by performing encryption, decryption, pairing, and separation (*i.e.*, projection) operations, and by generating genvals that are not in $\mathrm{uniqOrigRqrd}_h^{\mathcal{M}}(S)$, where $\mathrm{uniqOrigRqrd}_h^{\mathcal{M}}(S)$ is the set of genvals g that originate from a node in S and are required to be uniquely originated in h (by item (4) in the definition of history). Similar derivability relations or functions have been considered by several researchers, *e.g.*, [9]. The new twist here is in the treatment of genvals.

3.7. Correctness Requirements

Genval Secrecy. Informally, genval secrecy says: the values of specified genval parameters are not revealed to the penetrator. Formally, a genval secrecy requirement for a system $\langle \Pi, pik \rangle$ is specified by a set of uniquely-originated genval parameters of Π. A history $h = \langle tr, \stackrel{msg}{\rightarrow}, role \rangle$ of a system $\mathcal{M}$ satisfies a genval secrecy requirement G iff, for every $r.x \in G$, for every uncompromised regular strand s for r, if $x \in \mathrm{dom}(args(role(s), tr(s)))$, then $\mathcal{N}_{tr} \not\vdash_h^{\mathcal{M}} args(role(s), tr(s))(x)$. For example, $\mathcal{M}_{NSL}$ satisfies the genval secrecy requirement $\{\mathrm{Init}_{NSL}.ni, \mathrm{Init}_{NSL}.nr, \mathrm{Resp}_{NSL}.ni, \mathrm{Resp}_{NSL}.nr\}$.

Agreement. Informally, agreement says: if some uncompromised strand executes a certain role to a certain point with certain arguments, then some strand must have executed a certain role to a certain point with certain arguments. An agreement requirement for a protocol Π has the form "$\langle r_1, len_1, xs_1 \rangle$ precedes $\langle r_2, len_2, xs_2 \rangle$", where $r_1 \in \Pi$, $r_2 \in \Pi$, and xs_1 and xs_2 are sequences of parameters of r_1 and r_2, respectively, such that $\mathrm{len}(xs_1) = \mathrm{len}(xs_2)$ and for $j \in \{1, 2\}$, every parameter in xs_j occurs in $r_j(0), r_j(1), \ldots, r_j(len_j - 1)$. A history $\langle tr, \stackrel{msg}{\rightarrow}, role \rangle$ of a system $\langle \Pi, pik \rangle$ satisfies that agreement requirement iff, if tr contains an uncompromised strand s_2 such that $role(s_2) = r_2$ and $\mathrm{len}(tr(s_2)) \geq len_2$, then tr contains a strand s_1 such that $role(s_1) = r_1$ and $\mathrm{len}(tr(s_1)) \geq len_1$ and the sequence of arguments of s_2 corresponding to parameters xs_2 equals the sequence of arguments of s_1 corresponding to parameters xs_1. For example, $\mathcal{M}_{NSL}$ satisfies the agreement requirement $\langle \mathrm{Resp}_{NSL}, 1, \langle\!\langle i, r, ni, nr \rangle\!\rangle \rangle$ precedes $\langle \mathrm{Init}_{NSL}, 1, \langle\!\langle i, r, ni, nr \rangle\!\rangle \rangle$.

4. Restrictions

Hereafter, we consider only systems $\langle \Pi, pik \rangle$ that satisfy the following static restrictions.

Shallow Ciphertext Restriction. In every term in every role in Π and in every term in pik, $encr$ does not occur in the scope of $encr$.

Unsent Long-Term Keys Restriction. In every parameterized term in every role of Π and in every term in $pik \setminus (Key_{sym} \cup Key_{asym})$, the operators key, $pubkey$, and $pvtkey$ occur only in the second argument of $encr$. This implies that long-term keys not in pik are not sent in messages.

4.1. Support

Informally, a set S' of nodes supports a set S of nodes if S' contains all of the nodes in S and all of the regular nodes on which nodes in S depend. Let $\mathcal{N}_h^{\mathrm{reg}}$ denote the set of regular nodes in history h of system $\mathcal{M}$. For a genval g that uniquely originates in a history h, let $\mathrm{origin}_h(g)$ denote the node from which g originates in h.

A set S' of nodes is a *support* for a set S of nodes in a history h of a system $\mathcal{M}$ if

Su1. $S \subseteq S' \subseteq \mathcal{N}_h$, and S' is backwards-closed with respect to $\stackrel{lcl}{\rightarrow}$.

Su2. Every received term is derivable from preceding terms, *i.e.*, for all negative nodes n in S', $\text{preds}_h(n) \cap S' \cap \mathcal{N}_h^{\text{reg}} \vdash_h^{\mathcal{M}} \text{term}_h(n)$.

Su3. For $g \in \text{genvals}(\text{term}_h(S')) \cap (\text{uniqOrigRqrd}_h^{\mathcal{M}}(\mathcal{N}_h) \setminus \text{uniqOrigRqrd}_h^{\mathcal{M}}(S'))$, g occurs in the clear in $\text{term}_h(\text{origin}_h(g))$. (Su3 is needed for Lemma 2.)

If S' is a support for S, we say that S' supports S. For a strand s, if S' supports $\text{nodes}(s)$, we say that S' supports s. An algorithm for computing supports is in [11].

To illustrate the treatment of unique origination, consider the following history of a generic server-based authentication protocol that reveals at least one genval that originates from the initiator role; the Yahalom and Otway-Rees protocols are specific examples of this kind. Suppose s_I, s_R, and s_S are initiator, responder, and server strands, respectively, that interact without interference from the penetrator. Let n be a nonce that uniquely originates on s_I. The penetrator then behaves as an initiator, interacting with a responder strand s'_R and a server strand s'_S, except that the penetrator uses n instead of a fresh nonce. A support for s'_R or s'_S need *not* contain nodes on s_I. In that sense, s'_R and s'_S do not depend on s_I, even though the chain of messages that conveys n means that there is causal dependence between those nodes in the classical sense of Lamport [5]. Informally, that classical dependence can be ignored here because the penetrator could generate a nonce n' and replace n with n' in the terms of nodes on s'_R and s'_S. The careful treatment of unique origination in the definition of derivability allows such inessential classical dependencies to be ignored. If they were not ignored, few interesting protocols would satisfy BSR, defined in Section 4.2. The next lemma says that a support can be transformed into a history by adding only penetrator nodes.

Given a strand space tr, a strand $s \in \text{dom}(tr)$, and a set S of nodes of tr that is backwards-closed with respect to $\stackrel{lcl}{\rightarrow}$, S contains nodes on a prefix of $tr(s)$; let $\text{prefix}_{tr}(s, S)$ denote that prefix.

Lemma 1 *If S' is a support for S in a history $h = \langle tr, \stackrel{msg}{\rightarrow}, role \rangle$ of a system $\mathcal{M} = \langle \Pi, pik \rangle$, then there exists a history $h' = \langle tr', \stackrel{msg'}{\rightarrow}, role' \rangle$ of $\mathcal{M}$ such that*

$$(\forall s \in \text{strand}(S') : s \in \text{dom}(tr') \ \wedge \ tr'(s) = \text{prefix}_{tr}(s, S') \ \wedge \ role'(s) = role(s))$$
$$\wedge \ (\forall s \in \text{dom}(tr') \setminus \text{strand}(S') : role'(s) \in \Pi_P(pik))$$

Proof: h' is constructed by combining nodes in S with histories that witness the derivability of terms, as required by Su2. Details are in [11]. ∎

Lemma 2 *Supports are compositional, i.e., if S'_0 and S'_1 support S_0 and S_1, respectively, in a history h of a system $\mathcal{M}$, then $S'_0 \cup S'_1$ supports $S_0 \cup S_1$ in history h of $\mathcal{M}$.*

System	Strand count f for BSR(f)			$\mathrm{DW_R}$	Total strands		
	f(Init)	f(Resp)	f(Srvr)		Init	Resp	Srvr
NSL	1	1	none	2	3	3	none
Yahalom	1	2	1	2	3	6	3
Otway-Rees	1	1	1	2	3	3	3

Figure 1. Results for some well-known authentication protocols. $\mathrm{DW_R}$ is defined in Section 5. The right part of the table gives the total number of strands for each role that need to be considered in a history to verify correctness requirements and dynamic restrictions (*cf.* Section 8).

Proof: The proof is straightforward. Details are in [11]. ∎

4.2. Bounded Support Restriction

A *strand count* for a protocol Π is a function from Π to the natural numbers. A set S of nodes *has strand count* f iff, for each role r, S contains nodes from exactly $f(r)$ strands for r. If $\mathcal{N}_h$ has strand count f, then we say that history h has strand count f. We define a partial ordering $\preceq_{SC}$ on strand counts for a protocol: $\preceq_{SC}$ is the pointwise extension of the usual ordering on numbers.

A history h satisfies the *bounded support restriction for strand count* f, abbreviated BSR(f), iff for each regular strand s in h, there exists a support for s in h with strand count at most f.

Figure 1 lists some systems and, for each system, a strand count f for which the system satisfies BSR(f). By Theorem 2 in Section 6, these results can be verified automatically through state-space exploration of histories with strand counts bounded by the values in the right part of the table. Our method is not currently implemented, so these results were proven by hand, which is not difficult. The proof for the NSL protocol is in [11]; the other proofs are similar. Although we do not have formal guidelines for choosing a strand count f for verifying a given system, in practice, it appears that all correct and un-contrived authentication protocols satisfy BSR(f_2), where $f_2(r) = 2$ for all r.

4.3. Revealed Genval Restriction

A node n *directly reveals* a term t in a history h of a system $\mathcal{M}$ iff n is a positive regular node and $\{n\} \vdash_h^{\mathcal{M}} t$. A history h of a system $\mathcal{M}$ satisfies the *revealed genval restriction* (RGR) if, for every genval $g \in \mathrm{uniqOrigRqrd}_h^{\mathcal{M}}(\mathcal{N}_{tr})$, if the penetrator learns g (*i.e.*, $\mathcal{N}_h \vdash_h^{\mathcal{M}} g$), then h contains a node that directly reveals g. RGR prevents genvals from being revealed to the penetrator indirectly, *e.g.*, by encrypting one genval with another and then revealing the latter genval. RGR helps us obtain a static bound on the dependence width (see Section 5). The NSL, Yahalom, and Otway-Rees protocols satisfy RGR. By Theorem 2 in Section 6, this can be verified automatically through state-

space exploration of histories with the strand counts given in the right part of Figure 1. Currently. we proved these results by hand.

5. Dependence Width

Let r be a role of a system $\mathcal{M}$, and let i be the index of a negative parameterized term in r. An *instance* of $\langle r, i\rangle$ in a history h is a node $\langle s, i\rangle$ on a strand s for r. A *revealing set* for a term t at a node n in a history h of a system $\mathcal{M}$ is a set R of positive regular nodes of tr such that $R \cap \text{preds}_h(n) \vdash_h^{\mathcal{M}} t$.

The dependence width of $\langle r, i\rangle$ in $\mathcal{M}$ is the maximum, over all histories h of $\mathcal{M}$ and all instances n of $\langle r, i\rangle$ in h, of $|R \setminus \text{nodes}_h(\text{strand}(n))|$, where R is a minimum-size revealing set for $\text{term}_h(n)$ at n in h. For example, suppose h contains an instance $\langle s, i\rangle$ of $\langle r, i\rangle$ with $\text{term}_h(\langle s, i\rangle) = g_1 \cdot \{g_2\}_k$. Suppose g_1 and g_2 were sent in the clear at positive regular nodes n_1 and n_2 not on s, respectively, and $k \in pik$, and no other regular nodes send or receive g_1 and g_2. Then n_1 and n_2 together reveal t, and no single node reveals t, so the dependence width of $\langle r, i\rangle$ in h is at least 2. If we suppose instead that n_1 is on s, then we would not count n_1 in the dependence width. Note that a support for $\{n\}$ would (in general) include nodes that n_1 and n_2 causally depend on; a revealing set for n does not. Dependence width is used in the proof of Theorem 2 (in Section 6) to bound the number of strands involved in a violation of BSR.

Nodes on strand(n) are not counted in the dependence width, because dependence width is designed to bound the size of the index set of the rightmost union in equation (3), and those nodes appear in $\text{support}_{h_0}^{\mathcal{M}}(s_0)$ and hence are excluded from that index set.

The *dependence width* of a system $\mathcal{M}$ is the maximum, over all roles r of $\mathcal{M}$ and all negative parameterized terms $r(i)$ in r, of $\text{DW}(\langle r, i\rangle, \mathcal{M})$.

The proof of Theorem 2 relies on an upper bound on the dependence width of a system. It is convenient to base this bound on the syntactic structure of the protocol. This is difficult if a protocol sends terms of the forms $\{g\}_{k_1}$, $\{k_1\}_{k_2}$, $\{k_2\}_{k_3}$, …, $\{k_{i-1}\}_{k_i}$, k_i; in this case, a minimum-size revealing set for g might contain $i + 1$ nodes. RGR prohibits such behavior.

The *RGR dependence width* of $\langle r, i\rangle$ in $\mathcal{M}$, denoted $\text{DW}_{\text{R}}(\langle r, i\rangle, \mathcal{M})$, is defined like $\text{DW}(\langle r, i\rangle, \mathcal{M})$, except ignoring histories that do not satisfy RGR. The *RGR dependence width* of $\mathcal{M}$, denoted $\text{DW}_{\text{R}}(\mathcal{M})$, is defined analogously.

Let $\text{genvalPar}(r, i)$ be the set of genval parameters of role r that occur in the term $r(i)$. Let $\text{genvalParClr}(r, i) = \{x \in \text{genvalPar}(r, i) \mid x \text{ occurs in the clear in } r(i)\}$. Let $\text{genvalParClrBefore}(r, i)$ be the set of genval parameters of r that occur in the clear in some $r(j)$ with $j < i$.

Theorem 1 *Let $\mathcal{M} = \langle \Pi, pik\rangle$ be a system satisfying the shallow ciphertext and unsent long-term keys restrictions. Let $r \in \Pi$. If $r(i)$ is negative and contains at most one occurrence of encr, then*

$$\text{DW}_{\text{R}}(\langle r, i\rangle, \mathcal{M}) \leq \max(|\text{genvalPar}(r, i) \setminus \text{genvalParClrBefore}(r, i)|, |\text{genvalParClr}(r, i) \setminus \text{genvalParClrBefore}(r, i)| + 1) \quad (1)$$

Proof: Consider how the penetrator learns each ciphertext and genval in $\mathrm{term}(\langle s, i\rangle)$, where s is a strand for r, and sum the number of nodes involved in revealing each of them. Let $t = \mathrm{term}(\langle s, i\rangle)$. The shallow ciphertext restriction ensures that each ciphertext is directly revealed by some node. RGR implies that each genval is directly revealed by some node. The first argument of max in (1) corresponds to the case in which the penetrator learns all of the genvals in t and then performs an encryption to compute the ciphertext (if any) in t; the second argument of max corresponds to the case in which the penetrator learns the genvals that occur in the clear in t and the ciphertext in t. A genval that occurs in the clear in a term before $\langle s, i\rangle$ on s does not contribute to the RGR dependence width of t, because nodes on s are not counted in the dependence width. This justifies subtracting $\mathrm{genvalParClrBefore}(r, i)$ in (1). ∎

Theorem 1 yields the bounds on RGR dependence width in Figure 1. A simple correctness-preserving transformation was applied to some of the protocols to satisfy the "at most one ciphertext" hypothesis; specifically, each parameterized term of the form $-\{t\}_k \cdot \{t'\}_{k'}$ was replaced with the sequence $-\{t\}_k$, $-\{t'\}_{k'}$.

Generalizing Theorem 1 to apply to terms containing multiple shallow ciphertexts is not difficult. Generalizing it to eliminate the shallow ciphertext restriction is also possible, thereby completely eliminating the need for this restriction. This requires extending the proof of Theorem 1 to consider values that are revealed by sequences of decryptions applied to nested ciphertexts.

6. Reduction for Dynamic Restrictions

For a strand count f and a system $\mathcal{M}$, define a strand count $\beta(f, \mathcal{M})$ by

$$\beta(f, \mathcal{M})(r) = \max(\{\mathrm{DW}_{\mathrm{R}}(\mathcal{M}) + 1, 3\})f(r). \tag{2}$$

Theorem 2 *Let $\mathcal{M} = \langle \Pi, pik\rangle$ be a system satisfying the shallow ciphertext and unsent long-term keys restrictions. Let f be a strand count for Π. $\mathcal{M}$ satisfies BSR(f) and RGR iff all histories of $\mathcal{M}$ with strand count $\beta(f, \mathcal{M})$ do.*

Proof: (A more detailed proof is in [11].) The forward direction ($\Rightarrow$) of the "iff" follows immediately from the definitions. For the reverse direction ($\Leftarrow$), we prove the contrapositive, *i.e.*, we suppose there exists a history h of $\mathcal{M}$ that violates BSR(f) or RGR, and we construct a history of $\mathcal{M}$ with strand count at most $\beta(f, \mathcal{M})$ that violates the same property.

BSR(f) and RGR are safety properties satisfied by histories with zero nodes, so there exists a $\preceq_h$-minimal node n_0 such that (1) $\mathrm{nodesToHist}_h^{\mathcal{M}}(\mathrm{preds}_h(n_0))$ satisfies BSR(f) and RGR, and (2) $\mathrm{nodesToHist}_h^{\mathcal{M}}(\mathrm{preds}_h(n_0) \cup \{n_0\})$ violates BSR(f) or RGR.

Let $h_0 = \text{nodesToHist}_h^{\mathcal{M}}(\text{preds}_h(n_0))$. Let $s_0 = \text{strand}(n_0)$ and $i_0 = \text{index}(n_0)$. Note that $n_0 \notin \mathcal{N}_{h_0}$. For a history h' of $\mathcal{M}$ that satisfies $BSR(f)$, for a regular strand s of h', let $\text{support}_{h'}^{\mathcal{M}}(s)$ denote a support for s in h' that has strand count at most f and contains no penetrator nodes. Consider cases based on the sign of n_0.

Suppose n_0 is negative. n_0 cannot cause a violation of RGR, so n_0 causes a violation of BSR(f). Suppose $i_0 > 0$ (the proof for $i_0 = 0$ is similar). n_0 directly depends on $\langle s_0, i_0 - 1\rangle$ and on a revealing set R for $\text{term}(n_0)$ at n_0. Let

$$S_1 = \{n_0\} \cup \text{support}_{h_0}^{\mathcal{M}}(s_0) \cup \bigcup_{n \in R \setminus \text{nodes}_{h_0}(s_0)} \text{support}_{h_0}^{\mathcal{M}}(\text{strand}(n)). \quad (3)$$

h_0 satisfies RGR, so Theorem 1 implies $|R \setminus \text{nodes}_{h_0}(s_0)| \leq \text{DW}_{\text{R}}(\mathcal{M})$. h_0 satisfies BSR(f), so each support in (3) has strand count at most f. n_0 is on s_0, so it does not increase the strand count of S_1. Thus, S_1 has strand count at most $\beta(f, \mathcal{M})$. It is easy to show that S_1 supports $\{n_0\}$ in h. Lemma 1 implies that S_1 can be transformed into a history h_1 of $\mathcal{M}$ by adding penetrator nodes. It is easy to show that h_1 violates BSR(f).

Suppose n_0 is positive. n_0 cannot cause a violation of BSR(f), so n_0 causes a violation of RGR in h. Let g_0 be a genval that is "indirectly" revealed jointly by n_0 and other nodes, causing a violation of RGR. Now perform a series of case analyses based on the decryption keys that the penetrator uses to obtain g_0. In each case, one can identify a set S_1 of nodes such that S_1 has strand count at most $\beta(f, \mathcal{M})$ and $S_1 \vdash_h^{\mathcal{M}} g_0$ and $\text{origin}_h(g_0) \in S_1$. Using Lemmas 1 and 2, one can show that S_1 can be transformed into a history h_1 of $\mathcal{M}$ by adding penetrator nodes. Furthermore, h_1 violates RGR. ∎

7. Reduction for Correctness Requirements

Given a strand count f for a protocol Π, define a strand count $\text{dbl}(f)$ for Π by: $\text{dbl}(f)(r) = 2f(r)$.

Theorem 3 *Let $\mathcal{M} = \langle \Pi, pik \rangle$ be a system satisfying the shallow ciphertext and unsent long-term keys restrictions. Let f be a strand count for Π. Let ϕ be a genval secrecy or agreement requirement. Suppose all histories of $\mathcal{M}$ with strand count $\beta(f, \mathcal{M})$ satisfy BSR(f) and RGR. $\mathcal{M}$ satisfies ϕ iff all histories of $\mathcal{M}$ with strand count* $\text{dbl}(f)$ *do.*

Proof: The proof is in [11]. It is similar in strategy to the proof of Theorem 2, but simpler. ∎

8. Bounds for Sample Protocols

The right part of Figure 1 contains the maximum of the bounds obtained from Theorems 2 and 3, *i.e.*, $\max(\beta(f, \mathcal{M}), \text{dbl}(f))$.

References

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

[2] Iliano Cervesato, Nancy Durgin, Patrick Lincoln, John Mitchell, and Andre Scedrov. Relating strands and multiset rewriting for security protocol analysis. In Paul Syverson, editor, *Proc. 13th IEEE Computer Security Foundations Workshop*, pages 35–51. IEEE Press, 2000.

[3] Edmund M. Clarke, Orna Grumberg, and Somesh Jha. Verifying parameterized networks using abstractions and regular languages. In *Proc. Sixth Int'l. Conference on Concurrency Theory (CONCUR)*, 1995.

[4] James Heather and Steve Schneider. Towards automatic verification of authentication protocols on an unbounded network. In *Proc. 13th IEEE Computer Security Foundations Workshop (CSFW)*, July 2000.

[5] Leslie Lamport. Time, clocks, and the ordering of events in a distributed system. *Communications of the ACM*, 21(7):558–564, 1978.

[6] Gavin Lowe. Breaking and fixing the Needham-Schroeder public-key protocol using FDR. In *Proc. Workshop on Tools and Algorithms for The Construction and Analysis of Systems (TACAS)*, volume 1055 of *Lecture Notes in Computer Science*, pages 147–166. Springer, 1996.

[7] Gavin Lowe. Towards a completeness result for model checking of security protocols. *The Journal of Computer Security*, 7(2/3):89–146, 1999.

[8] Dave Otway and Owen Rees. Efficient and timely mutual authentication. *Operating Systems Review*, 21(1):8–10, January 1987.

[9] L. C. Paulson. The inductive approach to verifying cryptographic protocols. *The Journal of Computer Security*, 6(1/2):85–128, 1996.

[10] A. W. Roscoe and P. J. Broadfoot. Proving security protocols with model checkers by data independence techniques. *The Journal of Computer Security*, 7(2/3), 1999.

[11] Scott D. Stoller. A bound on attacks on authentication protocols. Technical Report 526, Computer Science Dept., Indiana University, July 1999. Revised April 2001. Also available at www.cs.sunysb.edu/~stoller/TR526.html .

[12] Scott D. Stoller. A bound on attacks on payment protocols. In *Proc. 16th Annual IEEE Symposium on Logic in Computer Science (LICS)*, pages 61–70. IEEE Press, June 2001.

[13] F. Javier Thayer Fábrega, Jonathan C. Herzog, and Joshua D. Guttman. Strand spaces: proving security protocols correct. *The Journal of Computer Security*, 7:191–230, 1999.

[14] Thomas Woo and Simon S. Lam. A semantic model for authentication protocols. In *Proc. 14th IEEE Symposium on Research in Security and Privacy*, pages 178–194. IEEE Press, 1993.