The Faithfulness of Abstract Protocol Analysis: Message Authentication*

Joshua D. Guttman F. Javier Thayer Lenore D. Zuck
The MITRE Corporation New York University

ABSTRACT

Dolev and Yao initiated an approach to studying cryptographic protocols which abstracts from possible problems with the cryptography so as to focus on the structural aspects of the protocol. Recent work in this framework has developed easily applicable methods to determine many security properties of protocols. A separate line of work, initiated by Bellare and Rogaway, analyzes the way specific cryptographic primitives are used in protocols. It gives asymptotic bounds on the risk of failures of secrecy or authentication.

In this paper we show how the Dolev-Yao model may be used for protocol analysis, while a further analysis gives a quantitative bound on the extent to which real cryptographic primitives may diverge from the idealized model. We develop this method where the cryptographic primitives are based on Carter-Wegman universal classes of hash functions. This choice allows us to give specific quantitative bounds rather than simply asymptotic bounds.

1. INTRODUCTION

Cryptographic protocols are simple sequences of messages that use cryptography to achieve security goals such as authentication and establishing new shared secrets. Despite their simplicity, they are often wrong, sometimes disastrously. Much work (including [5, 15, 16, 13, 21, 18, 22, 11]) has been done to develop methods to ensure their correctness, starting with Dolev and Yao [8], who represent encryption as a free operator on terms, and abstract from the mathematical properties of particular cryptographic primitives. If an attack succeeds against a protocol assuming this abstract, perfect cryptography, then the same attack will also succeed when the protocol is implemented with real cryptographic primitives. By contrast, a proof that there are no attacks, based on the assumption of abstract cryptography, will no longer be valid when concrete, less-than-perfect primitives are selected. Possibly a penetrator can manipulate the details of the cryptography to create attacks that would not succeed against abstract encryption.

Goals of this paper One form of the Dolev-Yao approach, the strand space theory, has now developed convenient methods to find what authentication and confidentiality goals a protocol achieves [11]; to determine when protocols may safely be combined [10]; to determine when type information may safely be omitted from a protocol [12]; and to generate protocols automatically to achieve given goals [19]. However, the approach relies on Dolev-Yao abstract cryptography. In this paper, we begin to adapt the strand space theory to the realities of cryptographic operators.

First, we show how to quantify the divergence between concrete cryptographic operators and traditional abstract encryption in the Dolev-Yao style, as used in a protocol, introducing the notion of ϵ-*faithfulness*. A protocol security goal, proved using abstract encryption, is ϵ-*faithful to a cryptographic primitive* if the probability that execution of the protocol—implemented using that primitive—violates the goal is $\leq \epsilon$. Establishing ϵ-faithfulness requires some stochastic assumptions. The security goals we will consider in this paper are authentication goals [24, 14, 22].

Second, for a particular primitive, we give precise, quantitative bounds on this divergence. If an attack does not succeed against a protocol with the perfect abstract cryptography of the Dolev-Yao approach, then the likelihood it succeeds against the same protocol when implemented using this cryptographic primitive is below the bound ϵ. The particular primitive we consider here is a type of message authentication code. A function is chosen (using a shared secret) from a universal class in the sense of Carter and Wegman [7, 23]; the protocol participants apply the chosen function to their messages to construct tags. The tag serves to authenticate that an attacker not privy to the shared secret has not originated the message, or altered it before delivery. We expect that our methods will also extend to some other primitives, possibly symmetric key encryption algorithms under statistical assumptions.

For Carter-Wegman tagging functions, we achieve specific bounds for a probability of failure such as $\epsilon = 2^{-32}$ (see Section 4.4). The bounds are based on parameters. One parameter is the security parameter k, which summarizes the lengths of randomly chosen values, such as keys.

*This work supported by the National Security Agency under US Army CECOM contract number DAAB07-99-C-C201. Author's affiliations: The MITRE Corporation, Bedford MA, USA, and also (for L. Zuck) New York University. Authors' email addresses: guttman,jt@mitre.org zuck@cs.nyu.edu

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
CCS'01, November 5–8, 2001, Philadelphia, Pennsylvania, USA.
Copyright 2001 ACM 1-58113-385-5/01/0011 ...$5.00.

Another parameter is the number of runs; it bounds how many guesses the penetrator may make and how many random values the regular participants must choose. In effect, this parameter dictates a re-keying schedule. Keys must be changed often enough to limit the number of sessions before re-keying, counting all sessions by non-penetrator participants.

Our main ideas We use two main ideas to achieve our goals. Both focus on the *bundle* as defined in previous work on strand spaces [22, 11], which provides a model of protocol execution. A bundle is a directed graph describing the behavior of the penetrator as well as the regular (non-penetrator) principals. The arrows represent either message transmission and reception (in which case they are written as single arrows $\rightarrow$) or the transition of a single principal through successive actions of a single session (in which case they are written as double arrows $\Rightarrow$). Bundles represent protocol execution using abstract encryption when the messages transmitted and received belong to a suitable free algebra. They represent protocol execution with particular cryptographic primitives when the messages transmitted and received are bitstrings generated using those primitives. We call bundles whose messages belong to a free algebra *abstract bundles*, while we call bundles whose messages are bitstrings *concrete bundles*.

Our first idea interrelates concrete and abstract bundles. For each concrete bundle $\mathcal{B}_c$, there is a possibly empty set $\Phi(\mathcal{B}_c)$ of corresponding abstract bundles. The correspondence Φ has the property that if there exists an abstract bundle $\mathcal{B}_a$ such that $\mathcal{B}_a \in \Phi(\mathcal{B}_c)$ and $\mathcal{B}_a$ satisfies an authentication goal, then $\mathcal{B}_c$ satisfies the same authentication goal. We find a condition on concrete bundles such that, for any concrete bundle $\mathcal{B}_c$ satisfying this condition, $\Phi(\mathcal{B}_c)$ is non-empty. Therefore, if a concrete bundle $\mathcal{B}_c$ is a counterexample to some authentication goal proved to hold of all abstract bundles, then $\mathcal{B}_c$ does not satisfy our condition.

Our second idea helps quantify the probability that $\Phi(\mathcal{B}_c) = \emptyset$ for a concrete bundle $\mathcal{B}_c$. We consider a random variable B (in the sense of probability theory) taking concrete bundles as values. We make some stochastic assumptions about B, that certain parameters of the resulting bundles are stochastically independent of each other. We also assume that certain parameters of the bundles are uniformly distributed. From these assumptions, it follows that the probability that B takes a value $\mathcal{B}_c$ such that $\Phi(\mathcal{B}_c) = \emptyset$ is less than a suitable ϵ.

These two ideas therefore bound the divergence between what may happen in concrete bundles $\mathcal{B}_c$ using the concrete cryptographic primitive, when all abstract bundles $\mathcal{B}_a$ satisfy some security goal.

Related Work Recent works by Pfitzmann et al. and Abadi and Rogaway [20, 1] have studied types of concrete cryptography that do not introduce additional attacks, beyond those predicted by the abstract protocol analysis. Or more precisely, any strategy of the penetrator has a negligible probability of producing an attack. "Negligible" is defined asymptotically in this line of work, to mean that the probability of success decreases faster than $1/p(k)$, for any polynomial p, as the security parameter k increases.

These conclusions are akin to those of Bellare and Rogaway [4], who studied protocols without abstracting from cryptography, and established security results for specific

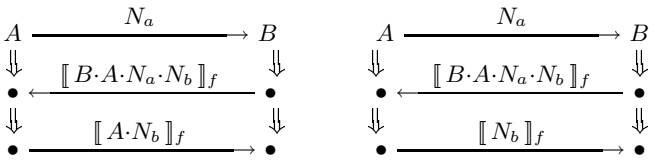

Figure 1: Intended Runs of MAP1 (left) and MAP1.1 Protocols

protocols directly from the way that specific cryptographic operators are used in them. However, the newer work of Abadi and Rogaway is a more convenient way to reach these results, although the penetrator model of [1] is limited to a passive adversary. The problem is split into a part specific to the cryptographic primitives and a separate part specific to the protocol. The protocol-specific part uses the abstract cryptography of the Dolev-Yao tradition. Similarly, Pfitzmann et al. [20] separate a cryptographic lower layer from an upper layer that applies formal methods (state machine simulation, in their approach) to protocol analysis.

The asymptotic approaches do not lead to results as specific as ours. They show only that, for any polynomial p, there exists some K_0 such that for $k \geq K_0$, the likelihood of success for the penetrator is below $1/p(k)$. They provide no way to show a key length such as $k = 128$ bits is sufficient, when the tolerance is $\epsilon = 1/(p(128))$ for a particular p.

Not all work is asymptotic [2, 3], but the current paper focuses on protocols with more multiparty interaction and has a richer penetrator model.

2. BACKGROUND

In this section, we first describe the class of "pure authentication protocols" that will be our focus in this paper, and give an example (Section 2.1). We then review the strand space ideas (Section 2.2).

2.1 Pure Authentication Protocols

The protocols that interest us in the current paper are pure authentication protocols that involve honest participants, whom we will call *regular* principals, and a penetrator. The regular principals agree on a tagging function f, shared among all of themselves, chosen from a large class of possible functions. We assume the penetrator does not know which function has been chosen.

For example, consider the protocol MAP1 of Bellare and Rogaway [4], whose intended behavior is summarized on the left in Figure 1. In this protocol, the initiator (called A here) sends in the clear a nonce (random bit string) of the form N_a to start an exchange intended for a responder (called B here). The responder B generates a fresh nonce N_b, which we assume is distinct from N_a, and responds to A's message by sending a term of the form $[\![B{\cdot}A{\cdot}N_a{\cdot}N_b]\!]_f = (B{\cdot}A{\cdot}N_a{\cdot}N_b) \cdot f(B{\cdot}A{\cdot}N_a{\cdot}N_b)$. Since f is unknown to the penetrator, the value $f(B{\cdot}A{\cdot}N_a{\cdot}N_b)$ is intended to serve as a signature, guaranteeing the integrity of the message to the recipient. When the A receives $[\![B{\cdot}A{\cdot}N_a{\cdot}N_b]\!]_f$, it responds with $[\![A{\cdot}N_b]\!]_f$, thereby assuring B that the value N_b has been received by A. Again, $[\![A{\cdot}N_b]\!]_f$ is really a concatenation $(A{\cdot}N_b) \cdot f(A{\cdot}N_b)$.

MAP1 is a pure authentication protocol: If A has had a run with intended respondent B, then B has undertaken at

least the first two steps of a run with intended initiator A, and the runs agree on the nonces N_a, N_b. Conversely, if B has had a run with intended initiator A, then A has had a run with intended respondent B, and the runs agree on the nonces N_a, N_b.

The protocols we consider here do not have the goal of causing the participants to agree on any new secret. Of course, preserving the secrecy of the choice of f is necessary. However, if the secrecy of f fails, then the authentication goals will also fail. Hence, we will not need to treat secrecy goals directly.

Authentication goals require some freshness assumptions, or as we call them, origination assumptions. For instance, nonces should not be reused. In MAP1, if B reuses the nonce N_b, then the penetrator can save $[\![A{\cdot}N_b]\!]_f$, start sessions purporting to be A, and complete the run as soon as B reuses N_b. We assume that in a bundle involving the value N_b, there will be just one point in one session at which N_b originates. By *origination*, we mean a message transmission in which N_b is sent without having been received previously in that session.

Using the authentication test method of [11], we can easily show that MAP1 achieves its authentication goals. Indeed, we may wonder about fine points, such as whether A's name is needed in the last message. Again using the same methods, we can show that the answer is no, and that the modified protocol MAP1.1, shown in Figure 1 on the right, which omits A's name from the last message, achieves the same authentication goals for essentially the same reasons.

2.2 Strand Spaces

We very briefly summarize the ideas behind the strand space model [22, 11]; see also Appendix A. Let A be a set of messages that can be sent between principals; we are interested in various choices of A. For each choice of A, we assume that there is a subterm relation, written $t \sqsubset t'$.

A *strand* is a sequence of message transmissions and receptions, where transmission of a term t is represented as $+t$ and reception of term t is represented as $-t$. Each vertical column in Figure 1 shows a strand, assuming that particular values are chosen for the parameters A, B, N_a, and N_b. A strand element is called a *node*. A *strand space* Σ is a set of strands. (See Definition A.1.)

If s is a strand, $\langle s, i\rangle$ is the i^{th} node on s. The relation $n \Rightarrow n'$ holds between nodes n and n' if $n = \langle s, i\rangle$ and $n' = \langle s, i+1\rangle$. The relation $n \rightarrow n'$ represents inter-strand communication; it means that $\mathrm{term}(n_1) = +t$ and node $\mathrm{term}(n_2) = -t$. The two relations $\Rightarrow$ and $\rightarrow$ jointly impose a graph structure on the nodes of Σ. The vertices of this graph are the nodes, and the edges are the union of $\Rightarrow$ and $\rightarrow$.

A term t *originates* at a node $n = \langle s, i\rangle$ if the sign of n is positive; $t \sqsubset \mathrm{term}(n)$; and $t \not\sqsubset \mathrm{term}(\langle s, i'\rangle)$ for every $i' < i$. Thus, n represents a message transmission that includes t, and it is the first node in s including t. If a value originates on only one node in the strand space, we call it *uniquely originating*; uniquely originating values are desirable as nonces. (See Definition A.2.)

A *bundle* is a causally well-founded collection of nodes and arrows of both kinds. In a bundle, when a strand receives a message m, there is a unique node transmitting m from which the message was received. By contrast, when a strand transmits m, many strands (or none) may receive m. (See Definition A.3.)

A strand represents the local view of a participant in a run of a protocol. For a legitimate participant, it represents the messages that participant would send or receive as part of one particular run of his side of the protocol. We call a strand representing a legitimate participant a *regular* strand. Typically, the regular strands of Σ are the instances of a finite number of parameterized strands (See Section 3.1.)

For the penetrator, the strand represents an atomic deduction. More complex actions can be formed by connecting several penetrator strands. While regular principals are represented only by what they say and hear, the behavior of the penetrator is represented more explicitly, because the values he deduces are treated as if they had been said publicly. We partition penetrator strands according to the operations they exemplify. C-strands and S-strands concatenate and separate terms, respectively; K-strands emit keys from a set of known keys; and M-strands emit known atomic texts or guesses. In protocols which use a genuine encryption operator, E-strands encrypt when given a key and a plaintext; D-strands decrypt when given a decryption key and matching ciphertext. (See Definition A.6.) We will adapt the E-strands and D-strands below to reflect our current interest in pure authentication protocols using tagging.

As an example of an authentication goal, consider the responder's guarantee in MAP1. Suppose that the responder B has a run apparently with A, using the nonces N_a and N_b. B may assume that the nonce N_b is uniquely originating, because he generates it himself using highly random methods. B's authentication guarantee is the implication:

$\mathfrak{A}_B$: **if** N_b is uniquely originating, **then** A has had a matching run apparently with B, using the nonces N_a and N_b.

3. PROTOCOLS AND THEIR IMPLEMENTATIONS

We turn now to the questions how to represent protocols in the strand space theory (Section 3.1), and what it means to implement protocols using concrete primitives or abstract messages. We talk about algebras of bitstrings in Section 3.2, and relate them to abstract (free) message algebras in Section 3.3.

3.1 Representing Protocols in Strand Spaces

A protocol requires regular participants to play a number of different roles, such as initiator, responder, or key server. The protocol itself consists of a number of *schematic strands*, one for each role played by the regular principals. These schematic strands may be determined by programs executed by the principals against their local state; our concern is exclusively with the resulting behaviors.

A schematic strand consists of a parameter list $X_1, \ldots, X_n$, together with a sequence of a fixed number of signed schematic terms in which the parameters may occur. A signed schematic term, in turn, is $+$ or $-$ together with a term in which some parts have been replaced by parameters X_i. For instance, the schematic strand MAP1Init$[A, B, N_a, N_b]$ that has parameters A, B, N_a, N_b and signed terms

$$\langle +N_a,\ -[\![B{\cdot}A{\cdot}N_a{\cdot}N_b]\!]_f,\ +[\![A{\cdot}N_b]\!]_f \rangle$$

defines the MAP1 initiator's behavior. The responder's behavior MAP1Resp$[A, B, N_a, N_b]$ is the complementary sche-

matic strand with behavior

$$\langle -N_a,\ +[\![\, B{\cdot}A{\cdot}N_a{\cdot}N_b \,]\!]_f,\ -[\![\, A{\cdot}N_b \,]\!]_f \rangle.$$

The parameters A, B range over names, while the parameters N_a, N_b range over nonces. No parameter here ranges over concatenated terms such as $A{\cdot}N_a$.

Given some particular algebra of messages A, we may *instantiate* a schematic strand by choosing suitable values from A for the parameters $X_1, \ldots, X_n$. The result is a strand. The messages sent and received are the results of filling in these values in place of the parameters in the successive signed schematic terms.

We identify a protocol with the set of schematic strands which specify it. A protocol may also have parameters. In MAP1, the shared secret f is a parameter of the protocol itself; given a value for f, all of the regular participants use that value. That is why f is not listed as a parameter of the schematic strands. Thus, MAP1, acting with shared secret f, as the set with two parametric strands, $\Pi_f = \{\text{MAP1Init}[A, B, N_a, N_b], \text{MAP1Resp}[A, B, N_a, N_b]\}$.

Given a message algebra A, a protocol Π determines a strand space Σ, which we call the *strand space generated by* Π *over* A. The instances of a schematic strand are all behaviors resulting from choosing values in A of appropriate type for each parameter. The strand space Σ contains, as its regular strands, instantiations of each schematic strands with each appropriate value, for instance all bitstrings of the correct length for a nonce N_a, and all properly formed domain names or IP addresses for a parameter ranging over names. In MAP1, no strand can be an instance of both schematic strands, because the patterns of $+$ and $-$ terms are different, and this is effectively always the case.

There are two types of message algebra A that specially interest us, each of which generates a strand space Σ from a Π. First, there are free algebras, in which $[\![\, A{\cdot}N_b \,]\!]_f$ is a term distinct from any constructed in a different way. Second, there are algebras consisting of bitstrings, in which concatenation is an operator (possibly a partially defined operator) producing bitstrings from bitstrings. Likewise, the tagging operator produces particular bitstrings when given bitstrings as arguments, and it has collisions, i.e. cases in which different messages yield the same tag.

In MAP1, the parameters range only over names and nonces, not over concatenated or tagged terms. This is the case for all (natural) pure authentication protocols, so we will assume it throughout the remainder of the paper. The assumption would not hold for other protocols, particularly shared-key protocols using a key server, such as Otway-Rees [17] or Carlsen [6]; see [11, Section 5.1.3] for an explanation.

3.2 Implementing Protocols with Bitstrings

We consider first schemes that may be used to encode messages via bitstrings.

An Example An atom consists of one letter followed by a string of hexadecimal digits. The letter indicates its intended use. Names or addresses begin with a. Randomly chosen nonces begin with n; the set of such atoms is N. Tags for verifying integrity begin with v; the set of such atoms is V. Concatenations are s-expressions in the style of Lisp. Two terms t_0 and t_1 are concatenated to form the string '$(t_0\ .\ t_1)$'.

This is an unambiguous encoding, since it is always clear whether a string represents an atom or a concatenation, and if it is a concatenation, where each of the two arguments begin and end. Every message is built from atoms by a finite number of concatenations. The result of concatenation may not always be a valid message. If the total number of characters exceeds some maximum, then the message may be rejected because it overflows the receiver's input buffer. If the depth of nested parentheses exceeds some maximum, then it may be rejected because parsing it requires too large a stack.

Tagging functions, by contrast, because they have collisions. The output bitstrings are tags in V, typically of limited length, and the inputs may be bitstrings of arbitrary length. However, in the case of MAP1, each tag immediately follows the message body it is meant to validate. Given a message body such as $t = B{\cdot}A{\cdot}N_a{\cdot}N_b$, the recipient knows that the next component must be $f(t)$. Thus, if tags in V occur only in the context $t{\cdot}f(t)$, then there is never any ambiguity about the body to which the tag applies, and every occurrence of a tag contributes to representing an authenticated message $[\![\, t \,]\!]_f$ with no choice about what t is being tagged. Tags occur nowhere else.

Rigid Schemes In a scheme such as the one we have just described, any bitstring received by a principal can be interpreted as a protocol message in at most one way, and any message sent can be constructed in at most one way, give the parameters selected. Thus, a principal always knows uniquely what strand parameters are compatible with the bitstrings it has received and sent. We say that a scheme for encoding messages is *rigid* for a protocol when it has this property.

A *rigid scheme* for a protocol Π_f consists three ingredients: a set of bitstrings M, a concatenation function $\cdot$, and a set $\mathcal{F}$ of possible tagging functions (where $f \in \mathcal{F}$). We again refer to the tags as V, and require that all $f \in \mathcal{F}$ have type $f\colon M \to \mathsf{V}$. The *atoms* of the scheme, written $\mathsf{atom}(M)$, are all values $x \in M$ such that x is not of the form $t_0{\cdot}t_1$ for any $t_0, t_1 \in M$; we require that $\mathsf{V} \subset \mathsf{atom}(M)$. We assume that a bitstring in M can be a concatenation in at most one way, and that every member of M may be built from atoms by a finite sequence of concatenations.

We also require that tags $v \in \mathsf{V}$ occur only in the form $t{\cdot}v$, where $v = f(t)$, in messages of Π_f. Thus, tags contribute only to tagging messages $[\![\, t \,]\!]_f$.

DEFINITION 3.1. *If $(M, \cdot, \mathcal{F})$ is a rigid scheme, then the subterm relation for it, written $t_0 \sqsubset t_1$, is the smallest reflexive, transitive relation such that $t \sqsubset t{\cdot}t'$ and $t' \sqsubset t{\cdot}t'$.*

A bundle, whose messages are encoded as bitstrings using a rigid scheme, will be called a *concrete bundle*, and usually denoted by $\mathcal{B}_c$.

Given a protocol under a rigid scheme and a bundle $\mathcal{B}_c$, each regular strand in $\mathcal{B}_c$ has a unique set of possible parameters, which are names (of participants) or nonces. Thus, parameters are in $\mathsf{atom}(M) \setminus \mathsf{V}$, the set of atoms that are not tags.

Penetrator Strands In the concrete model, the penetrator can do anything. The penetrator can choose any bitstring to deliver, or given a number of bitstrings, can apply any function g to them to determine a new bitstring to deliver. Thus any strand of the form $\langle -x_1 \Rightarrow \cdots - x_n \Rightarrow$

$+g(x_1, \ldots, x_n)\rangle$ is a penetrator strand, which we call a g-strand.

In case $n = 0$, a g-strand amounts to guessing a constant value $g()$ independent of input; for instance, the penetrator may choose any pair $a \cdot v$ to deliver when a tagged value is needed. Indeed, he may choose $a \cdot v$ for $v = f(a)$ without knowing that he did so, and may thus apply f-strands unknowingly.

3.3 Free Message Algebras

Given a rigid scheme for a protocol Π_f, we define an associated abstract (free) encryption algebra. We regard the set of tagging functions as if they were keys, because they are a shared secret. Being functions, though, these "keys" are never transmitted as part of a message belonging to the protocol.

DEFINITION 3.2. *Let $(M, \cdot, \mathcal{F})$ be a rigid scheme for Π_f. The algebra E of* abstract tagging *over $(M, \cdot, \mathcal{F})$ is freely generated from:*

- *two sets: texts in $\mathsf{atom}(M) \setminus \mathsf{V}$ and "keys" in $\mathcal{F}$,*
- *via two operations: concatenation $\cdot_{\mathsf{E}}$ and tagging $[\![t]\!]_f$ for $t \in \mathsf{E}$ and $f \in \mathcal{F}$.*

If the protocol Π_f consists of a set of parameterized strands $\mathrm{Role}_i[X_1, \ldots, X_{n_i}]$, and the parameters X_j range only over atoms in $\mathsf{atom}(M) \setminus \mathsf{V}$, then we can regard it as determining messages in either M or E. We write Role_i^M or $\mathrm{Role}_i^{\mathsf{E}}$ when we want to distinguish them.

What protection is offered by tagging? Although only someone possessing f can create $[\![h]\!]_f$ from h, anyone can extract h from $[\![h]\!]_f$. Thus, we adapt the penetrator strands shown in Definition A.6 slightly, replacing the decryption penetrator strand with the untagging strand shown here, and updating the encryption strand to our tagging notation:

$\mathsf{E}_{h,f}$ Encryption: $\langle -f,\ -h,\ +[\![h]\!]_f \rangle$

U_h Untagging: $\langle -[\![h]\!]_f,\ +h \rangle$

We refer to these strands and the remaining M, K, C, and S strands from Definition A.6 as *abstract penetrator strands.*

We are interested in the case where the protocol Π_f is executed using a secret tagging function hidden from the penetrator, so we assume that $f \notin \mathsf{K}_{\mathcal{P}}$, the set of keys initially available to the penetrator.

3.4 Bundle Abstraction

Suppose that we have a pure authentication protocol $\Pi = \{\mathrm{Role}_i[X_1, \ldots, X_{n_i}] : 1 \leq i \leq n\}$, implemented using a rigid scheme $(M, \cdot, \mathcal{F})$. Let $\mathcal{B}_c$ be a concrete bundle, and suppose s is a regular strand with some nodes occurring in $\mathcal{B}_c$. Possibly only an initial segment of s is in $\mathcal{B}_c$. We say the $\mathcal{B}$-*height* of a strand is the number of nodes of s in $\mathcal{B}$. If s has no nodes in $\mathcal{B}$, then its $\mathcal{B}$-height is 0.

Since the operations yield bitstrings as determined by $(M, \cdot, \mathcal{F})$, the messages sent and received in s are particular bitstrings in M. From Section 3.2, we know that there is a unique parameterization of s as some $\mathrm{Role}_i^M[\vec{a}]$, and the parameters $\vec{a}$ are atoms of M which are not tags. Therefore, these parameters are also atoms of E, the algebra of abstract tagging corresponding to $(M, \cdot, \mathcal{F})$, and there is also an abstract strand $s' = \mathrm{Role}_i^{\mathsf{E}}[\vec{a}]$, in which the same parameters determine abstract terms using the free algebra. The *abstract skeleton* of $\mathcal{B}_c$ is the result of transforming each regular strand s of $\mathcal{B}_c$ in this way, annotating each resulting strand with the $\mathcal{B}_c$-height of s.

DEFINITION 3.3. *The* abstract skeleton *of $\mathcal{B}_c$, which we write $\mathsf{skel}(\mathcal{B}_c)$, is the set of pairs (s', h) where $s' = \mathrm{Role}_i^{\mathsf{E}}[\vec{a}]$ and $h > 0$ is the $\mathcal{B}_c$-height of $s = \mathrm{Role}_i^M[\vec{a}]$.*

The abstract skeleton $\mathsf{skel}(\mathcal{B}_c)$ is not an abstract bundle; it is simply a set of regular strands annotated with heights. We also sometimes regard it as a set of nodes, namely the first h nodes on s when (s', h) is in $\mathsf{skel}(\mathcal{B}_c)$.

Although $\mathsf{skel}(\mathcal{B}_c)$ is not a bundle, we may be able to turn it into a bundle by adding abstract penetrator strands and connecting message transmissions and receptions using arrows $\rightarrow$. There may be multiple ways to do so. Alternatively, if the penetrator exploited something peculiar in the way the bitstrings worked out in $\mathcal{B}_c$, it may be impossible to mimic this via abstract penetrator strands. We then regard $\mathcal{B}_c$ as having been a lucky outcome from the penetrator's point of view.

DEFINITION 3.4. *$\Phi(\mathcal{B}_c)$ is the set of all abstract bundles $\mathcal{B}_a$ such that for regular strands s', the $\mathcal{B}_a$-height of $s' = h$ and $h > 0$ if and only if $(s', h) \in \mathsf{skel}(\mathcal{B}_c)$.*

If $\Phi(\mathcal{B}_c) = \emptyset$, then $\mathcal{B}_c$ is a lucky strike.

A lucky strike is a concrete bundle that is inexplicable, relative to the abstract model of the powers of the penetrator. The penetrator either guessed or did something specific to the way that concatenation and tagging interact with the bitstrings in M.

3.5 Lucky Strikes and Forgeries

There is only one way that a lucky strike can occur: The penetrator selects a tag $v \in \mathsf{V}$, and delivers $t \cdot v$ to a regular participant, who verifies that $v = f(t)$. We call this a *forgery.* Of course, it is anomalous only if no regular participant has previously sent $t \cdot v$. We understand *previously* by the bundle partial ordering $\preceq_{\mathcal{B}}$ (Definition A.4) according to which $n_0 \preceq_{\mathcal{B}} n_1$ if there is a path of zero or more arrows $\rightarrow$ and $\Rightarrow$ in $\mathcal{B}$ leading from n_0 to n_1.

DEFINITION 3.5. *A* forgery *is a negative regular node $n_1 \in \mathcal{B}_c$ such that $t \cdot f(t) \sqsubset \mathit{term}(n_1)$ and there is no positive regular node $n_0 \in \mathcal{B}_c$ such that $n_0 \preceq_{\mathcal{B}_c} n_1$ and $t \cdot f(t) \sqsubset \mathit{term}(n_0)$.*

In the Introduction we mentioned the need for a property that ensures that $\Phi(\mathcal{B}_c)$ is non-empty. This is the property of containing no forgeries.

PROPOSITION 3.6. *If $\mathcal{B}_c$ is a lucky strike, then there exists a forgery $n_1 \in \mathcal{B}_c$.*

PROOF. Suppose that there is no forgery in $\mathcal{B}_c$; we show that $\mathcal{B}_c$ is not a lucky strike by building an abstract bundle $\mathcal{B}_a$ from $\mathsf{skel}(\mathcal{B}_c)$. We do so by starting with the empty abstract bundle $\mathcal{B}_0$. Inductively we define a sequence of abstract bundles; at each step $\mathcal{B}_{i+1}$ has one new regular node, together with 0 or more penetrator nodes and new arrows as needed.

If all of the nodes in $\mathsf{skel}(\mathcal{B}_c)$ have been used, then we have constructed $\mathcal{B}_a$. Otherwise, let n_c be a regular node in $\mathcal{B}_c$ that is $\preceq$-minimal (in the precedence ordering for $\mathcal{B}_c$) among nodes not yet used, and let n_a be the corresponding

node in $\mathsf{skel}(\mathcal{B}_c)$. $\mathcal{B}_{i+1}$ will contain n_a. To satisfy the bundle definition, we must construct $\mathrm{term}(n_a)$ from nodes already in $\mathcal{B}_i$ together with new penetrator nodes if needed. Using C-strands, we can build $\mathrm{term}(n_a)$ if given all of its atomic components and all of its tagged components $[\![t]\!]_f$.

If m is an atomic component of $\mathrm{term}(n_a)$, then as noted before Definition 3.2, m is not a key $f \in \mathcal{F}$; keys are not transmitted in these protocols. Thus, we may add an M-strand initiating m. If $[\![t]\!]_f$ is a tagged component, then by the assumption that there is no forgery in $\mathcal{B}_c$, some regular node $n_0 \preceq n_c$ emits the corresponding bitstring. By the $\preceq$-minimality of n_c, the regular node corresponding to n_0 is already in $\mathcal{B}_i$.

Thus, in all cases, we may construct $\mathcal{B}_{i+1}$. By the finiteness of $\mathcal{B}_c$ this process must terminate with all of the nodes in $\mathsf{skel}(\mathcal{B}_c)$ used. ■

By this proposition, if an authentication property holds of all abstract bundles, then a concrete bundle is not a counterexample unless it has a forgery. To quantify the divergence between possible concrete behaviors and the abstract, Dolev-Yao model, and to prove ϵ-faithfulness in the sense of our Introduction, we must show that the probability of a bundle having a forgery is $\leq \epsilon$. We show how to do this in Section 4.3.

There is another reason why authentication may fail, apart from forgery. An authentication theorem such as $\mathfrak{A}_B$ in Section 2.2 states an implication: *if* a regular participant chooses uniquely originating nonces, *then* its peer has engaged in a matching strand. However, two regular principals could choose the same nonce. Then the conclusion might not be true.

In Section 4.4, we bound the probability of this event. We combine the bound on the likelihood of forgery with the bound on the likelihood of nonce collision to infer an overall bound on the risk of authentication failure, assuming the protocol is correct in the abstract Dolev-Yao model.

4. STOCHASTIC MODEL

To bound the probability that a concrete bundle contains a forgery, we need a stochastic model of protocol behavior. This model consists of an underlying probability space, together with some random variables[1] that extract aspects of the behavior. We must assume some constraints, which require either that a random variable is uniformly distributed, or else that random variables are independent of one another.

We call the probability space (Ω, P). For convenience, we assume that it is finite, as we may do because the sets of messages (bitstrings of bounded size) are finite and the size of the bundles of interest are bounded. (Ω, P) encapsulates an array of information including the choice of nonces and of interlocutors by the regular participants.

We assume that the penetrator has some strategy. It determines his behavior as a function of two arguments, namely first what he observes of the regular participants and second some random choices determined by the probability space. The strategy determines the behavior of the penetrator, including his choices about what genuine messages to deliver, and especially what messages and tags to try as forgeries. A protocol implementation is *ϵ-faithful* to an authentication goal $\mathfrak{A}$ if the ϵ bounds the probability of the event that a bundle is chosen in which $\mathfrak{A}$ fails.

[1]A random variable (sometimes we write just "variable") is a function on the underlying probability space.

4.1 Random Variables

Each choice of $\omega \in \Omega$ determines a bundle $B(\omega)$. We assume that the regular strands of $B(\omega)$ are ordered in some arbitrary way, so that $S_i(\omega)$ enumerates $\mathsf{skel}(B(\omega))$, as a function of i. The model focuses on 4 random variables.

1. *The random variable $F\colon \Omega \to \mathcal{F}$ determines the secret tagging* function f.

2. *The random variable $R\colon \Omega \to \{0,1\}^*$ is the penetrator's source of* randomness.

3. *The random variable $N\colon \Omega \to (\mathbb{N} \times \mathbb{N}) \to \mathsf{N}$ determines, given integers i and j, the j^{th}* nonce *chosen to originate on the i^{th} regular strand $S_i(\omega)$.*

4. *The random variable $T\colon \Omega \to (\mathbb{N} \times \mathbb{N}) \to (M \times \mathsf{V})$ determines, given integers i and j, the j^{th}* tagged *value $t \cdot f(t)$ sent by the i^{th} regular strand $S_i(\omega)$.*

In MAP1, regular strands use a single nonce, so $N(\omega)(i,j)$ is defined only when $j = 1$. Likewise, they send a single tagged message each, so $T(\omega)(i,j)$ is also defined only when $j = 1$. T is certainly not independent of N, since a fresh nonce is part of each tagged message sent by a regular strand. Likewise, T is not independent of F, because the values T delivers are pairs $\langle t, f(t) \rangle$. However, we assume below that this is the only dependence of T on F, i.e. that the values of the first components t are independent of F (Assumption 2).

The key aspect of penetrator behavior is the set of forgery attempts. We formalize the penetrator strategy by a function $G(T(\omega), R(\omega))$ which, given the signed messages sent by the regular participants and the penetrator's randomness, returns a set of pairs $(b, v) \in M \times \mathsf{V}$. The penetrator's forgery attempts in $B(\omega)$ are the messages $b \cdot v$ for $(b, v) \in G(T(\omega), R(\omega))$. For these to be forgeries rather than replays, the messages b must be different from those sent by the regular participants, so we assume that there is no message b and tags v, v' such that $(b, v) \in G(T(\omega), R(\omega))$ and $(b, v') = T(\omega)(i,j)$.

$B(\omega)$ has a successful forgery if a value in $G(T(\omega), R(\omega))$ has the form $b \cdot f(b)$ where $f = F(\omega)$. Thus, we are interested in the event

$$\mathsf{forge} = \{\omega\colon \text{for some } (b, v) \in G(t, r), v = F(\omega)(b)\}$$

whose probability we want to show is small, conditional on any particular values $T(\omega) = t$ and $R(\omega) = r$.

4.2 Model Assumptions

We need to make assumptions that some of the variables are independent and that some are uniformly distributed. We regard $N(\omega)(i,j)$ as a family of nonce-valued random variables indexed by i and j.

ASSUMPTION 1. *Any two different variables $N(\omega)(i,j)$ and $N(\omega)(i',j')$ are independent, and each variable $N(\omega)(i,j)$ is uniformly distributed.*

This assumption is used only in Section 4.4.

Let fst be the function that delivers the first component of a pair, so that $\mathsf{fst} \circ T$ is the function that delivers the bodies but not the tags of the tagged messages sent by regular participants.

ASSUMPTION 2. *The variable F is stochastically independent of R and $T' = \mathsf{fst} \circ T$ taken jointly:*

$$\mathsf{P}\{F(\omega) = f \wedge T'(\omega) = x \wedge R(\omega) = r\} = \mathsf{P}\{F(\omega) = f\} \cdot \mathsf{P}\{T'(\omega) = x \wedge R(\omega) = r\}$$

Hence, F and R are pairwise independent, as are F and T'.

ASSUMPTION 3. *The distribution of F on $\mathcal{F}$ is uniform, that is, for any $E \subseteq \mathcal{F}$*

$$\mathsf{P}\{F(\omega) \in E\} = \frac{\mathsf{card}\, E}{\mathsf{card}\, \mathcal{F}}$$

The protocol limits the number of new nonces sent by a single regular strand. It also limits the number of signed expressions sent by a single regular strand. And it limits the number of signed expressions that can be received by a single regular strand. In MAP1, all of these numbers equal 1, though in another protocol they may have some maximum ν. Therefore, if a bundle $\mathcal{B}_c$ has at most Γ many regular strands, the risk of two strands re-using a nonce is limited because only $\Lambda = \nu$ times Γ nonces are used. The number of samples of the tagging function f that the regular participants show the penetrator is limited by Λ. And the number of forgeries that the penetrator may submit to the regular participants is bounded by Λ.

ASSUMPTION 4. *The number of nonces, tagged values sent, and tagged values received on regular nodes in $B(\omega)$ is bounded by some value Λ.*

This assumption is part of the justification for taking Ω to be finite. The restriction to no more than Γ many regular strands is ultimately justified by a re-keying schedule. We require the participants to agree on a new value of f before Γ many sessions can have occurred.

4.3 The Probability of Forgery

Given a particular ω, the penetrator may observe $t = T(\omega)$ and $r = R(\omega)$, the first being the tagged messages chosen by the regular participants and the second being the penetrator's source of randomness. The penetrator uses G to choose forgery attempts, so we must bound $\mathsf{P}(\mathsf{forge} \mid T(\omega) = t \wedge R(\omega) = r)$, i.e.

$$\mathsf{P}\{\text{for some } (b, v) \in G(t, r), v = F(\omega)(b) \mid T(\omega) = t \wedge R(\omega) = r\}$$

The penetrator, having observed the regular participants sending the signed messages in $T(\omega)$, can exclude some tagging functions $f \in \mathcal{F}$, because they are incompatible with a signed message $t \cdot v = T(\omega)(i, j)$. We refer to the set of remaining candidates as the part of $\mathcal{F}$ compatible with ω, or $\mathcal{F}_\omega$, where

$$\mathcal{F}_\omega = \{g \in \mathcal{F} \colon \forall i, j, t, v \,.\, T(\omega)(i, j) = (t, v) \text{ implies } g(t) = v\}$$

Calculation using Assumptions 2 and 3 yields

$$\mathsf{P}(\mathsf{forge} \mid T(\omega) = t \wedge R(\omega) = r) = \frac{\mathsf{card}(\{g \in \mathcal{F}_\omega \,:\, \exists (b, v) \in G(t, r) \,,\, g(b) = v\})}{\mathsf{card}(\mathcal{F}_\omega)}$$

which in turn equals $\mathsf{P}(\{g \in \mathcal{F}_\omega \,:\, \exists (b, v) \in G(t, r) \,.\, g(b) = v\})$, by uniformity.

Suppose now that the set of tagging functions $\mathcal{F}$ is a universal class, following the classic papers by Carter and Wegman [7, 23]. We define the notion in the form:

DEFINITION 4.1. *A set of functions $\mathcal{F} \subseteq Y^X$ is n-strongly universal just in case the following two conditions are met: (1) $\mathsf{card}(X)$ is at least n, and (2) if $x_1, \ldots, x_n$ are any n pairwise distinct values in X, then the distribution of the evaluation mapping $f \mapsto \langle f(x_1), \ldots, f(x_n)\rangle$ is uniformly distributed.*

In Appendix B we give an example of an n-strongly universal class (Example B.4), and derive a key lemma (Lemma B.7):

LEMMA 4.2. *If $\mathcal{F} \subseteq Y^X$ is n-strongly universal then $\mathcal{F}$, then for any $\ell \leq n$ and $x_1, \ldots, x_\ell \in X$, and any $y_1, \ldots, y_\ell \in Y$,*

$$\mathsf{P}\{f \in \mathcal{F} : \exists i \leq \ell \,.\, f(x_i) = y_i\} \leq \frac{\ell}{\mathsf{card}\, Y}.$$

Observe that if $\mathcal{F}$ is n-universal and $T(\omega)$ provides at most m tagged messages, then $\mathcal{F}_\omega$ is $(n-m)$-universal. We therefore take $\mathcal{F}$ to be $(2\,\Lambda)$-universal and apply Lemma 4.2, instantiating $\mathcal{F}$ with $\mathcal{F}_\omega$, and observing that $\ell \leq \Lambda$. This last inequality is justified because Λ bounds the number of forgery attempts ℓ. Thus,

$$\mathsf{P}(\mathsf{forge}) \leq \frac{\Lambda}{\mathsf{card}(\mathsf{V})}.$$

4.4 Likelihood of Anomalies

In the analysis of bundles by the abstract bundle representation theorem (Proposition 3.6), there are two events whose likelihood we would like to bound. Either could cause a failure of the conclusion that the bundle contains a matching strand, as in the authentication goal $\mathfrak{A}_B$ of Section 2.2. One is forge; the other is that the regular participants choose clashing nonces, which we define:

$$\mathsf{clash} = \{\omega \colon N(\omega)(i, j) = N(\omega)(i', j') \text{ where } i \neq i' \text{ or } j \neq j'\}$$

Since by Assumption 1 the random variables $N(\omega)(i, j)$ are uniformly distributed and mutually independent, determining a bound on the likelihood of a nonce anomaly is a special case of the "birthday problem" [9]. The total number of choices is bounded by Λ, so the likelihood of at least one collision is bounded above by $\Lambda(\Lambda - 1)/2\ \mathsf{card}(\mathsf{N})$.

As an example, consider a tolerance of $\epsilon = 2^{-32}$ for the likelihood of forgeries and clashes together, where we will allocate half of ϵ for each type of anomaly. If nonces are given by 64-bit strings, then $\mathsf{card}(\mathsf{N}) = 2^{64}$. To ensure that independent choices of Λ nonces has probability of anomaly below $\epsilon/2$, it suffices to restrict Λ so that $\Lambda^2/2{\cdot}2^{64} \leq 2^{-33}$, i.e., $\Lambda \leq 2^{16} = 65,536$. If, for example, we would like to use a shared secret choice of f without change for a year, this would allow 175 strands per day, since $65,000/365 > 175$.

For the case of forgeries, $\mathsf{P}(\mathsf{forge}) \leq \Lambda/\,\mathsf{card}\,\mathsf{V}$. We must use Carter-Wegman hash functions which are $(2\,\Lambda)$-strongly universal with $\Lambda = 2^{16}$ as before, i.e. 2^{17} − *universal*. To ensure that $\Lambda/\,\mathsf{card}\,\mathsf{V} \leq 2^{-33}$, we need $\mathsf{card}\,\mathsf{V} \geq 2^{49}$, so that 64-bit tags are ample.

Thus with $\Lambda = 2^{16}$, the likelihood of an authentication failure is

$$\epsilon \leq \mathsf{P}(\mathsf{forge}) + \mathsf{P}(\mathsf{clash}) \leq 2^{-33} + 2^{-33} \leq 2^{-32}$$

Each tag calculation requires substantial computation, but the rekeying is infrequent and the risk of authentication failure is very low. These numbers are only illustrative; the

point is that we have described a comprehensive method that yokes abstract protocol design and verification using strand spaces to low-level calculations of the risk of security compromise.

5. CONCLUSION

In this paper we have shown that abstract encryption is faithful in the sense that, when a protocol meets its security goals in an abstract model like the strand space model, then the probability that a penetrator can defeat it is below a suitable ϵ such as 2^{-32}. Specifically, we have established this in the case in which the cryptographic primitive is Carter-Wegman hashing; the protocol uses a single secret shared among all participants; and the implementation of the protocol is rigid in the sense of Section 3.2. It is likely that the restriction to a single shared secret is unnecessary. It is also likely that some other types of cryptography lead to analogous results.

6. REFERENCES

[1] Martín Abadi and Phillip Rogaway. Reconciling two views of cryptography (the computational soundness of formal encryption). In *IFIP International Conference on Theoretical Computer Science (IFIP TCS2000)*, Lecture Notes in Computer Science. Springer-Verlag, 2000.

[2] Mihir Bellare. Practice-oriented provable security. In E. Okamoto, G. Davida, and M. Mambo, editors, *First International Workshop on Information Security (ISW 97)*, volume 1396 of *LNCS*. Springer Verlag, 1998.

[3] Mihir Bellare, J. Killian, and Phillip Rogaway. The security of cipher block chaining. In Yvo Desmedt, editor, *Advances in Cryptology — Crypto 94*, volume 839 of *LNCS*. Springer Verlag, 1994.

[4] Mihir Bellare and Phillip Rogaway. Entity authentication and key distribution. In *Advances in Cryptology – Crypto '93 Proceedings*. Springer-Verlag, 1993. Full version available at http://www-cse.ucsd.edu/users/mihir/papers/eakd.ps.

[5] Michael Burrows, Martín Abadi, and Roger Needham. A logic of authentication. *Proceedings of the Royal Society*, Series A, 426(1871):233–271, December 1989. Also appeared as SRC Research Report 39 and, in a shortened form, in ACM Transactions on Computer Systems 8, 1 (February 1990), 18-36.

[6] Ulf Carlsen. Optimal privacy and authentication on a portable communications system. *Operating Systems Review*, 28(3):16–23, 1994.

[7] J. Lawrence Carter and Mark N. Wegman. Universal classes of hash functions. *Journal of Computer and System Sciences*, 18:143–54, 1979.

[8] D. Dolev and A. Yao. On the security of public-key protocols. *IEEE Transactions on Information Theory*, 29:198–208, 1983.

[9] W. Feller. *An Introduction to Probability Theory and its Applications*. John Wiley and Sons, Inc., New York, 1958.

[10] Joshua D. Guttman and F. Javier Thayer Fábrega. Protocol independence through disjoint encryption. In *Proceedings, 13th Computer Security Foundations Workshop*. IEEE Computer Society Press, July 2000.

[11] Joshua D. Guttman and F. Javier Thayer Fábrega. Authentication tests and the structure of bundles. *Theoretical Computer Science*, 2001. To appear.

[12] James Heather, Gavin Lowe, and Steve Schneider. How to prevent type flaw attacks on security protocols. In *Proceedings, 13th Computer Security Foundations Workshop*. IEEE Computer Society Press, July 2000.

[13] Gavin Lowe. Breaking and fixing the Needham-Schroeder public-key protocol using FDR. In *Proceeedings of* TACAS, volume 1055 of *Lecture Notes in Computer Science*, pages 147–166. Springer Verlag, 1996.

[14] Gavin Lowe. A hierarchy of authentication specifications. In *10th Computer Security Foundations Workshop Proceedings*, pages 31–43. IEEE Computer Society Press, 1997.

[15] Catherine Meadows. A model of computation for the NRL protocol analyzer. In *Proceedings of the Computer Security Foundations Workshop VII*, pages 84–89. IEEE, IEEE Computer Society Press, 1994.

[16] Jonathan K. Millen. The Interrogator model. In *Proceedings of the 1995 IEEE Symposium on Security and Privacy*, pages 251–60, 1995.

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

[18] Lawrence C. Paulson. Proving properties of security protocols by induction. In *10th IEEE Computer Security Foundations Workshop*, pages 70–83. IEEE Computer Society Press, 1997.

[19] Adrian Perrig and Dawn Xiaodong Song. Looking for diamonds in the desert: Extending automatic protocol generation to three-party authentication and key agreement protocols. In *Proceedings of the 13th IEEE Computer Security Foundations Workshop*. IEEE Computer Society Press, July 2000.

[20] Birgit Pfitzmann, Matthias Schunter, and Michael Waidner. Cryptographic security of reactive systems. *Electronic Notes in Theoretical Computer Science*, 32, 2000.

[21] Steve Schneider. Verifying authentication protocols with CSP. In *Proceedings of the 10th IEEE Computer Security Foundations Workshop*, pages 3–17. IEEE Computer Society Press, 1997.

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

[23] Mark N. Wegman and J. Lawrence Carter. New hash functions and their use in authentication and set equality. *Journal of Computer and System Sciences*, 22:265–79, 1981.

[24] Thomas Y. C. Woo and Simon S. Lam. Verifying authentication protocols: Methodology and example. In *Proc. Int. Conference on Network Protocols*, October 1993.

APPENDIX

A. STRANDS AND THE PENETRATOR

In this appendix, we define the basic strand space notions

used in the body of the paper. This material is derived from [22, 11].

A.1 Strand Spaces

Consider a set A, the elements of which are the possible messages that can be exchanged between principals in a protocol. We will refer to the elements of A as *terms*. We assume that a *subterm* relation is defined on A. $t_0 \sqsubset t_1$ means t_0 is a subterm of t_1. We also assume that A has a concatenation operator $\cdot$ and possibly also a cryptographic operator. We write $\{\!|t|\!\}_K$ for the result of applying the cryptographic operator to t using the secret K.

In a protocol, principals can either send or receive terms. We represent transmission of a term as the occurrence of that term with positive sign, and reception of a term as its occurrence with negative sign.

DEFINITION A.1. *A* signed term *is a pair $\langle \sigma, a \rangle$ with $a \in \mathsf{A}$ and σ one of the symbols $+, -$. We will write a signed term as $+t$ or $-t$. $(\pm\mathsf{A})^*$ is the set of finite sequences of signed terms. We will denote a typical element of $(\pm\mathsf{A})^*$ by $\langle \langle \sigma_1, a_1 \rangle, \ldots, \langle \sigma_n, a_n \rangle \rangle$.*

A strand space *over A is a set Σ together with a trace mapping $\mathrm{tr} : \Sigma \to (\pm\mathsf{A})^*$.*

By abuse of language, we will still treat signed terms as ordinary terms. For instance, we shall refer to subterms of signed terms. We will usually represent a strand space by its underlying set of strands Σ.

DEFINITION A.2. Fix a strand space Σ.

1. A *node* is a pair $\langle s, i \rangle$, with $s \in \Sigma$ and i an integer satisying $1 \leq i \leq \mathrm{length}(\mathrm{tr}(s))$. The set of nodes is denoted by $\mathcal{N}$. If $n = \langle s, i \rangle \in \mathcal{N}$ then $\mathrm{index}(n) = i$ and $\mathrm{strand}(n) = s$. Define $\mathrm{term}(n)$ to be $(\mathrm{tr}(s))_i$, i.e. the ith signed term in the trace of s.
2. There is an edge $n_1 \to n_2$ if and only if $\mathrm{term}(n_1) = +a$ and $\mathrm{term}(n_2) = -a$ for some $a \in \mathsf{A}$. Intuitively, the edge means that node n_1 sends the message a, which is received by n_2, recording a potential causal link between those strands.
3. When $n_1 = \langle s, i \rangle$ and $n_2 = \langle s, i+1 \rangle$ are members of $\mathcal{N}$, there is an edge $n_1 \Rightarrow n_2$. Intuitively, the edge expresses that n_1 is an immediate causal predecessor of n_2 on the strand s.
4. Suppose I is a set of unsigned terms. The node $n \in \mathcal{N}$ is an *entry point* for I iff $\mathrm{term}(n) = +t$ for some $t \in I$, and whenever $n' \Rightarrow^+ n$, $\mathrm{term}(n') \notin I$.
5. An unsigned term t *originates* on $n \in \mathcal{N}$ iff n is an entry point for the set $I = \{t' : t \sqsubset t'\}$.
6. An unsigned term t is *uniquely originating in* a set of nodes $S \subset \mathcal{N}$ iff there is a unique $n \in S$ such that t originates on n.
7. An unsigned term t is *non-originating in* a set of nodes $S \subset \mathcal{N}$ iff there is no $n \in S$ such that t originates on n.

A.2 Bundles and Causal Precedence

A *bundle* is a finite subgraph of the graph $\langle \mathcal{N}, (\to \cup \Rightarrow) \rangle$, for which we can regard the edges as expressing the causal dependencies of the nodes.

DEFINITION A.3. *Suppose $\to_{\mathcal{C}} \subset \to$; suppose $\Rightarrow_{\mathcal{C}} \subset \Rightarrow$; and suppose $\mathcal{C} = \langle \mathcal{N}_{\mathcal{C}}, (\to_{\mathcal{C}} \cup \Rightarrow_{\mathcal{C}}) \rangle$ is a subgraph of $\langle \mathcal{N}, (\to \cup \Rightarrow) \rangle$. $\mathcal{C}$ is a bundle if:*

1. *$\mathcal{N}_{\mathcal{C}}$ and $\to_{\mathcal{C}} \cup \Rightarrow_{\mathcal{C}}$ are finite.*
2. *If $n_2 \in \mathcal{N}_{\mathcal{C}}$ and $\mathrm{term}(n_2)$ 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.*

In conditions 2 and 3, it follows that $n_1 \in \mathcal{N}_{\mathcal{C}}$, because $\mathcal{C}$ is a graph.

DEFINITION A.4. *If $\mathcal{S}$ is a set of edges, i.e. $\mathcal{S} \subset \to \cup \Rightarrow$, then $\prec_{\mathcal{S}}$ is the transitive closure of $\mathcal{S}$, and $\preceq_{\mathcal{S}}$ is the reflexive, transitive closure of $\mathcal{S}$.*

The relations $\prec_{\mathcal{S}}$ and $\preceq_{\mathcal{S}}$ are each subsets of $\mathcal{N}_{\mathcal{S}} \times \mathcal{N}_{\mathcal{S}}$, where $\mathcal{N}_{\mathcal{S}}$ is the set of nodes incident with any edge in $\mathcal{S}$.

PROPOSITION A.5. *Suppose $\mathcal{C}$ is a bundle. Then $\preceq_{\mathcal{C}}$ is a partial order, i.e. a reflexive, antisymmetric, transitive relation. Every non-empty subset of the nodes in $\mathcal{C}$ has $\preceq_{\mathcal{C}}$-minimal members.*

A.3 Penetrator Strands

The actions available to the penetrator in the abstract Dolev-Yao model are relative to the set of keys that the penetrator knows initially. We encode this in a parameter, the set of penetrator keys $\mathsf{K}_{\mathcal{P}}$.

DEFINITION A.6. *A* penetrator trace *relative to $\mathsf{K}_{\mathcal{P}}$ is one of the following:*

$\mathbf{M}_t$ *Text message: $\langle +t \rangle$ where $t \in \mathsf{T}$*

$\mathbf{K}_K$ *Key: $\langle +K \rangle$ where $K \in \mathsf{K}_{\mathcal{P}}$*

$\mathbf{C}_{g,h}$ *Concatenation: $\langle -g, -h, +g{\cdot}h \rangle$*

$\mathbf{S}_{g,h}$ *Separation: $\langle -g{\cdot}h, +g, +h \rangle$*

$\mathbf{E}_{h,K}$ *Encryption: $\langle -K, -h, +\{\!|h|\!\}_K \rangle$*

$\mathbf{D}_{h,K}$ *Decryption: $\langle -K^{-1}, -\{\!|h|\!\}_K, +h \rangle$*

$\mathcal{P}_\Sigma$ is the set of all strands $s \in \Sigma$ such that $\mathrm{tr}(s)$ is a penetrator trace.

A strand $s \in \Sigma$ is a penetrator strand *if it belongs to $\mathcal{P}_\Sigma$, and a node is a* penetrator node *if the strand it lies on is a penetrator strand. Otherwise we will call it a* regular *strand or node.*

B. CARTER-WEGMAN HASH FUNCTIONS

We now develop Carter-Wegman universal classes [7, 23] to establish Lemma 4.2.

DEFINITION B.1. *$\phi : X \to Y$ is* uniformly distributed *iff ϕ maps the uniform distribution on X to the uniform distribution on Y. Thus,*

$$\frac{\mathsf{card}\big(\phi^{-1}(A)\big)}{\mathsf{card}(X)} = \frac{\mathsf{card}(A)}{\mathsf{card}(Y)}$$

for every $A \subseteq Y$.

Alternatively, ϕ is uniform iff the inverse image of each $y \in Y$ has cardinality $\mathsf{card}(X)/\mathsf{card}(Y)$.

For any $\phi : X \to Y$, X is the disjoint union of the sets $\phi^{-1}(y)$ for $y \in Y$. Uniform distribution means that all these sets have the same cardinality. Intuitively, uniformly distributed maps decompose X as a "product" $Y \times H$.

EXAMPLE B.2. *Let V, W be finite dimensional vector spaces over the finite field $\mathbb{F}_q$. An linear map $T : V \to W$ is uniformly distributed iff it is surjective. This will be the case iff* $\dim V - \dim \ker T = \dim W$.

PROOF. If T is surjective and $w \in W$, then $T^{-1}(w)$ is an subspace of dimension $T^{-1}(0)$.

DEFINITION B.3. *A set of functions $\mathcal{F} \subseteq Y^X$ is n-strongly universal iff $\mathsf{card}(X)$ is at least n and for any pairwise distinct $x_1, \ldots, x_n \in X$, the evaluation mapping*

$$f \mapsto \langle f(x_1), \ldots, f(x_n)\rangle$$

is uniform. Equivalently, for pairwise distinct $x_1, \ldots, x_n \in X$

$$\mathsf{P}\{f \in \mathcal{F} : \langle f(x_1), \ldots, f(x_n)\rangle = \langle y_1, \ldots, y_n\rangle\} = \frac{1}{(\mathsf{card}\, Y)^n}$$

The definition requires that the $x_1, \ldots, x_n$ be pairwise distinct. If some of the x_i's coincide, then $\langle f(x_1), \ldots, f(x_n)\rangle$ lies on a proper subspace of Y^n, in which case the evaluation mapping is non-uniform.

EXAMPLE B.4. *If $q \geq n$, the space of polynomial functions $p : \mathbb{F}_q \to \mathbb{F}_q$ with $\deg(p) \leq n-1$ is n-strongly universal. This follows from linearity of the evaluation mapping $p \mapsto (p(\theta_1), \ldots, p(\theta_n))$ and Lagrange interpolation.*

As a special case, the space of affine mappings $x \mapsto ax + b$ on finite fields is 2-strongly universal.

Note that the usual definition of n-strong universality does not require that $\mathsf{card}(X)$ be at least n. However, without this assumption, the following lemma fails.

LEMMA B.5. *If $\mathcal{F} \subseteq Y^X$ is n-strongly universal then $\mathcal{F}$ is m strongly universal for $m \leq n$.*

PROOF. If $x_1, \ldots x_m$ are pairwise distinct, extend to a pairwise distinct sequence $x_1, \ldots x_n$, which exists since $\mathsf{card}(X)$ is at least n, and use the fact the composition of uniform mappings is uniform. ■

Given $x_1, \ldots x_\ell \in X$, let us refer to a set of the form $\{i : 1 \leq i \leq \ell \wedge x_i = x\}$ as an *index class*. The set $\mathcal{C}$ of index classes clearly partition the set $\{1, \ldots, \ell\}$.

LEMMA B.6. *If $\mathcal{F} \subseteq Y^X$ is n-strongly universal, then for any $x_1, \ldots, x_n \in X$ (distinct or not) and $y_1, \ldots, y_n \in Y$,*

$$\mathsf{P}\{f \in \mathcal{F} : \langle f(x_1), \ldots, f(x_n)\rangle = \langle y_1, \ldots, y_n\rangle\} = \begin{cases} (\mathsf{card}\, Y)^{-\ell} \\ \textit{or} \quad 0 \end{cases}$$

where $\ell \leq n$ is the number of distinct $x_1, \ldots, x_n$.

PROOF. Consider the two cases: $y_i = y_j$ whenever i, j belong to the same index class and $y_i \neq y_j$ for some i, j belonging to the same index class. In the first case, we can reduce the result to the previous case by choosing an i in each index class. In the second case, there clearly can be no $f \in \mathcal{F}$ in the preimage of $\langle y_1, \ldots, y_n\rangle$. ■

We write $\langle x_1, \ldots, x_n\rangle \bowtie \langle y_1, \ldots, y_n\rangle$ to mean that corresponding elements are distinct, i.e. $x_i \neq y_i$ for all i with $1 \leq i \leq n$.

LEMMA B.7. *If $\mathcal{F} \subseteq Y^X$ is n-strongly universal, then for any $\ell \leq n$ and $x_1, \ldots, x_\ell \in X$, $y_1, \ldots, y_\ell \in Y$*

$$\mathsf{P}\{f \in \mathcal{F} : \langle f(x_1), \ldots, f(x_\ell)\rangle \bowtie \langle y_1, \ldots, y_\ell\rangle\} \geq 1 - \frac{\ell}{\mathsf{card}\, Y}.$$

Equivalently, $\mathsf{P}\{f \in \mathcal{F} : \exists i \leq \ell \,.\, f(x_i) = y_i\} \leq \ell/\mathsf{card}\, Y$.

PROOF. Assume first $x_1, \ldots, x_\ell \in X$ are distinct. By ℓ-strong universality, for each $z_1, \ldots, z_\ell \in Y$ with

$$\mathsf{P}\{f \in \mathcal{F} : \langle f(x_1), \ldots, f(x_\ell)\rangle = \langle z_1, \ldots, z_\ell\rangle\} = \left(\frac{1}{\mathsf{card}\, Y}\right)^\ell.$$

Now sum the previous inequality over $z_1, \ldots, z_\ell$ for which for all $i, 1 \leq i \leq \ell$, $z_i \neq y_i$. The cardinality of this set is $\big(\mathsf{card}\, Y - 1\big)^\ell$ so clearly in this case

$$\begin{aligned}\mathsf{P}\{f \in \mathcal{F} : \langle f(x_1), \ldots, f(x_\ell)\rangle \bowtie \langle z_1, \ldots, z_\ell\rangle\} \\ \geq & \left(1 - \frac{1}{\mathsf{card}\, Y}\right)^\ell \\ \geq & 1 - \frac{\ell}{\mathsf{card}\, Y}.\end{aligned}$$

as claimed.

Now consider the case in which there is one index class, but the y_i's are all distinct. In this case, the only way to get $\langle f(x_1), \ldots, f(x_\ell)\rangle \bowtie \langle y_1, \ldots, y_\ell\rangle$ is by choosing the common value of $f(x_i)$ distinct from all $y_1, \ldots, y_\ell$. By Assumption 3, the likelihood of this happening is $1 - \ell/\mathsf{card}(V)$.

The other cases fall somewhere in between. The case in which $y_i = y_j$ whenever i, j belong to the same index class easily reduces to the first case, by selecting an $i_C \in C$ for each index class C.

In the general case, note that the inequality worsens (that is, the left hand side decreases) as the number of y_j's increases for each index class. Thus if we assume the number of y_j's is as large as possible for each index class C, namely $\mathsf{card}(C)$ we obtain:

$$\begin{aligned}\mathsf{P}\{f \in \mathcal{F} : \langle f(x_1), \ldots, f(x_\ell)\rangle \bowtie \langle z_1, \ldots, z_\ell\rangle\} \\ \geq & \prod_{C \in \mathcal{C}} \left(1 - \frac{\mathsf{card}\, C}{\mathsf{card}\, Y}\right) \\ \geq & 1 - \sum_{C \in \mathcal{C}} \frac{\mathsf{card}\, C}{\mathsf{card}\, Y} \\ = & 1 - \frac{\ell}{\mathsf{card}\, Y}\end{aligned}$$