Shapes: Surveying Crypto Protocol Runs[1]

Joshua D. GUTTMAN
Worcester Polytechnic Institute

Abstract. Given a cryptographic protocol, and some assumptions, can we present *everything that can happen*, subject to these assumptions? The assumptions may include: (i) some behavior assumed to have occurred, (ii) some keys assumed to be uncompromised, and (iii) some values assumed to have been freshly chosen. An object representing these types of information is called a *skeleton*.

The *shapes* for a skeleton $\mathbb{A}$ are the minimal, essentially different executions that are compatible with the assumptions in $\mathbb{A}$. The set of shapes for an $\mathbb{A}$ is frequently but not always finite. Given a finite set of shapes for $\mathbb{A}$, it is evident whether a security goal such as authentication or confidentiality holds for $\mathbb{A}$.

In this paper, we describe a *search* that finds the shapes, starting from a protocol and a skeleton $\mathbb{A}$. The search is driven by the challenge-response patterns formalized in the strand space *authentication tests*.

1. Initial Examples

We develop here a search technique for finding the minimal, essentially different executions possible in a protocol, starting from some initial behavioral assumptions. This search gives counterexamples to false authentication and confidentiality assertions. Alternatively, the search proves these properties, when they hold and the search terminates, as it commonly though not universally does.

We start with intuitive analyses, using Blanchet's Simple Example Protocol [2] (see Fig. 1), and then proceed to formalize and justify them. Blanchet's protocol SEP requires an initiator A to generate a fresh symmetric key k, sign and encrypt it for a chosen responder B, and await reception of a message $\{|s|\}_k$.[2] Any responder B will await a message containing a signed and encrypted k, at which point it will select a secret s to transmit encrypted with k. A *strand* is a finite sequence of transmissions and receptions,

[1]Partly supported by the National Science Foundation, grant CNS-0952287. Preliminary versions of some of this material appeared in [7,6], written jointly with Shaddin Doghmi and Javier Thayer. That work was funded partly by MITRE-Sponsored Research, and partly by the National Security Agency. Address: guttman@{wpi.edu, mitre.org}.
This version corrects an error in Defn. 5.2.

[2]Since we frequently refer to sets, we reserve $\{vs\}$ for set formation. We write $\{|p|\}_K$ for the encryption of plaintext p using key K. If K and its inverse decryption key are equal, i.e. $K = K^{-1}$, then $\{|p|\}_K$ is a symmetric encryption, and otherwise the encryption is asymmetric.

We model digital signature by asymmetric encryption. A private encryption key K prepares the "signed" message $\{|p|\}_K$; to verify the latter, one uses the public inverse K^{-1}, recovering p. A hash is an encryption with a public key, for which no one has the matching decryption key.

Figure 1. SEP: Blanchet's Simple Example Protocol

so the actions of the initiator or responder in a single local session form a strand. Strands are written either vertically or horizontally as sequences of nodes connected by double arrows $\bullet \Rightarrow \bullet$. When n_1 follows n_0 on a strand, although possibly not immediately, then $n_0 \Rightarrow^+ n_1$.

Informally, a strand is the activity of some principal. It is the activity of a principal A if it requires using a secret known only by A, such as A's (uncompromised) signing key, or the (uncompromised) decryption key that matches A's public encryption key. However, in the formal model this principal is largely irrelevant, apart from its name, which appears as a parameter in the template defining the role. The principal parameter of the initiator role in Fig. 1 is A and the principal parameter of the responder role is B.

Our model also omits some information that is required in practice, such as the intended recipient of a message, and an indication of which key to use to decrypt it. This information is at the mercy of an adversary controlling the network. Thus, the remaining ingredients of the protocol must still be able to prevent security failures, even if these indications are misleading, so we can simplify the model by omitting them.

1.1. A's Point of View

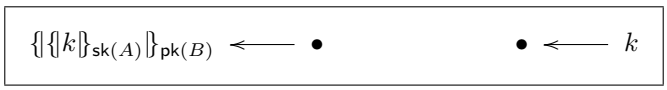
We start by exploring A's point of view. We assume that A has engaged in one or more steps of a local session of SEP, and we ask what other behavior must occur in any possible execution containing this behavior.

This is the *point-of-view principle*: *A knows* that he engaged in these steps of his local session. He would like to *infer* as much as possible about what other behavior must have occurred, or could not have occurred.

The point of view principle is central to protocol analysis, which is largely the activity of exploring what behaviors are possible, given some initially assumed behavior. This initial assumed behavior is usually a run (or part of a run) of one principal. In that case, the analysis tells us what behavior must have occurred in the distributed system, from the *point of view* of that principal.

Secrecy of the session key k. Suppose that an initiator A has executed at least the first node of a session, transmitting a session key k within a message $\{|\{|k|\}_{\mathsf{sk}(A)}|\}_{\mathsf{pk}(B)}$. Is A guaranteed that an adversary can never obtain the value k in a form protected by no encryption? The answer is *no*, in at least two cases.

1. When the key generator chosing k lacks randomness: An adversary may then generate the candidate keys and (possibly) test which was sent.
   Alternatively, the way k was chosen may ensure that it is fresh and unguessable; we use the term *uniquely originating* for such a k.



$\mathsf{non} = \{\mathsf{pk}(B)^{-1}\}$ $\quad$ $\mathsf{unique} = \{k\}$

Figure 2. Skeleton $\mathbb{A}_0$: Disclosure of k?

A originates k for this transmission, and any other place it is sent or received must then derive in an understandable way from this transmission or its later transformed forms. An adversary's generate-and-test would be a separate point of origination for the same value. Likewise, if a protocol participant were, by bad luck, to generate the same k for another run, that would be an additional point of origination for k. A reasonable cryptographic implementation of SEP should ensure that these events are of negligible likelihood.

2. When B's private decryption key $\mathsf{pk}(B)^{-1}$ is compromised: An adversary can then extract the signed unit from $\{|\{|k|\}_{\mathsf{sk}(A)}|\}_{\mathsf{pk}(B)}$, check the signature, and extract k.

   It is irrelevant whether B does this ("B is dishonest") or whether B's secret $\mathsf{pk}(B)^{-1}$ has fallen into the hands of a malicious party. In either case, B's private decryption key has been used in a way that is not stipulated in the protocol definition. Thus, we say that a key is *uncompromised* if it is used only in accordance with the protocol under analysis.

   In our formalism, a key used contrary to the stipulations of the protocol must always originate. Thus, we call an uncompromised long-term key *non-originating*.

A strand of the protocol is called a *regular* strand. Thus, all local behaviors divide into regular strands and adversary behaviors. We sometimes say that a principal A is *regular* if its private keys are used only in regular strands.

Is there any third way that an adversary could obtain the key k?

To answer this question, we carry out an experiment. We start with a diagram (Fig. 2), representing a transmission of $\{|\{|k|\}_{\mathsf{sk}(A)}|\}_{\mathsf{pk}(B)}$ and the reception of k, somehow shorn of all cryptographic protection. We call a node like the right hand node of Fig. 2 a *listener* node, since it listens, and hears the value k, thereby witnessing that k has been disclosed. The diagram also incorporates the assumption that neither case 1 nor case 2 above applies, i.e. k is uniquely originating and $\mathsf{pk}(B)^{-1}$ is non-originating, which we express as $\mathsf{unique} = \{k\}$ and $\mathsf{non} = \{\mathsf{pk}(B)^{-1}\}$. Can we embed Fig. 2 into a more informative diagram representing a possible execution?

To answer this question, we appeal to a *minimality principle*. It states that in any execution, if a set E of transmission and reception nodes is non-empty, then E has some earliest member. Moreover, if E is defined by the contents of the messages, then any earliest node in E is a transmission node. The message must have been sent before it could have been received.

Since in $\mathbb{A}_0$, there is a node in which k appears without any sort of encryption, by the minimality principle, there is a transmission node which is the earliest point at which k appears outside of the cryptographic protection of $\{|\{|k|\}_{\mathsf{sk}(A)}|\}_{\mathsf{pk}(B)}$. If the adversary could use $\mathsf{pk}(B)^{-1}$, this could occur via an adversary decryption, but the assumption

$\mathsf{pk}(B)^{-1} \in \mathsf{non}$ excludes this. If the adversary could be lucky enough to re-originate the same k, then this re-origination would be an earliest transmission unprotected by $\{|\{|k|\}_{\mathsf{sk}(A)}|\}_{\mathsf{pk}(B)}$. The assumption $\mathsf{unique} = \{k\}$ excludes this. Thus, any earliest transmission of k outside the form $\{|\{|k|\}_{\mathsf{sk}(A)}|\}_{\mathsf{pk}(B)}$ lies on a regular strand of the protocol.

However, when we examine Fig. 1, we see that a symmetric key is received by a participant only on the first node of a responder strand. This key however is not retransmitted; instead, k is used to encrypt the payload s, and the ciphertext $\{|s|\}_k$ can never lead to the disclosure of k. A principal that already knows k can use it to obtain s, but a principal that does not yet have information about k cannot obtain k from $\{|s|\}_k$. If an adversary can recognize s and has a hypothesis about the value of k, then it can use the message $\{|s|\}_k$ to *check* the hypothesis. However, we will be concerned only with full disclosure, not with a subtler notion of secrecy that resists hypothesis checking.

We have now exhausted all the possibilities. $\mathbb{A}_0$ is a dead end. No enrichment of $\mathbb{A}_0$ can be an execution that can possibly occur. We call it a *dead skeleton*.

This conclusion relies on a principle central to the search for shapes:

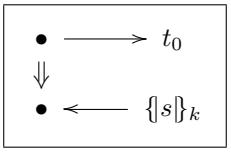
Principle 1.1 (The Nonce Test) *Suppose that $c \in \mathsf{unique}$, and c is found in some message received in a skeleton $\mathbb{A}$ at a node n_1. Moreover, suppose that, in the message of n_1, c is found outside all of a number of encrypted forms $\{|t_1|\}_{K_1}, \ldots, \{|t_j|\}_{K_j}$. Then in any enrichment $\mathbb{B}$ of $\mathbb{A}$ such that $\mathbb{B}$ is a possible execution, either:*

1. *One of the matching decryption keys ${K_i}^{-1}$ is disclosed before n_1 occurs, so that c could be extracted by the adversary; or else*
2. *Some regular strand contains a node m_1 in which c is transmitted outside $\{|t_1|\}_{K_1}, \ldots, \{|t_j|\}_{K_j}$, but in all previous nodes $m_0 \Rightarrow^+ m_1$, c was found (if at all) only within these encryptions. Moreover, m_1 occurs before n_1.*

This says that if c is extracted from the encrypted forms, then, in any possible execution, either the adversary can do so (Case 1), which we witness by adding a listener node for a decryption key ${K_i}^{-1}$; or else some regular strand has done so (Case 2). We have just applied Principle 1.1 in the case where $c = k$, $j = 1$, $K_1 = \mathsf{pk}(B)$, and $t_1 = \{|k|\}_{\mathsf{sk}(A)}$. In this application, Case 1 was excluded by the assumption $\mathsf{pk}(B)^{-1} \in \mathsf{non}$. The protocol in Fig. 1 does not furnish any instance of the behavior described in Case 2. Hence the dead end.

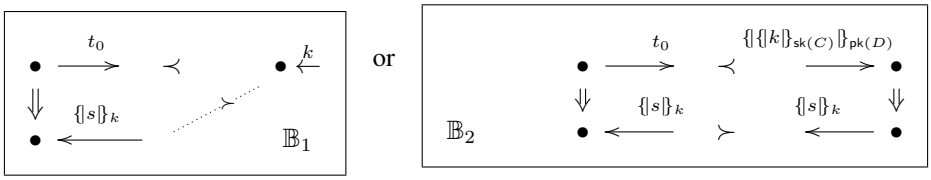
We use the list $\{|t_1|\}_{K_1}, \ldots, \{|t_j|\}_{K_j}$ because a protocol may re-use a nonce several times. After a nonce has been transmitted inside the encryption $\{|t_1|\}_{K_1}$ and received back inside the encryption $\{|t_2|\}_{K_2}$, it may be retransmitted inside the encryption $\{|t_3|\}_{K_3}$. If it is ever received back in some new form $\{|t_4|\}_{K_4}$, then that transformation needs an explanation of one of the two forms mentioned in Principle 1.1. If it is ever received back with no further encryption, then it can no longer be reused in this way.

A's Authentication Guarantee. Suppose that an initiator has executed a local session of its role in SEP. What forms are possible for the execution as a whole global behavior? In exploring this question, we will make the same assumptions about non and unique. Thus, we represent this graphically in the form shown in Fig. 3, where for brevity we write $t_0 = \{|\{|k|\}_{\mathsf{sk}(A)}|\}_{\mathsf{pk}(B)}$. We again ask what explanations could exist for the various nodes, i.e. what enrichment could elaborate $\mathbb{B}$ into a skeleton that represents a possible execution. The first node requires no explanation, since A transmits $\{|\{|k|\}_{\mathsf{sk}(A)}|\}_{\mathsf{pk}(B)}$ just as the protocol indicates.



$$\mathsf{non} = \{\mathsf{pk}(B)^{-1}\} \qquad \mathsf{unique} = \{k\}$$

Figure 3. Skeleton $\mathbb{B}$; t_0 is $\{\!|\{\!|k|\!\}_{\mathsf{sk}(A)}|\!\}_{\mathsf{pk}(B)}$



$$\mathsf{non} = \{\mathsf{pk}(B)^{-1}\} \qquad \mathsf{unique} = \{k\}$$

Figure 4. Analysis of $\mathbb{B}$, Step 1; t_0 is $\{\!|\{\!|k|\!\}_{\mathsf{sk}(A)}|\!\}_{\mathsf{pk}(B)}$

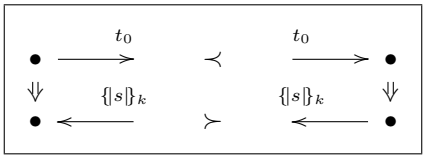
By contrast, the second node, A's reception of $\{\!|s|\!\}_k$, does require an explanation: Where did $\{\!|s|\!\}_k$ come from?

1. Possibly k is disclosed to the adversary, who then prepared the message $\{\!|s|\!\}_k$. We may test this candidate explanation by adding a listener node to witness disclosure of the encryption key k.
2. Alternatively, we may add a strand of the protocol, including a node that transmits $\{\!|s|\!\}_k$. As is evident from Fig. 1, this must be the second node of a responder strand. However, what values are possible for the other parameters of the strand, i.e. the names of the initiator and responder in this session? We will postpone the question by choosing new, unconstrained values C, D.

This leads us to the two descendants of $\mathbb{B}$, shown as $\mathbb{B}_1, \mathbb{B}_2$ in Fig. 4. We may now immediately exclude $\mathbb{B}_1$. It must be a dead end, because it is an enrichment of $\mathbb{A}_0$ in Fig. 2. If any enrichment of $\mathbb{B}_1$ were a possible execution, then it would be the enrichment of an enrichment of $\mathbb{A}_0$, and—since the composition of enrichments is an enrichment—some enrichment of $\mathbb{A}_0$ would be a possible execution.

Turning to $\mathbb{B}_2$, it has an unexplained node, the upper right node n_D receiving $\{\!|\{\!|k|\!\}_{\mathsf{sk}(C)}|\!\}_{\mathsf{pk}(D)}$. If it happens that $C = A$ and $D = B$, then nothing further need be done.

Otherwise, we may apply Principle 1.1. The value k, having been previously observed only in the form t_0, is now received on n_D in a different form, namely $\{\!|\{\!|k|\!\}_{\mathsf{sk}(C)}|\!\}_{\mathsf{pk}(D)}$. Since $\mathsf{pk}(B)^{-1} \in \mathsf{non}$, case 1 does not apply. We must thus have a regular strand that receives k only within the encrypted form t_0 and retransmits it outside of t_0. However, in analyzing $\mathbb{A}_0$, we have already seen that the protocol contains no such strand.



$\mathsf{non} = \{\mathsf{pk}(B)^{-1}\}$ $\quad$ $\mathsf{unique} = \{k\}$

Figure 5. Analysis of $\mathbb{B}$, Step 2: Its shape $\mathbb{B}_{21}$

Thus, we are left with the single case of $\mathbb{B}_2$ in which $C = A$ and $D = B$, which is the desired execution $\mathbb{B}_{21}$ shown in Fig. 5. The index 21 is meant to indicate the path along which it was encountered, as the sole child of $\mathbb{B}_2$, which is itself the rightmost child of $\mathbb{B}$. $\mathbb{B}_{21}$ is the sole *shape* for $\mathbb{B}$: Any execution compatible with $\mathbb{B}$ must contain at least the behavior shown in $\mathbb{B}_{21}$.

We have made use of two additional principles in this analysis. One asserts that death persists; the other concerns the origin of encrypted messages.

Principle 1.2 *If a skeleton $\mathbb{A}$ is dead, then so is any enrichment $\mathbb{B}$ of $\mathbb{A}$.*

We applied Principle 1.2 to discard $\mathbb{B}_1$.

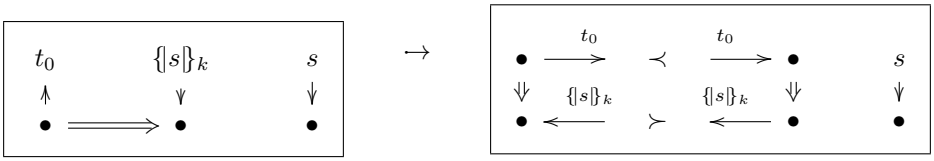
Principle 1.3 (The Encryption Test, 1) *Suppose that $\{|t|\}_K$ is found in some message received in a skeleton $\mathbb{A}$ at a node n_1. Then in any enrichment $\mathbb{B}$ of $\mathbb{A}$ such that $\mathbb{B}$ is a possible execution, either:*

1. *The encryption key K is disclosed before n_1 occurs, so that the adversary could construct $\{|t|\}_K$ from t; or else*
2. *A regular strand contains a node m_1 in which $\{|t|\}_K$ is transmitted, but no earlier node $m_0 \Rightarrow^+ m_1$ contains $\{|t|\}_K$. Moreover, m_1 occurs before n_1.*

We applied Principle 1.3 to construct skeletons $\mathbb{B}_1, \mathbb{B}_2$, using the instance $t = s$ and $K = k$. Case 1 furnished $\mathbb{B}_1$ and Case 2 yielded $\mathbb{B}_2$. The node n_1 is the later (reception) node of $\mathbb{B}$, and m_1 is the lower right transmission node in $\mathbb{B}_2$.

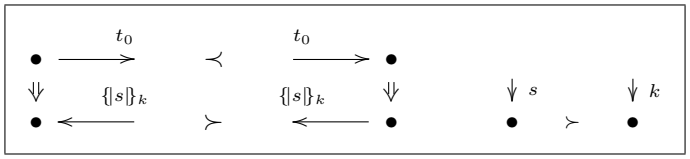
We will strengthen Principle 1.3 and combine it in a single form with Principle 1.1, resulting in the Authentication Test Principle, Theorem 5.5 of Section 5.

Secrecy of s. Can A be sure that the value s remains a secret between A and B? To test this, we start with an expansion of skeleton $\mathbb{B}$ in which there is also a listener node that observes s shorn of all cryptographic protection, as shown in the left portion of Fig. 6. The question is only relevant if s is assumed fresh and unguessable. $\mathbb{C}$ is an enrichment of $\mathbb{B}$. Every execution enriching $\mathbb{B}$ must contain at least the structure we found in $\mathbb{B}_{21}$, and it must also contain the listener node for s. Thus, it must be an enrichment of $\mathbb{C}_{21}$. Now, at this point we can apply Principle 1.1, instantiating $c = s$ and node n_1 being the listener at the lower right. The index $j = 1$, and the encrypted form containing s is $\{|s|\}_k$. Since k is a symmetric session key, $k^{-1} = k$. Since no regular strand of SEP receives a value encrypted by a symmetric key and retransmits that value in any other form, Case 2 of the principle is vacuous. Thus, we add a listener node for k, witnessing for its disclosure,



$$\mathsf{non} = \{\mathsf{pk}(B)^{-1}\} \qquad \mathsf{unique} = \{k, s\}$$

Figure 6. Skeletons $\mathbb{C}$ and $\mathbb{C}_{21}$



$$\mathsf{non} = \{\mathsf{pk}(B)^{-1}\} \qquad \mathsf{unique} = \{k, s\}$$

Figure 7. Dead skeleton $\mathbb{C}_{211}$

obtaining $\mathbb{C}_{211}$ in Fig. 7. $\mathbb{C}_{211}$ is dead as a consequence of Principle 1.2, since $\mathbb{C}_{211}$ certainly enriches the dead skeleton $\mathbb{A}_0$ in Fig. 2.

Thus, SEP fulfills its goals, from the point of view of an initiator A.

In the step from $\mathbb{C}$ to $\mathbb{C}_{21}$, we used an additional principle:

Principle 1.4 *Suppose that $\mathbb{B}$ has the shapes $\mathbb{S}_1, \ldots, \mathbb{S}_i$. If $\mathbb{C}$ enriches $\mathbb{B}$, then every execution enriching $\mathbb{C}$ is an enrichment of some $\mathbb{S}_j$, where $1 \leq j \leq i$.*

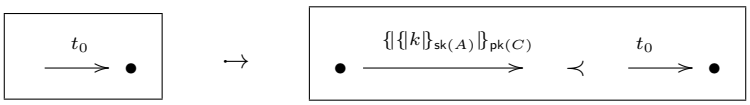
Since $\mathbb{B}$ had the single shape $\mathbb{C}_{21}$, we applied Principle 1.4 with $i = 1$ and $\mathbb{S}_1 = \mathbb{B}_{21}$, allowing us to jump right from $\mathbb{C}$ to $\mathbb{C}_{21}$. We could also have reconstructed its contents using several applications of the other principles.

1.2. B's Point of View

The story is quite different when we turn to the point of view of a responder B.

B's Authentication Guarantee. Suppose that a responder B has received a message of the form $t_0 = \{|\{|k|\}_{\mathsf{sk}(A)}|\}_{\mathsf{pk}(B)}$. Assuming now, in skeleton $\mathbb{D}$ of Fig. 8, that both A's private signature key $\mathsf{sk}(A)$ and B's private decryption key $\mathsf{pk}(B)^{-1}$ are non-originating, what else must have happened in any enrichment of $\mathbb{D}$ that is a possible execution? We may try to apply Principle 1.3 again, where the encrypted unit $\{|t|\}_K$ is $t_0 = \{|\{|k|\}_{\mathsf{sk}(A)}|\}_{\mathsf{pk}(B)}$. However, Case 1 then requires only that the public encryption key $\mathsf{pk}(B)$ of B is available to the adversary, from which we learn nothing.

We may more profitably apply Principle 1.3 by taking the encrypted unit $\{|t|\}_K$ to be $\{|k|\}_{\mathsf{sk}(A)}$. Since the key $\mathsf{sk}(A)$ is non-originating, Case 1 is vacuous. Thus, every possible execution must include an enrichment with a regular node producing $\{|k|\}_{\mathsf{sk}(A)}$. By Fig. 1, this must be the first node of an initiator strand. We know that the parameter



$$\mathsf{sk}(A), \mathsf{pk}(B)^{-1} \in \mathsf{non}$$

Figure 8. Skeleton $\mathbb{D}$: B's Point of View, and its shape $\mathbb{D}_1$

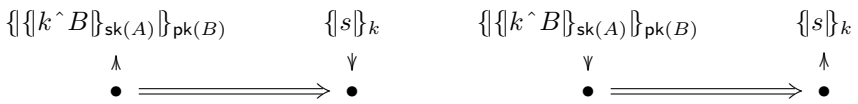


Figure 9. SEPC: the Simple Example Protocol Corrected

representing the initiator's name is A, and the parameter representing the session key has the value k. However, we know nothing about the remaining parameter appearing in an initiator's first node, i.e. the name of the intended responder. Since this value is unconstrained, we fill it in with some new C, thus obtaining the skeleton $\mathbb{D}_1$.

Unfortunately, we cannot collect any more information about the parameter C, unlike our situation in skeleton $\mathbb{B}_2$ (Fig. 4). $\mathbb{D}_1$ contains all of the regular behavior needed for an execution. It is the sole shape for $\mathbb{D}$.

Nothing says that C's decryption key is uncompromised, so the adversary can decrypt the outer layer, using the public key $\mathsf{pk}(B)$ to re-encrypt $\{\!|k|\!\}_{\mathsf{sk}(A)}$ in the desired form. Evidently, the session key k may also be disclosed in this process. Thus, in SEP, a responder B does get a guarantee that A initiated a session with key k. However, since A may have chosen a compromised party C as partner for that conversation, B cannot count on much, certainly not that k, or any s encrypted with k, will remain confidential.

1.3. Correcting SEP

Principles 1.1 and 1.3 are central to protocol design [9] as well as to protocol analysis [7]. In SEP, our analysis of B's guarantee applied Principle 1.3 to the critical term $\{\!|k|\!\}_{\mathsf{sk}(A)}$. Since this term involves only the two parameters k, A, evidently this cannot force agreement on a particular responder B.

To force A to agree with B on the responder, it suffices to add B's name to this critical term. The resulting protocol SEPC takes the form given in Fig. 9. B's authentication result is shown in Fig. 10.

If we add to $\mathbb{E}_1$ a listener node for k, and assume k uniquely originating, the resulting skeleton is an enrichment of the SEPC analogue to $\mathbb{A}_0$. It again follows that k cannot be disclosed. When we extend B's strand to include its second node, transmitting $\{\!|s|\!\}_k$, it will also lead to the conclusion that s is undisclosed.

Our correction of SEP is tighter or more minimal than Blanchet's [2], where the signed unit $\{\!|k \,\hat{}\, A \,\hat{}\, B|\!\}_{\mathsf{sk}(A)}$ is used. The occurrence of A is unnecessary here. Principle 1.3 helped identify exactly the parameters need to appear in the critical term to achieve the protocol refinement.

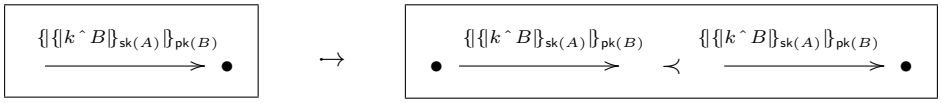


Figure 10. Skeleton $\mathbb{E}$: B's Point of View, and its shape $\mathbb{E}_1$

1.4. Goals of this Chapter

Blanchet's Simple Example Protocol has illustrated the idea that, from a particular starting point, one can find all of the minimal, essentially different things that can happen, compatible with that starting point. We call the minimal, essentially different executions compatible with a starting point $\mathbb{A}$ its *shapes*.

This chapter describes a search procedure that finds shapes systematically. Each search step from the starting point $\mathbb{A}$ adds information to $\mathbb{A}$, until the search has found behavior of the regular (uncompromised) participants, which—when combined with some activity of the adversary—yields a full execution. Each search step increases information; i.e. it is an enrichment in the sense we will formalize in the notion of *homomorphism* (Def. 3.6).

There are three important ways to add information. We can add information by adding *listener nodes* to witness for the assumption that a value is disclosed (Case 1 of Principles 1.1 and 1.3). We can add information by adding new protocol *message transmissions* that help to explain the messages that have been received (Case 2 of Principles 1.1 and 1.3). And we can add information by identifying different parameters, as we identified C and D with A and B respectively, to produce skeleton $\mathbb{B}_{21}$. When there are different possible ways to explain some one aspect of existing behavior, the search branches. We implemented this search in a tool called CPSA, a Cryptographic Protocol Shape Analyzer [17].

The purpose of this chapter is to present a theory underlying CPSA. The now very informative software that implement this theory—specified, from a different point of view, in [18]—will be the subject on another occasion. We prove three main results about the search steps as a process of refinement leading to shapes:

1. The search steps are *sound* (Thm. 6.7), so that—when we take a step—every possible execution compatible with the assumptions before the step is still compatible on at least one branch after the step.
2. The search steps are *finitely branching* (Thm. 6.8), so that each step produces only finitely many new skeletons to explore.
3. The search steps are *complete* (Thm. 6.5), so that every shape is reached by some finite sequence of search steps.

These results do not imply decidability for security goals, since the search may enumerate an infinite set of shapes. If one of these is a counterexample to a goal, then we certainly learn the goal is false. However, if a goal is in fact true, but $\mathbb{A}$ has infinitely many shapes, then we cannot be sure it is true at any finite stage.

The shape search is related to Athena [19], which also searched for executions that extend a given partial description. Athena's representation included adversary behav-

ior as well as regular strands. Athena used a more straightforward backward search, in which the tool seeks all possible transmission points—whether adversary actions or regular strands—for each component of a reception node that cannot yet be explained. Athena later incorporated an early version of the authentication test method (Principles 1.1 and 1.3) to prune its search [15]. Cremers's Scyther [5,4] refined Athena's ideas, combining them with the notion of *characterization* [8], which we describe below in Section 3.3. Unlike our search, Scyther can provide a bounded-session analysis when the unbounded-session analysis proves too costly, and this is an advantage. Our work differs in its emphasis on the authentication test method, and in its systematic treatment of enrichment via the skeletons-and-homomorphisms theory of Section 3.

1.5. Structure of this Chapter

Principles 1.1 and 1.3 have a different character from Principles 1.2 and 1.4. The former determine the potential explanations for unexplained behavior, and they drive the form of the search steps. By contrast, the latter are general observations about skeletons, about enrichments or *homomorphisms*—as we will call them—between them, and about executions. We will examine the latter in Section 3, after introducing the basic strand space terminology in Section 2.

Section 4 introduces a second example, a modified form of the Trusted Computing Group's protocol for constructing certificates for Attestation Identity Keys [1]. This suggests a strengthened version of Principle 1.3, which is parallel with Principle 1.1 in form. Section 5 states a combined version of Principles 1.1 and 1.3, showing that they characterize when a skeleton is an execution (Thm. 5.5). Section 6 defines the search algorithm, and concludes with the three main results.

2. Messages, Strands, Protocols

In our examples in Section 1, the strands send and receive messages that are built up using tupling, written $t_0 \hat{} t_1$, and encryption, written $\{|t|\}_K$. The basic values that contribute to messages are keys such as k or $\mathsf{sk}(A)$, names A, B, etc., and payloads such as s. We provide more information about the messages in Sections 2.1–2.3. Sections 2.4 and 2.5 define *strands* and *protocols* respectively.

2.1. Algebras of Messages

We regard the messages as forming an algebra M. M is closed under the two operations of *tupling* and *encryption*. We allow tupling to take a *tag*, selected from some set TAG, and any number $k \geq 0$ messages, to produce a message. The tags serve as a kind of constant to conveniently distinguish among different forms of message, and they may be implemented using distinctive bit-patterns inside messages. Encryption applies to two messages, one representing the plaintext and the other representing the key.

Definition 2.1 *Let* X *be an infinite set of objects called* indeterminates*, and let* B *be an algebra of objects called* basic values*, disjoint from the indeterminates. Let* TAG *be a disjoint set of objects called tags.*

An algebra $\mathsf{M}_{\mathsf{X}}[\mathsf{B}]$ of messages *is freely generated from* X, B *by two operations:*

Tagged Tupling *If* $tag \in \mathsf{TAG}$ *and* $t_1, \ldots, t_k \in \mathsf{M_X[B]}$*, for* $k \geq 0$*, then the* tagged k-tuple $tag\ t_1 \hat{} \ldots \hat{} t_k$ *is in* $\mathsf{M_X[B]}$.

Encryption *If* $t_1, t_2 \in \mathsf{M_X[B]}$*, then their* encryption $\{|t_1|\}_{t_2} \in \mathsf{M_X[B]}$.

We stipulate that there is a distinguished tag $null \in \mathsf{TAG}$, and we write $null\ t_1 \hat{} \ldots \hat{} t_k$ in the form $t_1 \hat{} \ldots \hat{} t_k$. That is, the tag $null$ is invisible; the examples in Section 1 and most of those later in this chapter use $null$.

We refer to $\mathsf{M_X[B]}$ as an algebra, because we are interested in its homomorphisms. We focus on its homomorphisms $\mathsf{M_X[B]} \to \mathsf{M_X[B]}$ to itself ("endomorphisms"). We will not fully specify B, but will give a sample algebra in Section 2.2, and in Section 2.3 we will identify the crucial properties we will depend on for the later development. Each B will be equipped with its own homomorphisms $\mathsf{B} \to \mathsf{B}$.

We often omit X or both X and B, and write M or $\mathsf{M[B]}$.

Definition 2.2 *A homomorphism* $\mathsf{M} \to \mathsf{M}$ *is a pair* $\alpha = (\eta, \chi)$*, where* $\eta\colon \mathsf{B} \to \mathsf{B}$ *is a homomorphism on* B *and* χ *is a function* $\chi\colon \mathsf{X} \to \mathsf{M}$*. It is defined for all* $t \in \mathsf{M}$ *by the conditions:*

$$\begin{array}{ll} \alpha(a) = \eta(a), & \text{if } a \in \mathsf{B} \\ \alpha(x) = \chi(x), & \text{if } x \in \mathsf{X} \end{array} \qquad \begin{array}{r} \alpha(\{|t_0|\}_{t_1}) = \{|\alpha(t_0)|\}_{\alpha(t_1)} \\ \alpha(tag\ t_1 \hat{} \ldots \hat{} t_n) = tag\ \alpha(t_1) \hat{} \ldots \hat{} \alpha(t_n) \end{array}$$

Indeterminates x are untyped "blank slots," replaceable by any message. Tags remain constant under homomorphisms. Messages are abstract syntax trees:

Definition 2.3 1. *Let* ℓ *and* r *be the partial functions where:*

$$\begin{array}{rll} t = \{|t_1|\}_{t_2} & \text{implies} & \ell(t) = t_1 \text{ and } r(t) = t_2; \\ t = tag\ t_1 \hat{} t_2 \hat{} \ldots \hat{} t_j & \text{implies} & \ell(t) = t_1 \text{ and } r(t) = t_2 \hat{} \ldots \hat{} t_j; \\ t \in \mathsf{B} & \text{implies} & \ell(t) \text{ and } r(t) \text{ are undefined.} \end{array}$$

2. *A* path p *is a sequence in* $\{\ell, r\}^*$*. We write* $\mathsf{cons}(f, p)$ *for the sequence whose first member is* f *and whose successive elements are those of* p*. We write* $p_1 \frown p_2$ *for the result of appending* p_2 *to the end of* p_1.
   We regard p *as a partial function, where* $\langle\rangle = \mathsf{Id}$ *and* $\mathsf{cons}(f, p) = p \circ f$*. When the rhs is defined, we have:*
   (a) $\langle\rangle(t) = t$*;*
   (b) $\mathsf{cons}(\ell, p)(t) = p(\ell(t))$*; and*
   (c) $\mathsf{cons}(r, p)(t) = p(r(t))$.
3. p traverses a key edge *in* t *if* $p = p_1 \frown \langle r \rangle \frown p_2$ *and* $p_1(t)$ *is an encryption.*
4. t_0 is an ingredient of t*, written* $t_0 \sqsubseteq t$*, if* $t_0 = p(t)$ *for some* p *that does not traverse a key edge in* t.
5. t_0 appears in t*, written* $t_0 \ll t$*, if* $t_0 = p(t)$ *for some* p.
6. p traverses a member of S *in* t *if* $p = p_1 \frown p_2$*, where* $p_1(t) \in S$ *and* $p_2 \neq \langle\rangle$.

As an example, consider the message $t = \{|\{|k|\}_{\mathsf{sk}(A)}|\}_{\mathsf{pk}(B)}$; for

$p_0 = \langle \ell, \ell \rangle$**,** we have $k = p_0(t)$. Since p_0 does not traverse a key edge, $k \sqsubseteq \{|\{|k|\}_{\mathsf{sk}(A)}|\}_{\mathsf{pk}(B)}$.

$p_1 = \langle r \rangle$**,** we have $\mathsf{pk}(B) = p_1(t)$. However, since p_1 traverses a key edge, we have established only the weaker $\mathsf{pk}(B) \ll \{|\{|k|\}_{\mathsf{sk}(A)}|\}_{\mathsf{pk}(B)}$.

$p_2 = \langle \ell, r \rangle$, we have $\mathsf{sk}(A) = p_2(t)$. Since p_1 again traverses a key edge, we have only $\mathsf{sk}(A) \ll \{\!|\{\!|k|\!\}_{\mathsf{sk}(A)}|\!\}_{\mathsf{pk}(B)}$.

In $\{\!|s|\!\}_k$, only the path $\langle r \rangle$ leads to k. Hence, $k \ll \{\!|s|\!\}_k$ but $k \not\sqsubseteq \{\!|s|\!\}_k$. Since $\langle \ell \rangle$ leads to s, $s \sqsubseteq \{\!|s|\!\}_k$.

2.2. A Sample Algebra of Basic Values

The algebra of basic values may take many forms, and its details affect the remainder of our development only in a limited way. In this section, we give a sample B, for the sake of concreteness, which suffices for the example of Section 1. The example of Section 4 requires some additional machinery in B, but no new ideas. Lemma 2.7 characterizes the essential requirements: if any B yields an $\mathsf{M}[\mathsf{B}]$ satisfying Lemma 2.7, then the remainder of the chapter will hold for that $\mathsf{M}[\mathsf{B}]$.

Let $\mathbb{N}$ be the natural numbers. We use the following constructors, i.e. injective functions with disjoint range:

$\mathsf{nm}, \mathsf{txt}, \mathsf{skey}$ with domain $\mathbb{N}$. Their ranges are called *names*, *texts*, and *symmetric keys*, respectively.

sk, pk with domain *names*, yielding values called *signature keys* and *public keys* respectively.

invk with domain *signature keys* $\cup$ *public keys*, yielding values called *verification keys* and *private decryption keys* resp.

We define the inverse function a^{-1} so that

$$\begin{array}{ll} \mathsf{sk}(\mathsf{nm}(j))^{-1} = \mathsf{invk}(\mathsf{sk}(\mathsf{nm}(j))) & \mathsf{invk}(\mathsf{sk}(\mathsf{nm}(j)))^{-1} = \mathsf{sk}(\mathsf{nm}(j)) \\ \mathsf{pk}(\mathsf{nm}(j))^{-1} = \mathsf{invk}(\mathsf{pk}(\mathsf{nm}(j))) & \mathsf{invk}(\mathsf{pk}(\mathsf{nm}(j)))^{-1} = \mathsf{pk}(\mathsf{nm}(j)), \end{array}$$

while for all other values $a^{-1} = a$.

Each homomorphism $\eta \colon \mathsf{B} \to \mathsf{B}$ is determined by a function f mapping names to names, texts to texts, and symmetric keys to symmetric keys. Moreover, each such f extends to a unique homomorphism.

Indeterminates, as we mentioned, are akin to untyped variables, since homomorphisms can send them to any value in $\mathsf{M}_\mathsf{X}[\mathsf{B}]$. By contrast, basic values are akin to sorted variables. Homomorphisms map them to values of the same sort.

PARAMS is the union of the indeterminates X and the set of all basic values in B of the forms $\mathsf{nm}(i), \mathsf{txt}(i)$, and $\mathsf{skey}(i)$. The *parameters* of a message $t \in \mathsf{M}$—written $\mathsf{params}(t)$—are all of its subexpressions $s \in \mathsf{PARAMS}$.

Lemma 2.4 1. *Each message t has finitely many parameters* $\mathsf{params}(t)$*; and*
2. *Letting $\alpha, \beta \colon \mathsf{M}_\mathsf{X}[\mathsf{B}] \to \mathsf{M}_\mathsf{X}[\mathsf{B}]$, if (i) $\mathsf{params}(t) \subseteq \mathsf{X} \cup \mathsf{B}$, and (ii) for all $v \in \mathsf{params}(t)$, $\alpha(v) = \beta(v)$, then $\alpha(t) = \beta(t)$.*

2.3. Properties of Homomorphisms

Definition 2.5 *Consider homomorphisms α, β, ι between subalgebras of $\mathsf{M}_\mathsf{X}[\mathsf{B}]$.*

1. *A parameter $v \in \mathsf{PARAMS}$ is a* source parameter *of α iff $\alpha(v) \neq v$.*

2. α *is* finitely generated *iff it has finitely many source parameters.*
3. *Messages* $v, w \in \mathsf{M}$ have a common instance *iff* $\alpha(v) = \alpha(w)$ *for some* α*, in which case* α unifies v, w.
4. *A homomorphism* ι *is an* isomorphism*, or a* renaming*, iff there exists a* κ *such that* $\kappa \circ \iota$ *and* $\iota \circ \kappa$ *are the identity* Id.
   By the symmetry of the definition, in this case κ *is a renaming also.*
5. α, β differ by a renaming *iff for some renaming* ι, $\iota \circ \alpha = \beta$.
6. *If there exists a* β *such that* $\gamma = \beta \circ \alpha$*, then we say that* γ *is* at least as specific as α*, and write* $\alpha \leq_s \gamma$.

In clause 4, κ may be defined only on a subalgebra such as the range of ι, and likewise ι in clause 5 may be defined only on the range of α. A *preorder* means a reflexive, transitive relation.

Lemma 2.6 1. *If* $\beta \circ \alpha = \mathsf{Id}$*, then* α *is a renaming.*
2. Differing by a renaming *is an equivalence relation.*

Proof: 1. Suppose α is defined on $\mathsf{M}_{\mathsf{X}_i}[\mathsf{B}_i]$. First, α is injective: if $\alpha(t) = \alpha(t')$, then since $t = \beta(\alpha(t)) = \beta(\alpha(t')) = t'$, $t = t'$. Moreover, letting $\mathsf{X}_j = \alpha(\mathsf{X}_i)$ and B_j be generated from $\alpha(\mathsf{PARAMS}(\mathsf{B}_i))$, we have α surjective onto $\mathsf{M}_{\mathsf{X}_j}[\mathsf{B}_j]$. Thus, α is an isomorphism $\alpha\colon \mathsf{M}_{\mathsf{X}_i}[\mathsf{B}_i] \to \mathsf{M}_{\mathsf{X}_j}[\mathsf{B}_j]$, and has an inverse $\kappa\colon \mathsf{M}_{\mathsf{X}_j}[\mathsf{B}_j] \to \mathsf{M}_{\mathsf{X}_i}[\mathsf{B}_i]$.

2. *Differing by a renaming* is reflexive because Id is a renaming, and it is transitive because the composition of two renamings is a renaming.

To see that it is symmetric, suppose that $\iota \circ \alpha = \beta$. Since ι is a renaming, there is a κ such that $\kappa \circ \iota = \iota \circ \kappa = \mathsf{Id}$. Applying κ to each side of the assumed equation, $\kappa \circ (\iota \circ \alpha) = \kappa \circ \beta$. However, $\kappa \circ (\iota \circ \alpha) = (\kappa \circ \iota) \circ \alpha = \mathsf{Id} \circ \alpha = \alpha$. So κ is a renaming such that $\alpha = \kappa \circ \beta$. □

We sometimes call α, β *isomorphic* when they differ by a renaming.

Lemma 2.7 1. $\leq_s$ *is a preorder on homomorphisms.*
2. $\alpha \leq_s \gamma \leq_s \alpha$ *implies that* α *and* γ *differ by a renaming. Hence,* $\leq_s$ *is a partial order on the equivalence classes.*
3. *When* $\gamma = \beta \circ \alpha = \beta' \circ \alpha$*, then* $\beta(a) = \beta'(a)$ *for all* $a \in \mathrm{rng}(\alpha)$*. Thus, the choice of* β *in Def. 2.5, Clause 6 is unique, on* $\mathrm{rng}(\alpha)$.
4. *For any finitely generated* γ*, the set* $\{\alpha\colon \alpha \leq_s \gamma\}$ *contains only finitely many non-isomorphic members.*

Proof: 1. Using Id and composition.

2. If $\alpha \leq_s \gamma \leq_s \alpha$, then there exist β, δ such that $\beta \circ \alpha = \gamma$ and $\delta \circ \gamma = \alpha$. Hence, letting β' be the restriction of β to $\mathsf{ran}(\alpha)$, we may apply Lemma 2.6, clause 1.

3. If $a = \alpha(a_0)$, then $\beta(a) = \beta(\alpha(a_0)) = \gamma(a_0) = \beta'(\alpha(a_0)) = \beta'(a)$.

4. Define $\mathsf{ascore}(\alpha, a)$, for $a \in \mathsf{PARAMS}$, to be $|\{b \in \mathsf{PARAMS}\colon \alpha(b) = a\}| - 1$. Define $\mathsf{xscore}(\alpha, x)$, for $x \in \mathsf{X}$, to be number of encryptions and concatenations in $\alpha(x)$ plus the number of multiple occurrences of basic values and indeterminates in $\alpha(x)$. Define $\mathsf{xyscore}(\alpha, x, y)$ to be 0 if $x = y$, and otherwise the number of occurrences of basic values and indeterminates shared between $\alpha(x)$ and $\alpha(y)$. The *score* of α is the sum:

$$\sum_{a \in \mathsf{PARAMS}} \mathsf{ascore}(\alpha, a) + \sum_{x \in \mathsf{X}} \mathsf{xscore}(\alpha, x) + \sum_{x,y \in \mathsf{X}} \mathsf{xyscore}(\alpha, x, y).$$

Then score is non-negative, and well defined when α is finitely generated. Moreover, when $\alpha \leq_s \gamma$, and γ is finitely generated, $\mathsf{score}(\alpha) \leq \mathsf{score}(\gamma)$. If $\mathsf{score}(\alpha) = \mathsf{score}(\gamma)$, then α, γ differ by a renaming. Hence, $\ldots \leq_s \alpha_i \leq_s \ldots \leq_s \alpha_1 \leq_s \alpha_0$ can have only finitely many non-isomorphic members. Indeed, when $\mathsf{score}(\gamma) = k + 1$, there are only finitely many non-isomorphic $\alpha \leq_s \gamma$ such that $\mathsf{score}(\alpha) = k$. □

A homomorphism γ *unifies* messages v, w if $\gamma(v) = \gamma(w)$, and γ is a *most general unifier* for v, w if in addition, whenever γ' unifies v, w, then $\gamma \leq_s \gamma'$.

Lemma 2.8 1. *Assume $v, w \in \mathsf{M}$ have a common instance. There exists a finitely generated γ which is a most general unifier of v and w.*

2. *Let $v = \alpha(u)$ and $w = \beta(u)$. Then α, β have a most specific common generalization γ for u. That is, there is a finitely generated γ such that:*
   (a) *For some α_1, $v = (\alpha_1 \circ \gamma)(u)$,*
   (b) *For some β_1, $w = (\beta_1 \circ \gamma)(u)$, and*
   (c) *if γ' satisfies Clauses 2a–2b, then $\gamma' \leq_s \gamma$.*

Lemmas 2.4, 2.6–2.8 summarize our central assumptions on the algebra. We believe that for any algebra of basic values B, where every basic value can be assigned a finite set of parameters, such that the resulting $\mathsf{M}_\mathsf{X}[\mathsf{B}]$ satisfies these lemmas, will satisfy the remaining results in this chapter.

Adjusted versions of our method appear still usable when the m.g.u. property fails, but instead a finite set of unifiers cover the common instances of any two messages, and where a finite set of homomorphisms replace the most specific common generalization of Clause 2. Sometimes the adversary has useful additional operations on basic values, and some additional authentication test patterns must be defined on the basic values. By contrast, in algebras with exclusive-or (e.g.), which may be applied to tuples or encryptions rather than just basic values, and is subject to equational laws, other entirely new ideas are required.

2.4. Strands and Origination

A single local session of a protocol at a single principal is a *strand*, containing a linearly ordered sequence of transmissions and receptions that we call *nodes*. A transmission of message t is a directed term $+t$, and a reception of message t is a directed term $-t$.

The i^{th} node on strand s is $s \downarrow i$, using 1-based indexing. If n, m are successive nodes on the same strand, $n \Rightarrow m$ holds, meaning that $n = s \downarrow i$ and $m = s \downarrow i + 1$. The transitive closure of $\Rightarrow$ is $\Rightarrow^+$, and its reflexive transitive closure is $\Rightarrow^*$. The message sent or received on the node n is $\mathsf{msg}(n)$.

Origination. A message t_0 *originates* at a node n_1 if (1) n_1 is a transmission node; (2) $t_0 \sqsubseteq \mathsf{msg}(n_1)$; and (3) whenever $n_0 \Rightarrow^+ n_1$, $t_0 \not\sqsubseteq \mathsf{msg}(n_0)$. Thus, t_0 originates when it was transmitted without having been either received or transmitted (as an ingredient) previously on the same strand. Cf. Def. 2.3 for $\sqsubseteq$.

Values assumed to originate only on one node in an execution—*uniquely originating* values—formalize the idea of freshly chosen, unguessable values. Values assumed to originate nowhere may be used to encrypt or decrypt, but are never sent as message ingredients. They are called *non-originating* values. For a non-originating value K, $K \not\sqsubseteq t$ for any transmitted message t. However, $K \ll \{|t_0|\}_K \sqsubseteq t$ possibly, which is why we distinguish $\sqsubseteq$ from $\ll$ in Def. 2.3.

As an example, when applying Principle 1.3 to Fig. 2, we stated that k is not retransmitted by any strand that has received it. This was meant in the sense of $\sqsubseteq$. We never have $k \sqsubseteq \mathsf{msg}(n)$ if for some reception node m, $m \Rightarrow^+ n$ and $k \sqsubseteq \mathsf{msg}(m)$, and n is a transmission node. However, we may have $k \ll \mathsf{msg}(n)$, which is harmless because it does not contribute to disclosure of k.

We say that n is *the origin of* t_0 in a set of nodes S if (1) $n \in S$, (2) t_0 originates at n, and (3) t_0 originates at no other node in S. It is *uniquely originating in* S if for some n, n is the origin of t_0 in S.

We say that t_0 is *non-originating in* S if there is no $n \in S$ such that $t_0 \sqsubseteq \mathsf{msg}(n)$. Evidently, if there is any $n \in S$ such that $t_0 \sqsubseteq \mathsf{msg}(n)$, then any full execution extending S will have to provide an origin for t_0.

Principals and Other Parameters. We extend the notion of *parameters* from messages to nodes and strands cumulatively. Suppose that s is a strand of length ℓ, and $i \leq \ell$; then $\mathsf{params}(s) = \mathsf{params}(s \downarrow \ell)$ where

$$\mathsf{params}(s \downarrow i) = \bigcup_{0<j\leq i} \mathsf{params}(\mathsf{msg}(s \downarrow j)).$$

The parameters to a node are the potentially varying arguments which can affect the messages sent or received up to and including that node. By Lemma 2.4, a strand s has only finitely many parameters, and any two homomorphisms agreeing on $\mathsf{params}(s)$ have the same action on messages sent and received along s.

In the model we use in this paper, there is no relevant entity acting as a principal. There are only names, and these names serve as parameters to some strands. Names are also associated with keys via the functions $\mathsf{pk}(\cdot)$, $\mathsf{sk}(\cdot)$. Thus, although informally we view several strands as being ongoing activities of a single principal at a particular time, in the model, there are only strands that share a particular name parameter and use the keys associated with that parameter. When a protocol may manipulate some long-term state belonging to the principals executing it, then we work in a richer model [11].

2.5. Protocols

A *protocol* Π is a finite set of strands, called the *roles* of Π, together with some constraints on unique and non-origination. We assume that every protocol Π contains the listener role $\mathsf{Lsn}[x]$, which consists of a single reception node $\stackrel{x}{\rightarrow} \bullet$. Instances of this listener role have already appeared in Figs. 2, 4, 6–7.

The constraints on origination for roles, which we will illustrate in an example in Section 4, may be used to ensure that a particular role always contributes a uniquely originating value to an execution, as for instance a session key server may be trusted always to choose a fresh and unguessable session key. They may also be used to ensure

that a particular role always involves a non-originating key. For instance, a protocol may ensure that the certificate authority role always uses a non-originating signing key. In modeling SEP, role origination constraints were not needed; we assumed only that particular values in a skeleton were non-originating or uniquely originating. There was no need to assume that every time we added a strand, some parameters to it would satisfy origination constraints.

The Indeterminate Acquisition Principle. Since indeterminates represent syntactically unconstrained messages received from protocol peers, or passed down as parameters a higher-level protocol, we require an indeterminate to be received as an ingredient before appearing in any other way:

If n_1 is a node on $\rho \in \Pi$, with an indeterminate $x \ll \mathsf{msg}(n_1)$,
then $\exists n_0$, $n_0 \Rightarrow^* n_1$, where n_0 is a reception node and $x \sqsubseteq \mathsf{msg}(n_0)$.

This Indeterminate Acquisition Principle is related to the requirement in constraint-solving approaches—originating with [13]—that variables in constraint sequences always appear first on the right-hand side of a constraint. The right hand sides represent message receptions, where the adversary must generate some instance of the constraint, and may select a value for the variable that makes this possible.

As an example protocol, the two strands shown in Fig. 1 are the roles of SEP. Our analysis of SEP did not rely on any origination constraints. Finally, if s were an indeterminate, then the responder role violates the Indeterminate Acquisition Principle. We instead interpret s as a basic value.

3. Skeletons and Homomorphisms

A *skeleton* $\mathbb{A}$ consists of (possibly partially executed) regular strands, i.e. a finite set of nodes, $\mathsf{nodes}(\mathbb{A})$, with two additional kinds of information:

1. A partial ordering $\preceq_{\mathbb{A}}$ on $\mathsf{nodes}(\mathbb{A})$;
2. Finite sets $\mathsf{unique}_{\mathbb{A}}, \mathsf{non}_{\mathbb{A}}$ of basic values assumed uniquely originating and respectively non-originating in $\mathbb{A}$.

More formally:

Definition 3.1 *A four-tuple* $\mathbb{A} = (\mathsf{nodes}, \preceq, \mathsf{non}, \mathsf{unique})$ *is a* preskeleton *if:*

1. nodes *is a finite set of regular nodes; moreover, for all* $n_1 \in \mathsf{nodes}$*, if* $n_0 \Rightarrow^+ n_1$ *then* $n_0 \in \mathsf{nodes}$*;*
2. $\preceq$ *is a partial ordering on* nodes *such that* $n_0 \Rightarrow^+ n_1$ *implies* $n_0 \preceq n_1$*;*
3. non *and* unique *are finite sets of basic values, and*
   (a) $\forall K \in \mathsf{non}\,.\,\forall n \in \mathsf{nodes}\,.\,K \not\sqsubseteq \mathsf{msg}(n)$*;*
   (b) $\forall K \in \mathsf{non}\,.\,\exists n \in \mathsf{nodes}\,.$ *either* $K \ll \mathsf{msg}(n)$ *or* $K^{-1} \ll \mathsf{msg}(n)$*; and*
   (c) $\forall a \in \mathsf{unique}\,.\,\exists n \in \mathsf{nodes}$ *s.t.* $a \sqsubseteq \mathsf{msg}(n)$.

A preskeleton $\mathbb{A}$ *is a* skeleton *if in addition:*

4. *for all* $a \in \mathsf{unique}$*, if* a *originates at* $n_0 \in \mathsf{nodes}$*, then*

(a) a *originates at no other node* $n_1 \in \mathsf{nodes}$*; and*
(b) *if* $a \sqsubseteq \mathsf{msg}(n_1)$ *where* $n_1 \in \mathsf{nodes}$*, then* $n_0 \preceq n_1$.

The parameters *of* $\mathbb{A}$ *form the union:* $\mathsf{params}(\mathbb{A}) = \bigcup_{n \in \mathsf{nodes}(\mathbb{A})} \mathsf{params}(n)$.

All of the "skeletons" in Figs. 2–10 are skeletons, except when we cheated on Clause 4b. Namely: In Fig. 2, the left hand node should precede the right hand node. In Fig. 6, the second skeleton $\mathbb{C}_{21}$ should have the node transmitting $\{|s|\}_k$ precede the listener node for s. In Fig. 7, the node transmitting t_0 should also precede the listener for k.

If $\mathbb{A}$ violates Clause 3a, no enrichment of $\mathbb{A}$ can satisfy it. By contrast, some preskeletons may violate Clause 4a or Clause 4b but enrich to a skeleton. Two nodes, at both of which some $a \in \mathsf{unique}$ originates, may map to a single node originating it, thereby satisfying Clause 4a. A strengthening of the ordering $\preceq$ may satisfy Clause 4b, while remaining acyclic. This works for the non-skeletons in Figs. 2, 6, and 7. We formalize these operations in Section 3.4.

3.1. Realized Skeletons

A skeleton is *realized* if its strands can really happen, without any other *regular* behavior. The regular behavior present in a realized skeleton is combined with some *adversary* behavior to explain how each message received by a regular node could have been generated by that time. The adversary behavior must not violate the skeleton's assumptions about non-origination and unique origination.

Penetrator Webs. We represent the actions of the adversary by means of strands of certain special forms. These originate basic values or indeterminates; compose and transmit a tuple, having received its components; separate and transmit the components of a tuple, having received the tuple; encrypt and transmit a ciphertext, having received an encryption key and a plaintext; and decrypt and transmit a plaintext, having received the matching decryption key and a ciphertext:

Definition 3.2 *An* adversary strand *has any of the following forms:*

M_a: $\langle +a \rangle$ *where* a *is a basic value*
M_g: $\langle +g \rangle$ *where* g *is an indeterminate*
C: $\langle -g \Rightarrow \cdots \Rightarrow -h \Rightarrow +tag\ g \hat{} \ldots \hat{} h \rangle$
S: $\langle -tag\ g \hat{} \ldots \hat{} h \Rightarrow +g \Rightarrow \cdots \Rightarrow +h \rangle$
E: $\langle -K \Rightarrow -h \Rightarrow +\{|h|\}_K \rangle$
D: $\langle -K^{-1} \Rightarrow -\{|h|\}_K \Rightarrow +h \rangle$

Because an adversary uses a key only by means of E and D strands, it can use a key only if it receives that key value. Since every value that is received must have been originated, it follows that an adversary can never use a non-originating value. This justifies our use of "non-originating" to model "uncompromised."

The adversary can link together a number of strands, forming an "adversary web" that accomplishes some compound task.

Definition 3.3 (Adversary web, derivable) *Let* $G = \langle \mathcal{N}_G, (\rightarrow_G \cup \Rightarrow_G) \rangle$ *be a finite acyclic graph, where (i)* $n_0 \Rightarrow n_1$ *only if* n_0, n_1 *are successive nodes on an adversary strand; and (ii)* $n \rightarrow m$ *only if* n *is a transmission node,* m *is a reception node, and* $\mathsf{msg}(n) = \mathsf{msg}(m)$. G *is an* adversary web *with support* S_{spt} *and result* R *if* S_{spt} *and* R *are sets of messages and moreover:*

1. *If* $n_2 \in \mathcal{N}_G$ *is a reception node, then either* $\mathsf{msg}(n_2) \in S_{\mathrm{spt}}$ *or there is a unique* n_1 *such that* $n_1 \rightarrow_G n_2$.
2. *If* $n_2 \in \mathcal{N}_G$ *and* $n_1 \Rightarrow n_2$ *then* $n_1 \Rightarrow_G n_2$.
3. *For each* $t \in R$, *either* $t \in S_{\mathrm{spt}}$ *or for some positive* $n \in \mathcal{N}_G$, $\mathsf{msg}(n) = t$.

If V *is a set of basic values, then term* t_1 *is* derivable from S_{spt} avoiding V *if there is a web* G *with support* $S_G \subseteq S_{\mathrm{spt}}$ *and* $t_1 \in R_G$, *where no basic value in* V *originates on a penetrator strand in* G.

This suggests that $\mathbb{A}$ is a possible execution if, for each reception node n, there exists an adversary web deriving $\mathsf{msg}(n)$ using messages transmitted earlier in $\mathbb{A}$, and avoiding basic values constrained by $\mathsf{non}_{\mathbb{A}}$ and $\mathsf{unique}_{\mathbb{A}}$. However, if $a \in \mathsf{unique}_{\mathbb{A}}$, but it does not originate on any node in $\mathbb{A}$, then it is in fact unconstrained: It can originate just once, but on an adversary M_a node, after which the adversary can use it as much as necessary. The constrained values in $\mathsf{unique}_{\mathbb{A}}$ are those that originate on a regular node of $\mathbb{A}$.

Definition 3.4 (Realized) *The* avoidance set $\mathsf{avoid}(\mathbb{A})$ *for* $\mathbb{A}$ *is the set*

$$\mathsf{non}_{\mathbb{A}} \cup (\mathsf{unique}_{\mathbb{A}} \cap \{a\colon a \textit{ originates at some } n \in \mathsf{nodes}_{\mathbb{A}}\}).$$

The support $\mathsf{support}(n, \mathbb{A})$ of $n \in \mathsf{nodes}_{\mathbb{A}}$ in $\mathbb{A}$ *is* $\{\mathsf{msg}(m)\colon m \prec_{\mathbb{A}} n$ *and* m *is a transmission node*$\}$.

A reception node $n \in \mathsf{nodes}_{\mathbb{A}}$ *is* realized in $\mathbb{A}$ *if* $\mathsf{msg}(n)$ *is derivable from* $\mathsf{support}(n, \mathbb{A})$ *avoiding* $\mathsf{avoid}(\mathbb{A})$. $\mathbb{A}$ *is* realized *if it is a skeleton and every reception node* $m \in \mathsf{nodes}_{\mathbb{A}}$ *is realized in* $\mathbb{A}$.

We have been saying informally that $\mathbb{A}$ represents a possible execution, and we now stipulate that this means that $\mathbb{A}$ is realized. The adversary webs explain how the adversary can generate values that will satisfy each reception node in $\mathbb{A}$.

Realized skeletons from Section 1 include $\mathbb{B}_{21}$ and $\mathbb{D}_1$. In the case of $\mathbb{B}_{21}$, the adversary web needed to "explain" its two reception nodes are each the empty web, since each reception node has a message that has already been transmitted on an earlier node. However, $\mathbb{D}_1$ requires an adversary web that decrypts $\{|\{|k|\}_{\mathsf{sk}(A)}|\}_{\mathsf{pk}(C)}$ using the decryption key $\mathsf{pk}(C)^{-1}$, and then re-encrypts with the public key $\mathsf{pk}(B)$, neither of which is to be avoided in $\mathbb{D}_{11}$. Skeleton $\mathbb{E}_1$ is again realized, using the empty adversary web. By an old result [20, Lemma 2.9]:

Lemma 3.5 *If every node* $m \in \mathsf{nodes}_{\mathbb{A}}$ *is realized in the preskeleton* $\mathbb{A}$, *then Defn. 3.1, Clause 4b is satisfied in* $\mathbb{A}$.

It is easy to check whether there is an adversary web G that realizes m from $P = \{n\colon n \prec_{\mathbb{A}} m$ and n is a transmission node$\}$. When all of the keys are basic values, we can bring the web to a *normal form* in the Prawitz-like sense [16] that constructive adversary strands never precede destructive ones [14,3,12]. When the keys may include compound messages, there is a weaker normal form: Every path within the web either traverses a key being used for encryption or decryption on an adversary strand, or else traverses destructive strands before any constructive strands. In such a web, every message is a member of the finite set:

$$\{t\colon t \ll \mathsf{msg}(m) \vee \exists n \in P\,.\, t \ll \mathsf{msg}(n)\}.$$

3.2. Homomorphisms

Our intuitive notion of enrichment will be formalized by homomorphisms among skeletons or preskeletons. A preskeleton homomorphism has two components. One is a homomorphism α on the underlying algebra M. It says how to transform the messages sent and received on nodes of the source preskeleton to messages sent and received on the target preskeleton. The other component is a map ϕ from nodes of the source preskeleton to nodes of the target preskeleton. The most important condition is that $\mathsf{msg}(\phi(n)) = \alpha(\mathsf{msg}(n))$; i.e. the message sent or received on the target node should be α applied to the message sent or received on the source node. The target may contain additional nodes not of the form $\phi(n)$.

Definition 3.6 *Suppose $\mathbb{A}, \mathbb{B}$ are preskeletons, α is a homomorphism on M, and $\phi\colon \mathsf{nodes}_{\mathbb{A}} \to \mathsf{nodes}_{\mathbb{B}}$. $H = [\phi, \alpha]$ is a* homomorphism *if*

1. *ϕ preserves strand structure:*
   (a) *If $s \downarrow i \in \mathbb{A}$ then there is an s' s.t. for all $j \leq i$, $\phi(s \downarrow j) = s' \downarrow j$;*
   (b) *n and $\phi(n)$ agree in direction, either transmission $+$ or reception $-$;*
2. *For all $n \in \mathbb{A}$, $\mathsf{msg}(\phi(n)) = \alpha(\mathsf{msg}(n))$;*
3. *Skeleton information preserved:*
   (a) *$n \preceq_{\mathbb{A}} m$ implies $\phi(n) \preceq_{\mathbb{B}} \phi(m)$;*
   (b) *$\alpha(\mathsf{non}_{\mathbb{A}}) \subseteq \mathsf{non}_{\mathbb{B}}$;*
   (c) *$\alpha(\mathsf{unique}_{\mathbb{A}}) \subseteq \mathsf{unique}_{\mathbb{B}}$;*
4. *Origination preserved for uniquely originating values: If $a \in \mathsf{unique}_{\mathbb{A}}$ and a originates at $n \in \mathsf{nodes}_{\mathbb{A}}$, then $\alpha(a)$ originates at $\phi(n)$.*

We write $H\colon \mathbb{A} \hookrightarrow \mathbb{B}$ when H is a homomorphism from $\mathbb{A}$ to $\mathbb{B}$. When α, α' agree on $\mathsf{params}(\mathbb{A})$, then $[\phi, \alpha] = [\phi, \alpha']$; i.e., $[\phi, \alpha]$ is the equivalence class of pairs under this relation. The source *of H is $\mathbb{A}$, and the* target *of H is $\mathbb{B}$.*

We sometimes write ϕ_H and α_H to refer to a ϕ and α such that $H = [\phi, \alpha]$.

H is an inclusion map *if ϕ_H is the identity function. In this case, α_H is also the identity, i.e. $H = [\phi_H, \mathsf{Id}]$. $\mathbb{A}$ is a* subskeleton *of $\mathbb{B}$ if there is an inclusion $H\colon \mathbb{A} \hookrightarrow \mathbb{B}$.*

We regard the finitely generated homomorphism α_0 which is the identity for all $v \notin \mathsf{params}(\mathbb{A})$ as the canonical representative of $[\phi, \alpha]$. Thus, henceforth, we will assume that the message homomorphism α in $[\phi, \alpha]$ is finitely generated.

The condition on origination in Clause 4 avoids the degeneracy in which a point of origination is destroyed for some basic value $a \in \mathsf{unique}_{\mathbb{A}}$. We stipulate that such degenerate maps are not homomorphisms.

Lemma 3.7 *If $\mathbb{A}$ is a preskeleton, then the identity $\mathsf{Id}_{\mathbb{A}}\colon \mathbb{A} \hookrightarrow \mathbb{A}$. If $H\colon \mathbb{A} \hookrightarrow \mathbb{B}$ and $J\colon \mathbb{B} \hookrightarrow \mathbb{C}$, then $J \circ H\colon \mathbb{A} \hookrightarrow \mathbb{C}$. Skeletons form a category under homomorphisms.*

Section 1 incorporates many examples of homomorphisms, starting with an absence of homomorphisms. The deadness of Fig. 2 asserts that there are no homomorphisms from $\mathbb{A}_0$ whose target skeleton is realized. Figs. 3–4 illustrate a pair of homomorphisms $\mathbb{B} \hookrightarrow \mathbb{B}_1$ and $\mathbb{B} \hookrightarrow \mathbb{B}_2$. In each of these homomorphisms, $\mathbb{B}$ is included into a larger

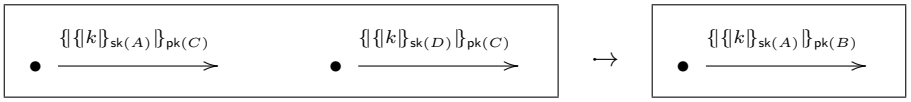


Figure 11. A non-injective homomorphism $\mathbb{A}_1 \rightarrow \mathbb{A}_2$

skeleton: the message homomorphism α is the identity and the node map ϕ is the identity in each case. The homomorphism $\mathbb{B}_2 \rightarrow \mathbb{B}_{21}$ is a bit more interesting, since its message homomorphism maps the names A, C to A and B, D to B. Its node map is a bijection. By composition, we also have a homomorphism $\mathbb{B} \rightarrow \mathbb{B}_{21}$.

The homomorphism $\mathbb{C} \rightarrow \mathbb{C}_{21}$ is an embedding, with a second embedding $\mathbb{C}_{21} \rightarrow \mathbb{C}_{211}$, and similar maps $\mathbb{D} \rightarrow \mathbb{D}_1$ and $\mathbb{E} \rightarrow \mathbb{E}_1$. We have not yet illustrated any case in which ϕ is non-injective. Fig. 11 is an artificial but simple example. A message homomorphism mapping A, D to the same value permits the two transmission nodes of $\mathbb{A}_1$ to be identified in the target $\mathbb{A}_2$. In this case, A, D are mapped to A, and C to B, but in fact the choice of target values is arbitrary. A renaming could change them to any other pair of distinct values, producing an isomorphic result.

In Fig. 11, there are also two homomorphisms H_1, H_2 in the opposite, right-to-left direction. One of them, say H_1, renames B to C and maps the single node of $\mathbb{A}_2$ to the left-hand node of $\mathbb{A}_1$. The other homomorphism H_2 renames B to C and also renames A to D; it maps the single node of $\mathbb{A}_2$ to the right-hand node of $\mathbb{A}_1$. These homomorphisms are, however, not essentially different. $\mathbb{A}_1$ has a non-trivial automorphism J, i.e. an isomorphism to itself. This is the map that interchanges the two nodes, while mapping A to D and D to A. In fact, $H_2 = J \circ H_1$, i.e. H_1 and H_2 differ by an isomorphism.

3.3. Characterizing Executions

An execution means a realized skeleton. We regard each skeleton $\mathbb{A}$ as describing a set of executions. This is the set of realized homomorphic images of $\mathbb{A}$.

A dead skeleton describes the empty set of executions. A realized skeleton describes a set of executions that includes itself. A skeleton that is neither dead nor realized describes a non-empty set that does not include itself.

Definition 3.8 *$\mathbb{A}$ is a* dead *skeleton if for every homomorphism $H\colon \mathbb{A} \rightarrow \mathbb{B}$, $\mathbb{B}$ is not realized.*

Lemma 3.9 *If $\mathbb{A}$ is dead and $H\colon \mathbb{A} \rightarrow \mathbb{B}$, then $\mathbb{B}$ is dead.*

Proof: If $J\colon \mathbb{B} \rightarrow \mathbb{C}$, then $J \circ H\colon \mathbb{A} \rightarrow \mathbb{C}$, so $\mathbb{C}$ is not realized. □

This lemma formalizes Principle 1.2.

Any non-dead skeleton $\mathbb{A}$ describes an infinite set of executions. Since $\mathbb{A}$ is non-dead, it has at least one homomorphism $H\colon \mathbb{A} \rightarrow \mathbb{B}$ with $\mathbb{B}$ realized. Moreover, we can always take the disjoint union $2\mathbb{B} = \mathbb{B} \cup \mathbb{B}'$ of the realized skeleton $\mathbb{B}$ with a renaming $\mathbb{B}'$ of itself. If the renaming is chosen to use new values that do not to overlap with the original ones, the disjoint union $2\mathbb{B} = \mathbb{B} \cup \mathbb{B}'$ will again be a realized skeleton. Clearly,

we can again rename $2\mathbb{B}$ to $(2\mathbb{B})'$, taking a union to obtain $4\mathbb{B} = 2\mathbb{B} \cup (2\mathbb{B})'$. We can repeat this process *ad infinitum*.

Since the set of realized skeletons contains so many members, we would like to focus on compact but informative ways to characterize this set. Given a skeleton $\mathbb{A}$, consider a set of homomorphisms $\mathcal{C}$ such that:

1. If $H \in \mathcal{C}$, then $H \colon \mathbb{A} \to \mathbb{B}$ where $\mathbb{B}$ is realized; and
2. If $J \colon \mathbb{A} \to \mathbb{C}$ where $\mathbb{C}$ is realized, then $J = K \circ H$ for some $H \in \mathcal{C}$ and some homomorphism K.

CPSA's goal is, given a "starting point" $\mathbb{A}$, to compute a such a set $\mathcal{C}$ for $\mathbb{A}$.

However, a key choice was to decide which set of this kind to select. For instance, one would prefer an algorithm that produces *finite* sets $\mathcal{C}$ rather than infinite ones when possible. However, there is another consideration. Fig. 11 illustrates that we may have a situation where $\mathbb{A}_1 \to \mathbb{A}_2 \to \mathbb{A}_1$; so should we then prefer the larger skeleton $\mathbb{A}_1$ or the smaller skeleton $\mathbb{A}_2$?

We have opted for the smaller skeleton $\mathbb{A}_2$. Why? Non-injective homomorphisms that map different nodes to the same result, such as the one in Fig. 11, are always informative. They tell us that two distinct strands in the source could have been the same, since they are the same in the target. However, injective homomorphisms may not be interesting. For instance, recall the realized skeletons $\mathbb{A}, 2\mathbb{A}, 4\mathbb{A}, \ldots$ we mentioned immediately after Lemma 3.9. The homomorphisms from each member in this sequence to later members do not tell us anything new. We opt to pack more information into the choice of $\mathcal{C}$, and less into the "onward" homomorphisms K that extend members of $\mathcal{C}$ to yield all other realized results. Hence, we use injective homomorphisms to relate shapes to their instances:

Definition 3.10

1. $H = [\phi, \alpha] \colon \mathbb{A} \to \mathbb{B}$ *is* node-injective *iff ϕ is an injective function* $\phi \colon \mathsf{nodes}(\mathbb{A}) \to \mathsf{nodes}(\mathbb{B})$.
2. $H \leq_n J$ *iff for some node-injective K, $J = K \circ H$.*
3. *A set $\mathcal{C}$ of homomorphisms is a* (node-injective) characterization *for $\mathbb{A}$ iff*
   (a) *If $H \in \mathcal{C}$, then $H \colon \mathbb{A} \to \mathbb{B}$ where $\mathbb{B}$ is realized; and*
   (b) *If $J \colon \mathbb{A} \to \mathbb{C}$ where $\mathbb{C}$ is realized, then $H \leq_n J$ for some $H \in \mathcal{C}$.*
4. *$\mathcal{C} \leq_n \mathcal{C}'$ iff for every $H \in \mathcal{C}$, there exists a $J \in \mathcal{C}'$ such that $H \leq_n J$.*
5. $\min(\mathcal{C}) = \{H \in \mathcal{C} \colon \forall J \in \mathcal{C} \,.\, J \leq_n H \text{ implies } H \leq_n J\}$.

CPSA's goal is, given an $\mathbb{A}$, to compute a minimum characterization for $\mathbb{A}$. We will write $\mathcal{C} \sim \mathcal{C}'$ iff every $H \in \mathcal{C}$ is isomorphic to some $H' \in \mathcal{C}'$ and vice versa. Then:

Lemma 3.11

1. *If $H \colon \mathbb{A} \to \mathbb{A}$ is node-injective, then H is an isomorphism from $\mathbb{A}$ to itself.*
2. *If $H \leq_n J \leq_n H$, then there is an isomorphism I such that $I \circ H = J$. Hence, $\leq_n$ is a partial order (to within isomorphism).*
3. *The relation $\leq_n$ is well founded (to within isomorphism).*
4. *A set S_H of homomorphisms is finite if (i) $K \in S_H$ implies $K \leq_n H$, and (ii) no two $K_1, K_2 \in S_H$ differ by an isomorphism.*
5. *$\{H \colon H \colon \mathbb{A} \to \mathbb{B} \ \wedge \ \mathbb{B}$ is realized$\}$ is a characterization for $\mathbb{A}$.*
6. *$\min(\mathcal{C})$ is a non-empty characterization for $\mathbb{A}$ if $\mathcal{C}$ is.*

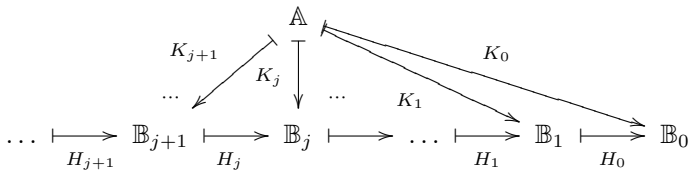
7. *If $\mathcal{C}, \mathcal{C}'$ are characterizations for* $\mathbb{A}$, $\min(\mathcal{C}) \sim \min(\mathcal{C}')$.

Proof: 1. If $H = [\phi, \alpha] \colon \mathbb{A} \rightarrowtail \mathbb{A}$ is injective, then by the finiteness of $\mathsf{nodes}(\mathbb{A})$, ϕ is a bijection. Since $\mathsf{nodes}(\mathbb{A}) \times \mathsf{nodes}(\mathbb{A})$ is finite, H does not introduce new pairs into the ordering. That is, $\phi(n) \preceq \phi(m)$ implies $n \preceq m$.

So we must show next that the message algebra homomorphism α is invertible. However, by Lagrange's theorem, there is a k such that the k^{th} iterate ϕ^k of ϕ is the identity. If $k = 1$, then α is the identity. If $k = j + 1$, then the j^{th} iterate α^j of α inverts α.

2. If $H \leq_n J \leq_n H$, then we have $J = G_1 \circ H$ and $H = G_2 \circ J$, where G_1, G_2 are node-injective. Hence $G_2 \circ G_1$ is a node-injective homomorphism from the target of H to itself. By Clause 1, $G_2 \circ G_1$ has an inverse F, i.e. $F \circ (G_2 \circ G_1)$ is the identity. Thus, by associativity of composition, $F \circ G_2$ is the desired inverse to G_1.

3. Let $\ldots K_{j+1} \leq_n K_j \leq_n \ldots K_0$ be an infinite descending chain of homomorphisms in the $\leq_n$ ordering, as in the following diagram, where each H_i is node-injective:



We want to ensure that all but finitely many of the H_i are isomorphisms. By Lemma 2.7, Clause 4, all but finitely many of the α_{K_i} differ from each other by renamings. Hence all but finitely many of the α_{H_i} are isomorphisms. For convenience, assume for some N_0, that all α_{H_i} with $i > N_0$ are the identity. Thus, all of the H_i for $i > N_0$ are inclusion maps. But a skeleton has only finitely many subskeletons.

4. Suppose that S_H satisfies (i). Let $K_1, K_2 \leq_n H$, where $H \colon \mathbb{A} \rightarrowtail \mathbb{C}$ and $K_i = [\phi_i, \alpha_i] \colon \mathbb{A} \rightarrowtail \mathbb{B}_i$.

Write $R(K_1, K_2)$ to mean: (a) $\mathsf{ran}(\phi_1) \subseteq \mathsf{ran}(\phi_2)$; (b) $\phi_1(\preceq_1) \subseteq \phi_2(\preceq_2)$; (c) $\alpha_1(\mathsf{non}_{\mathbb{B}_1}) \subseteq \alpha_2(\mathsf{non}_{\mathbb{B}_2})$; and (d) $\alpha_1(\mathsf{unique}_{\mathbb{B}_1}) \subseteq \alpha_2(\mathsf{unique}_{\mathbb{B}_2})$.

R is a preorder, and when $R(K_1, K_2)$, there is an injective map ψ from $\mathsf{nodes}(\mathbb{B}_1)$ to $\mathsf{nodes}(\mathbb{B}_2)$ defined by $\psi(n) = \phi_2^{-1}(\phi_1(n))$. If also $R(K_2, K_1)$, ψ is bijective.

Let $S(K_1) = \{K_2 \colon R(K_1, K_2) \wedge R(K_2, K_1)\}$. Since there are only finitely many R-equivalence classes $S(K)$, it suffices to show that if any of them is infinite, then it contains isomorphic members.

Let's say that a skeleton $\mathbb{D}$ is *clean* if each strand in $\mathbb{D}$ results from a role by a renaming, and all of these renamings have disjoint ranges.

For each member $\mathbb{B}$ of $S(K_1)$, there is a clean skeleton $\mathbb{D}$ and a message homomorphism β such that such that $\mathbb{B} = \beta(\mathbb{D})$. Moreover, by a cardinality argument, all of $S(K_1)$ consists of images of a finite number of clean $\mathbb{D}$. Thus, if $S(K_1)$ is infinite, there is a single $\mathbb{D}$ with infinitely many $\mathbb{B} \in S(K_1)$ such that there exists a β such that $\mathbb{B} = \beta(\mathbb{D})$. However, since for some $J \colon \mathbb{D} \rightarrowtail \mathbb{C}$, each such $\beta \leq_s \alpha_J$, by Lemma 2.7, Clause 4, so that this set of message homomorphisms has isomorphic members $\iota \circ \beta_1 = \beta_2$. Hence $[\psi, \iota]$ is an isomorphism between skeletons $\mathbb{B}_i$.

(Clause 3 also follows directly from Clause 4.)

5. Clause 3a of Def. 3.10 is immediate. Clause 3b holds because $J \leq_n J$.

6. Non-emptiness of $\min(\mathcal{C})$ follows from well-foundedness. Observe that if $H \in \mathcal{C} \setminus \min(\mathcal{C})$, then there is some $J \in \min(\mathcal{C})$ such that $J \leq_n H$. Since $\mathcal{C}$ is a characterization, for any $K\colon \mathbb{A} \to \mathbb{C}$ with $\mathbb{C}$ realized, there is a $H \in \mathcal{C}$ such that $H \leq_n K$. If $H \notin \min(\mathcal{C})$, then there is some $J \in \min(\mathcal{C})$ with $J \leq_n H \leq_n K$, so $J \leq_n K$. Thus, $\min(\mathcal{C})$ is a characterization.

7. If either $\mathcal{C}$ or $\mathcal{C}'$ is empty, then so is the other (i.e. $\mathbb{A}$ is dead); in this case the result is immediate. So assume both non-empty. Since $\min(\mathcal{C})$ is a characterization, for every $J \in \min(\mathcal{C}')$ there is a $H \in \min(\mathcal{C})$ such that $H \leq_n J$. Since $\min(\mathcal{C}')$ is a characterization, there is also a $K \in \min(\mathcal{C}')$ such that $K \leq_n H \leq_n J$. By the definition of $\min$, $J \leq_n K$. Hence H, J differ by an isomorphism. Symmetry of $\min(\mathcal{C}), \min(\mathcal{C}')$ completes the proof.

□

This establishes that the CPSA goal, to compute the minimum characterization, is well-defined to within isomorphism.

Definition 3.12 *The* shapes *for a skeleton $\mathbb{A}$, written* $\mathsf{shapes}(\mathbb{A})$*, are the members of the nodewise minimum characterization for $\mathbb{A}$.*

The $\mathsf{shapes}(\mathbb{A})$ form a well-defined set of homomorphisms (to within isomorphism), since we may apply Lemma 3.11, Clauses 6–7 to the characterization $\mathcal{C}$ containing all homomorphisms from $\mathbb{A}$ to realized skeletons. $\mathbb{A}$ is dead iff $\mathsf{shapes}(\mathbb{A}) = \emptyset$. We may now justify Principle 1.4 directly from the definitions.

Lemma 3.13 *Suppose $H\colon \mathbb{A} \to \mathbb{B}$, and $J\colon \mathbb{B} \to \mathbb{D}$ where $\mathbb{D}$ is realized. Then $J \circ H = L \circ K$ for some $K\colon \mathbb{A} \to \mathbb{S}$ with $K \in \mathsf{shapes}(\mathbb{A})$, and some node-injective $L\colon \mathbb{S} \to \mathbb{D}$.*

3.4. The Hull of a Preskeleton

If $\mathbb{A}$ is a preskeleton but not a skeleton, then for some $a \in \mathsf{unique}_{\mathbb{A}}$, either

1. a originates at two or more nodes (Def. 3.1, Cl. 4a); or else
2. a originates at n_0 and $a \sqsubseteq \mathsf{msg}(n_1)$, although $n_0 \not\preceq_{\mathbb{A}} n_1$ (Def. 3.1, Cl. 4b).

In this subsection, we say how to "fix" those situations. When they can be fixed at all, there is a single, canonical, most general way to do so.

A map f is *universal* in some set of maps F if $f \in F$ and, for every $f' \in F$, there is exactly one g such that f' is of the form $f' = g \circ f$.

Lemma 3.14 *Suppose $\mathbb{A}, \mathbb{B}$ are preskeletons, with $H\colon \mathbb{A} \to \mathbb{B}$.*

1. *If $\gamma \leq_s \alpha_H$, then there is a $\mathbb{B}_0$ and a $G\colon \mathbb{A} \to \mathbb{B}_0$ such that G is universal among all homomorphisms K with source $\mathbb{A}$ where $\gamma \leq_s \alpha_K$.*
2. *Suppose that $\preceq_{\mathbb{A}} \subseteq \preceq_1$ and $\phi_H(\preceq_1) \subseteq \preceq_{\mathbb{B}}$. Then there is a $\mathbb{B}_0$ and a $G\colon \mathbb{A} \to \mathbb{B}_0$ such that G is universal among all homomorphisms $K\colon \mathbb{A} \to \mathbb{B}_1$ where $\phi_K(\preceq_1) \subseteq \preceq_{\mathbb{B}_1}$.*
3. *If $\phi_H(n_0) = \phi_H(n_1)$ for $n_0, n_1 \in \mathsf{nodes}(\mathbb{A})$, then there is a $\mathbb{B}_0$ and a $G\colon \mathbb{A} \to \mathbb{B}_0$ such that G is universal among all homomorphisms from $\mathbb{A}$ which identify n_0 and n_1.*

Proof: 1. Define $\mathsf{nodes}(\mathbb{B}_0)$ by applying γ to each strand s that contributes to $\mathbb{A}$, and let ψ be the bijection that maps each node $s \downarrow i \in \mathsf{nodes}(\mathbb{A})$ to $\gamma(s) \downarrow i$. Let $\preceq_{\mathbb{B}_0} = \psi(\preceq_{\mathbb{A}})$, $\mathsf{non}_{\mathbb{B}_0} = \gamma(\mathsf{non}_{\mathbb{A}})$, and $\mathsf{unique}_{\mathbb{B}_0} = \gamma(\mathsf{unique}_{\mathbb{A}})$.

Then $\mathbb{B}_0$ is a preskeleton unless Def. 3.1, Clause 3a fails. However, if $a \sqsubseteq \mathsf{msg}(n) \in \mathsf{nodes}(\mathbb{B}_0)$ but $a \in \gamma(\mathsf{non}_{\mathbb{A}})$, then this property is preserved under composition. Thus, since $\gamma \leq_s \alpha_H$, and $\mathbb{B}$ satisfies Clause 3a, so does $\mathbb{B}_0$.

Moreover, $[\psi, \gamma]$ is a homomorphism unless Def. 3.6, Clause 4 fails. However, if a originates on $s \downarrow i$, but $\gamma(a) \sqsubseteq \mathsf{msg}(\psi(s \downarrow j))$ for $j < i$, then $\alpha(a)_H \sqsubseteq \mathsf{msg}(\phi_H(s \downarrow j))$, contradicting the assumption that H is a homomorphism.

2. Define $\mathbb{B}_0$ to be the same as $\mathbb{A}$, except that the ordering is $\preceq_1$. This ordering is acyclic because its image under ϕ_H is acyclic.

3. $n_0 = s_0 \downarrow i$ and $n_1 = s_1 \downarrow i$, since if the two nodes lay at different indices, no homomorphism could identify them. The messages $\mathsf{msg}(s_0 \downarrow 1), \ldots, \mathsf{msg}(s_0 \downarrow i)$ are simultaneously unifiable with $\mathsf{msg}(s_1 \downarrow 1), \ldots, \mathsf{msg}(s_1 \downarrow i)$ resp., since α_H equates them. Let γ be their simultaneous m.g.u. Apply Clause 1 to this γ, obtaining $G_0 = [\psi_0, \gamma] \colon \mathbb{A} \rightarrow \mathbb{B}_0$.

By Clause 2, we may extend the ordering $\preceq_{\mathbb{B}_0}$ so that $s_0 \downarrow j$ precedes (succeeds) every node that $s_1 \downarrow j$ precedes (succeeds), and vice versa.

We now construct $\mathbb{B}_1$ by selecting the strand $\gamma(s_0)$. Let ψ_1 extend ψ_0 by mapping the nodes $s_1 \downarrow j$ to $s_0 \downarrow j$, discarding the unnecessary nodes. $G_1 = [\psi_1, \gamma]$ is the desired homomorphism. □

Lemma 3.14 is used in the next proof, and also repeatedly in Section 6.

Lemma 3.15 (Hull) *If $H \colon \mathbb{A} \rightarrow \mathbb{B}$ and $\mathbb{B}$ is a skeleton, then there is a $G_{\mathbb{A}} \colon \mathbb{A} \rightarrow \mathbb{B}_0$, such that $G_{\mathbb{A}}$ is universal among homomorphisms from $\mathbb{A}$ to skeletons.*

Proof: If $\mathbb{A}$ is a preskeleton but not a skeleton, then there is a counterexample either to Def. 3.1, Clause 4a or else to Def. 3.1, Clause 4b. In the first case, there are two nodes n_0, n_1 at both of which the same $a \in \mathsf{unique}_{\mathbb{A}}$ originates. By Def. 3.6, Clause 4, $\alpha_H(a)$ originates at $\phi_H(n_0)$ and at $\phi_H(n_1)$. Since $\mathbb{B}$ is a skeleton, $\phi_H(n_0) = \phi_H(n_1)$. Thus, we may apply Lemma 3.14, Clause 3.

In the second case, for some $a \in \mathsf{unique}_{\mathbb{A}}$, $a \sqsubseteq \mathsf{msg}(n_1)$ but with the origin n_0 of a, $n_0 \not\preceq_{\mathbb{A}} n_1$. In this case, we apply Lemma 3.14, Clause 2.

If the result of a step is not a skeleton, we iterate; however, we must terminate: At each step of the first kind, we reduce the number of nodes. At each step of the second kind, we reduce the number of incomparable nodes. □

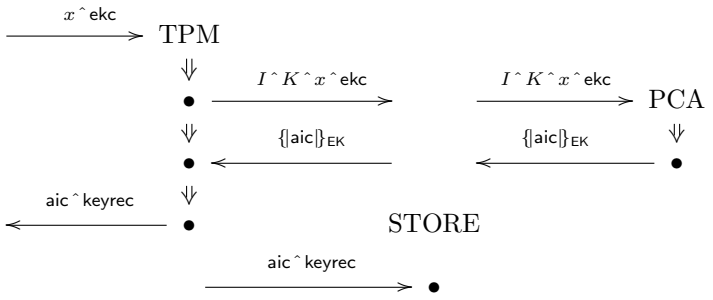
Definition 3.16 *The* hull *of preskeleton $\mathbb{A}$, written $\mathsf{hull}(\mathbb{A})$, is the universal map $G_{\mathbb{A}}$ given in Lemma. 3.15, when it exists.*

We sometimes use *hull* to refer to the skeleton $\mathbb{B}_0$, target of $\mathsf{hull}(\mathbb{A}) \colon \mathbb{A} \rightarrow \mathbb{B}_0$.[3]

4. Attestation Identity Protocol

In Section 1, we examined SEP to extract a number of search principles, one of which, Principle 1.3, concerns receiving an encrypted unit. Unfortunately, however, Principle 1.3

[3] The hull idea is due to Javier Thayer, as was the first proof of Lemma 3.15.



$$\mathsf{ekc} = \{\!|ekc\ \mathsf{MF}\hat{}\mathsf{EK}|\!\}_{\mathsf{sk(MF)}} \qquad \mathsf{aic} = \{\!|aic\ I\hat{}K\hat{}x|\!\}_{\mathsf{sk(PCA)}}$$
$$\mathsf{keyrec} = \{\!|aikrec\ K, K^{-1}|\!\}_{\mathsf{SRK}}$$

Figure 12. Modified Anonymous Identity Protocol MAIP

is not strong enough as formulated. It does not cover all the transformations that protocols apply to encrypted units, but only the most fundamental transformation, the act of creating the encrypted unit in the first place.

In this section, we will examine a second example to motivate a strengthening of our Principle 1.3, namely the Trusted Computing Group's protocol for generating certificates for "Attestation Identity Keys" (AIKs) [1]. These signature keys are intended to be resident within a Trusted Platform Module (TPM), and never to be available outside it. The certificate for an AIK verification key K ensures that the private signature part K^{-1} is resident in *some* TPM, without allowing the recipient to determine which one. They provide, thus, anonymous assurance that signatures were prepared within some TPM.

AIP Roles. The *privacy certificate authority* that prepares certificates on AIKs will prepare a certificate for any key K presented in a well-formatted message. So how does it ensure that the private part K^{-1} is TPM-resident? It encrypts the certificate aic using a public encryption key EK. That key is accompanied by a certificate from the TPM's manufacturer saying that the matching decryption key EK^{-1} is itself a long-term TPM-resident value. A TPM liberates the AIK certificate from this encryption only if it holds the signature key K^{-1} matching the key K in the certificate. The (slightly modified) protocol is shown in Fig. 12.

Non-Origination Assumptions. We associate two non-origination assumptions with the PCA's transmission node in this protocol.

1. When the PCA accepts an endorsement key certificate $\{\!|ekc\ \mathsf{MF}\hat{}\mathsf{EK}|\!\}_{\mathsf{sk(MF)}}$, it must check that the signing key is known to be the signature key of a recognized manufacturer. We model this by adding $\mathsf{sk}(\mathsf{MF})$ to the keys assumed to be non-originating.
2. The point of the ekc is to vouch that the private part EK^{-1} is TPM-resident, and therefore used only in accordance with the rules. Hence, we also add EK^{-1} to the keys assumed to be non-originating.

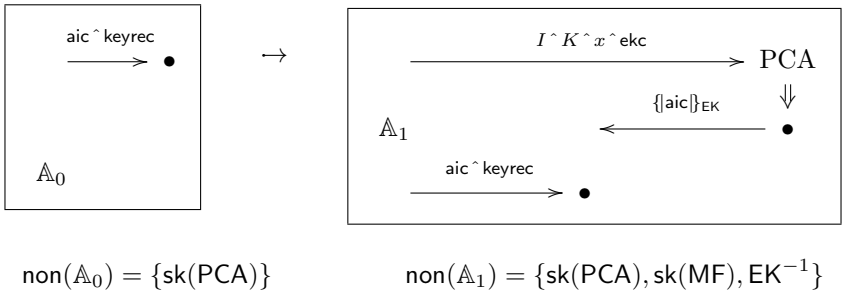


Figure 13. PCA Analysis, step 1 (Point of view: Store)

If ρ is the PCA role, we can express this assumption in the form

$$\mathsf{role\_non}(\rho \downarrow 2) = \{\mathsf{sk}(\mathsf{MF}), \mathsf{EK}^{-1}\}.$$

A node of a role may also have some associated values that are guaranteed to be uniquely originating, which we express with $\mathsf{role\_unique}(\rho \downarrow i)$. For instance, the session key transmitted by a key server should often be handled in this way.

AIP Transformations. In MAIP, there are two central transformations. The job of constructing and emitting an aic is one "transformation," which can be performed only by the privacy certifying authority. However, it is equally essential to the working of the protocol, that the PCA emits the aic only encrypted, and in such a way that the aic can be decrypted and transmitted in usable form only by a genuine TPM.

Of these two transformations, the first is certainly an instance of Principle 1.3. The value $\{|aic\ I \hat{} K \hat{} x|\}_{\mathsf{sk}(\mathsf{PCA})}$ is emitted, without having been contained as an ingredient of any previous node. Thus, if we assume that the signature key of PCA is uncompromised, any execution containing an instance of the STORE role must also contain a matching instance of the PCA role, as shown in Fig. 13.[4]

However, the TPM's transformation to free the aic from its encryption is not an instance of Principle 1.3. The digitally signed unit must be received before being retransmitted. Thus, Principle 1.3, Clause 2 cannot apply. Moreover, Principle 1.1 does not apply. The AIK K may be a freshly chosen value, but it has already been transmitted outside all encryptions at the time that the PCA receives it. So Principle 1.1 implies nothing.

What we need here is an analog to Principle 1.1, but applying to encryptions rather than to fresh values. It needs one additional case, to cover the possibility that the adversary could independently generate the encryption. Thus, it would take the form:

Principle 4.1 (The Encryption Test) *Suppose that $e = \{|t|\}_K$, is an encryption, and e is found in some message received in a skeleton $\mathbb{A}$ at a node n_1. Moreover, suppose that, in the message of n_1, e is found outside all of a number of encrypted forms $\{|t_1|\}_{K_1}, \ldots, \{|t_j|\}_{K_j}$. Then in any enrichment $\mathbb{B}$ of $\mathbb{A}$ such that $\mathbb{B}$ is a possible execution, either:*

[4] Observe that in $\mathbb{A}_1$ we have added $\mathsf{sk}(\mathsf{MF}), \mathsf{EK}^{-1}$ to the keys assumed non-originating, in accord with the origination constraint we associated with the PCA role.

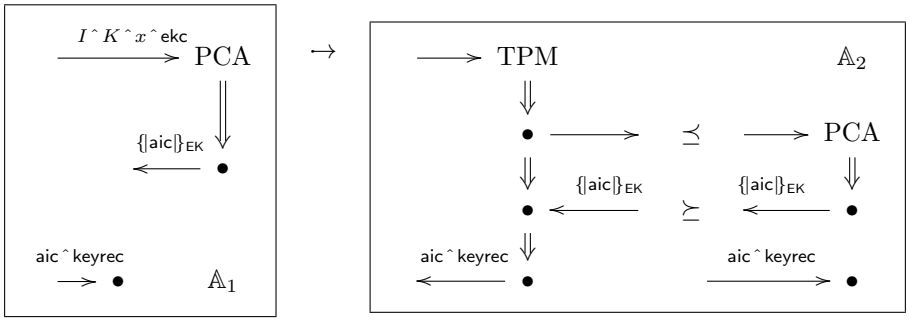


Figure 14. PCA Analysis, step 2 (Point of view: Store)

1. *One of the matching decryption keys K_i^{-1} is disclosed before n_1 occurs, so that e could be extracted by the adversary; or else*
2. *The encryption key K is disclosed before n_1 occurs, so that the adversary could construct $e = \{|t|\}_K$ from t; or else*
3. *Some regular strand contains a node m_1 in which e is transmitted outside the forms $\{|t_1|\}_{K_1}, \ldots, \{|t_j|\}_{K_j}$, but in all previous nodes $m_0 \Rightarrow^+ m_1$, e was found (if at all) only within the plaintexts $t_1 \ldots t_j$. Moreover, m_1 occurs before n_1.*

We apply this principle to the encryption $e = \{|aic\ I \hat{} K \hat{} x|\}_{\mathsf{sk(PCA)}}$, with the single encrypted form $\{|\mathsf{aic}|\}_{\mathsf{EK}}$. If we assume that the signature key $\mathsf{sk}(\mathsf{PCA})$ is uncompromised, as well as the TPM-resident value EK^{-1}, then the first two disjuncts are inapplicable, and we are left with the regular TPM strand that transforms the aic from the form $\{|\mathsf{aic}|\}_{\mathsf{EK}}$ to aic.

Principle 1.3 is, however, a special case of Principle 4.1. If $j = 0$ in the list of encrypted forms $\{|t_1|\}_{K_1}, \ldots, \{|t_j|\}_{K_j}$—so that this is the empty list—then the first disjunct is unsatisfiable. Moreover, in the last disjunct, no earlier occurrences of e are permitted. Hence, the old principle is nothing but the $j = 0$ case.

Indeed, now Principles 1.1 and 4.1 are in essentially the same form. The only differences are that (1) the "critical ingredient" is a uniquely originating basic value c in Principle 1.1 and an encryption $e = \{|t|\}_K$ in Principle 4.1, and (2) the possibility that K becomes compromised is relevant only in Principle 4.1. This suggests that we combine them, which we shall do after a few definitions, in the form of Def. 5.3 and Thm. 5.5.

5. The Authentication Tests

We regard Principles 1.1 and 4.1 as specifying how certain *tests* can be solved. In each one, the critical value c or e is found only inside a number of encryptions $S = \{\{|t_1|\}_{K_1}, \ldots, \{|t_j|\}_{K_j}\}$, and is subsequently received at node n_1 outside of these forms S. The test is to explain how it is extracted from S. We call S the *escape set*, since the critical value does escape from it; indeed, it has done so before being received at n_1.

The solutions are of *two kinds*: Either a key is compromised, so the *adversary* can create an occurrence of c outside S, or else a *regular strand* has a transmission node m_1

which transmits c or e outside S, although earlier nodes on the same strand contained the critical value only within S (if at all). There are only finitely many roles in Π, and the instances of their nodes yield all the candidates for regular solution nodes m_1. Later, in Section 6, we will also regard message homomorphisms as yielding solutions of a *third kind*, since they can make the test disappear. That is, the image of c no longer escapes from the image of S. We formalize "being contained within S" as follows:

Definition 5.1 *Let S be a set of encryptions. A message t_0* is found only within S in t_1, *written $t_0 \odot^S t_1$, iff for every path p such that $p(t_1) = t_0$, either (1) p traverses a key edge or else (2) p traverses a member of S before its end.*

Message t_0 is found outside S in t_1, *written $t_0 \dagger^S t_1$, iff not $(t_0 \odot^S t_1)$.*

Equivalently, $t_0 \dagger^S t_1$ iff for some path p, (1) $p(t_1) = t_0$, (2) p traverses no key edge, and (3) p traverses no $e \in S$ before its end. Thus, $t_0 \sqsubseteq t_1$ iff $t_0 \dagger^{\emptyset} t_1$.

For instance, let $t_0 = \{|\{|k|\}_{\mathsf{sk}(A)}|\}_{\mathsf{pk}(B)}$, and let $S_0 = \{t_0\}$ and $S_1 = \{\{|k|\}_{\mathsf{sk}(A)}\}$. The sole path $\langle \ell, \ell \rangle$ to k in t_0 traverses first t_0 and then $\{|k|\}_{\mathsf{sk}(A)}$, so

$$k \odot^{S_0} t_0 \quad \text{and} \quad k \odot^{S_1} t_0.$$

We used S_0 in Section 1.1, taking A's point of view in SEP. We used S_1 in Section 1.2, taking B's point of view. Moreover, for every S, $k \odot^S \{|s|\}_k$, because the only path to k traverses a key edge. However, $\{|k|\}_{\mathsf{sk}(A)} \dagger^{\emptyset} t_0$.

Looking at examples from MAIP next, and letting $S_2 = \{\{|\mathsf{aic}|\}_{\mathsf{EK}}\}$, we have

$$\mathsf{aic} \dagger^{\emptyset} \mathsf{aic} \quad \text{and} \quad \mathsf{aic} \dagger^{\emptyset} \{|\mathsf{aic}|\}_{\mathsf{EK}} \quad \text{but} \quad \mathsf{aic} \odot^{S_2} \{|\mathsf{aic}|\}_{\mathsf{EK}}.$$

5.1. Cuts and Tests

Definition 5.2 *A message c is* $\mathbb{A}$-critical *iff c is an encryption, or else $c \in \mathsf{unique}_{\mathbb{A}}$ and there is a node $n \in \mathsf{nodes}(\mathbb{A})$ such that c originates at n.*

When c is $\mathbb{A}$-critical, and S is a set of encryptions, then $\mathsf{Cut}(c, S, \mathbb{A})$, *the* cut for c, S *in $\mathbb{A}$, is defined and equal to:*

$$\mathsf{Cut}(c, S, \mathbb{A}) = \{n \in \mathsf{nodes}(\mathbb{A}) \colon \exists m \,.\, m \preceq_{\mathbb{A}} n \wedge c \dagger^S \mathsf{msg}(m)\}.$$

Thus, in Fig. 2, again letting $\{\{|\{|k|\}_{\mathsf{sk}(A)}|\}_{\mathsf{pk}(B)}\} = S_0$,

$$\mathsf{Cut}(k, S_0, \mathbb{A}_0) = \{ \bullet \xleftarrow{k} \},$$

i.e. the listener node at the right. In Fig. 3, the cut consists of the lower node:

$$\mathsf{Cut}(\{|s|\}_k, \emptyset, \mathbb{B}) = \{ \bullet \xleftarrow{\{|s|\}_k} \}.$$

In Fig. 4, for both skeletons $\mathbb{B}_1$ and $\mathbb{B}_2$, we were interested in the test $\mathsf{Cut}(k, S_0, \mathbb{B}_i)$. In Figs. 6–7, it is $\mathsf{Cut}(s, S_3, \mathbb{C}_i)$, with $S_3 = \{\{|s|\}_k\}$. The cuts driving the MAIP analysis, shown in Figs. 13–14, are (again with $S_2 = \{\{|\mathsf{aic}|\}_{\mathsf{EK}}\}$) $\mathsf{Cut}(\mathsf{aic}, \emptyset, \mathbb{A}_0)$ and $\mathsf{Cut}(\mathsf{aic}, S_2, \mathbb{A}_0)$.

Definition 5.3 *A node $n_1 \in Q = \mathsf{Cut}(c, S, \mathbb{A})$, is* solved *in Q, iff*

1. *for some transmission node $m_1 \in Q$, $m_1 \preceq_{\mathbb{A}} n_1$; or else*
2. *there is a listener node $m_1 = \mathsf{Lsn}[K]$ with $m_1 \prec_{\mathbb{A}} n_1$, and either*
   (a) *$c = \{\!|t_0|\!\}_K$, or else*
   (b) *for some $\{\!|t_0|\!\}_{t_1} \in S$, $K = t_1^{-1}$ is the matching decryption key.*

We also say that n_1, Q is solved in the cut Q if $n_1 \notin Q$. Q is solved *iff for all $n_1 \in Q$, n_1 is solved in Q. n_1, Q is a* test *if n_1 is unsolved in Q. A* solution *for n_1 Q is a transmission or listener node m_1 satisfying clause 1 or 2.*

In skeleton $\mathbb{A}_0$ (Fig. 2) there is no way to add a solution to $\mathsf{Cut}(k, S_0, \mathbb{A}_0)$, which showed that $\mathbb{A}_0$ is dead. In any homomorphic image of $\mathbb{A}_0$, the image of $\mathsf{Cut}(k, S_0, \mathbb{A}_0)$ remains unsolved. In Fig. 4, the solution to $\mathsf{Cut}(k, S_0, \mathbb{B}_1)$ is an instance of Clause 2a, while the solution to $\mathsf{Cut}(k, S_0, \mathbb{B}_2)$ is an instance of Clause 1. The solutions to the MAIP cuts $\mathsf{Cut}(\mathsf{aic}, \emptyset, \mathbb{A}_1)$ and $\mathsf{Cut}(\mathsf{aic}, S_2, \mathbb{A}_2)$ are both instances of Clause 1 (Figs. 13–14).

Solved cuts are derivable using adversary webs (Def. 3.3):

Lemma 5.4 ([6, Prop. 4]) *Let n_1 be a reception node in $\mathbb{A}$, and suppose that for every cut $Q = \mathsf{Cut}(c, S, \mathbb{A})$, $n_1 \in Q$ implies n_1 is solved in Q. Then there exists an adversary web G deriving $\mathsf{msg}(n_1)$ from $\mathsf{support}(n_1, \mathbb{A})$ avoiding $\mathsf{avoid}(\mathbb{A})$.*

Proof: Let $P = \mathsf{support}(n_1, \mathbb{A})$. The proof is by structural induction on the pair $P, \mathsf{msg}(n_1)$, i.e. induction on the well-founded relation that holds between P_1, t_1 and P_2, t_2 when $t_1 \sqsubseteq t_2$ and, for every $t \in P_1$, there is a $t' \in P_2$ such that $t \sqsubseteq t'$.

The induction hypothesis is that for every P_1, t_1 below $P, \mathsf{msg}(n_1)$ in this ordering, t_1 is derivable from P_1 using a web that avoids $\mathsf{avoid}(\mathbb{A})$.

Case $\mathsf{msg}(n_1) = a$**:** If the basic value $a \notin \mathsf{avoid}(\mathbb{A})$, then the one-node web that originates a suffices. If $a \in P$, then the empty web suffices.

Since $\mathsf{msg}(n_1) = a$, by the definition of skeleton, $a \notin \mathsf{non}_{\mathbb{A}}$. Thus, assume $a \in \mathsf{unique}_{\mathbb{A}}$ and a originates before n_1 in $\mathbb{A}$, and the subset $P^a = \{t \in P : a \sqsubseteq t\}$ is non-empty. If some concatenation $t_0 \hat{} t_1 \in P^a$, then apply the induction hypothesis to $(P^a \setminus \{t_0 \hat{} t_0'\}) \cup \{t_0\} \cup \{t_0'\}$. This asserts the existence of a penetrator web G_a deriving a. Obtain the desired web by prepending a separation S-strand above any occurrences of t_0 and t_0' in G_a.

Otherwise, P^a consists entirely of encryptions, and $\mathsf{Cut}(a, P^a, \mathbb{A})$ is well-defined, and by the assumption solved. By the definition of P^a, no transmission node $m_1 \preceq n_1$ can have $a \dagger^{P^a} \mathsf{msg}(m_1)$. Thus, there is a decryption key K^{-1} such that $\mathsf{Lsn}[K^{-1}] \preceq n_1$ and some $\{\!|t|\!\}_K \in P^a$. Apply the induction hypothesis to $(P^a \setminus \{\!|t|\!\}_K) \cup \{t\}$. This yields a web G_a deriving a. Obtain the desired web by prepending a decryption D-strand above any occurrences of t in G_a.

Case $\mathsf{msg}(n_1) = t_0 \hat{} t_1$**:** Apply the induction hypothesis to P, t_0 and P, t_1, obtaining a pair of webs G_0, G_1 deriving the two pieces. Obtain the desired web by appending a separation C-strand at the bottom to derive $t_0 \hat{} t_1$.

Case $\mathsf{msg}(n_1) = \{\!|t_0|\!\}_K$**:** If $\{\!|t_0|\!\}_K \in P$, then the empty web suffices. If P contains any concatenations, we reduce them as before.

Thus, assume P consists only of encryptions. Letting $P^e = \{t \in P : \{|t_0|\}_K \sqsubseteq t\}$, $\mathsf{Cut}(\{|t_0|\}_K, P, \mathbb{A})$ is well-defined, and since $\{|t_0|\}_K \notin P^e$, it has no solution by a transmission node. Hence, either $\mathsf{Lsn}[K] \preceq n_1$ or else $\mathsf{Lsn}[K_1^{-1}] \preceq n_1$ where some $\{|t_1|\}_{K_1} \in P^e$.

In the first case, apply the induction hypothesis to P^e, t_0, obtaining a web G. Obtain the desired web by appending an encryption E-strand using K to obtain $\{|t_0|\}_K$. In the second case, apply the induction hypothesis to $P^e \setminus \{|t_1|\}_{K_1} \cup \{t_1\}$ and $\{|t_0|\}_K$, and prepend a decryption D-strand above any uses of t_1. □

Theorem 5.5 (Authentication Test Principle, [6, Prop. 5]) 1. *If every cut in $\mathbb{A}$ is solved, then $\mathbb{A}$ is realized.*

2. *If $\mathbb{A}$ is realized, then $\mathbb{A}$ has an extension $\mathbb{A}'$, obtained by adding only listener nodes, in which every cut is solved.*

Proof: 1. From Lemma 5.4.

2. Since $\mathbb{A}$ is realized, for each reception node $n \in \mathsf{nodes}(\mathbb{A})$, there is an adversary web G_n deriving $\mathsf{msg}(n)$ from preceding transmission nodes. Build $\mathbb{A}'$ by adding—for each message t used as a key on an encryption or decryption strand in G_n—a listener node ℓ for t, where $\ell \prec_{\mathbb{A}'} n$ and (for all m) $m \prec_{\mathbb{A}'} n$ implies $m \prec_{\mathbb{A}'} \ell$. By construction, all of these listeners are derivable, since the adversary has in fact derived them in G_n.

In $\mathbb{A}'$, let $n_1 \in \mathsf{Cut}(c, S, \mathbb{A}')$ be minimal in the cut, and let $\mathcal{B}$ be the bundle combining the regular nodes of $\mathbb{A}'$ with the adversary nodes of the webs G_n. Since $c \dagger^S \mathsf{msg}(n_1)$, there are $\preceq_{\mathcal{B}}$-minimal nodes m_1 such that $c \dagger^S \mathsf{msg}(m_1)$, and $m_1 \preceq_{\mathcal{B}} n_1$ is a transmission node. If $m_1 = n_1$, n_1 is solved in the cut. Otherwise, since n_1 is minimal in the cut, m_1 is an adversary node. Since S is a set of encryptions, m_1 lies on an encryption or decryption strand. By the construction of $\mathbb{A}'$, there is a listener below n_1 for the encryption or decryption key used on this adversary strand. □

When $Q = \mathsf{Cut}(c, S, \mathbb{A})$ and $H = [\phi_H, \alpha_H] : \mathbb{A} \hookrightarrow \mathbb{B}$, we say that $H(Q) = \mathsf{Cut}(\alpha_H(c), \alpha_H(S), \mathbb{B})$. H *solves* the test n_1, Q if $\phi_H(n_1)$ is solved in $H(Q)$. H *destroys* the test n_1, Q if $H(Q)$ if $\phi_H(n_1) \notin H(Q)$. If H destroys a test, then it solves it.

If $H : \mathbb{A} \hookrightarrow \mathbb{B}_H$ and H solves test n_1, Q, then by Def. 5.3, every H solving n_1, Q is of at least one of three kinds, namely a solution:

by destruction if $\phi_H(n_1) \notin H(Q)$;

by transmission if there exists $m_1 \preceq_{\mathbb{B}_H} \phi_H(n_1)$ such that m_1 is a transmission node that is minimal in $H(Q)$; or

by listener if there exists $m_1 = \mathsf{Lsn}[K]$ such that (a) $m_1 \preceq_{\mathbb{B}_H} \phi_H(n_1)$, and (b) either $\alpha_H(c) = \{|t|\}_K$ or else $\{|t|\}_{K^{-1}} \in \alpha_H(S)$.

6. Cohorts of Solutions

A *solution cover* is a set covering all essentially different ways to solve or destroy a test (modulo $\leq_n$). A *cohort* is a one where distinct members are $\leq_n$-incomparable:

Definition 6.1 *Let $Q = \mathsf{Cut}(c, S, \mathbb{A})$; let n_1 be an unsolved minimal node in Q; and let $\mathcal{H}$ be a set of homomorphisms $H : \mathbb{A} \hookrightarrow \mathbb{B}_H$, where each $\mathbb{B}_H$ is a skeleton. $\mathcal{H}$ is a* solution cover *for n_1 and $Q = \mathsf{Cut}(c, S, \mathbb{A})$ iff*

1. *For each $H \in \mathcal{H}$, $\phi_H(n_1)$ is solved in $H(Q)$; and*
2. *If $J\colon \mathbb{A} \rightarrow \mathbb{C}$, where $\phi_J(n_1)$ is solved in $J(Q)$, then for some $H \in \mathcal{H}$, $H \leq_n J$.*

$\mathcal{H}$ is a cohort *for n_1 and $Q = \mathsf{Cut}(c, S, \mathbb{A})$ iff it is a solution cover such that*

3. *If $H_1, H_2 \in \mathcal{H}$ and $H_1 \leq_n H_2$, then $H_2 = H_1$.*

If no J solves n_1, Q, then $\emptyset$ is a cohort for n_1, Q (and conversely).

In Clause 2 we require that $H \leq_n J$, rather than merely that $J = K \circ H$ with a possibly non-node-injective K. This requires some solution covers to contain more members. Consider, for instance, $\mathbb{B}'_{21}$, which differs from $\mathbb{B}_{21}$ in Fig. 5 by omitting the ordering relations $\prec$ between the top two nodes and the bottom two nodes. That is, $\mathbb{B}'_{21}$ contains an initiator strand and a responder strand, but with the minimal partial order. Thus, for the lower left node n_1, the test $n_1, Q = \mathsf{Cut}(\{\!|s|\!\}_k, \emptyset, \mathbb{B}'_{21})$ is unsolved.

A solution cover for n_1, Q consists of three homomorphisms: (i) a solution by listener that adds a listener strand $\mathsf{Lsn}[k]$; (ii) a solution by transmission that adds another responder strand with transmission node $m_1 \prec n_1$; and (iii) a solution by transmission that does not add any nodes, but adds a pair to the ordering relation, namely, restoring the precedence relation between the existing responder strand transmission and the reception node n_1. Although (iii) is not a node-injective extension of (ii), it does factor through (ii). Thus, without node-injectivity in Clause 2, the two homomorphisms (i,ii) would suffice.

Let $[\cdot]$ choose a canonical representative from each isomorphism class of homomorphisms; i.e. (i) if $J \leq_n H \leq_n J$, then $[H] = [J]$, and (ii) $[H] \leq_n H \leq_n [H]$. We write $\mu(S)$ to compose $[\cdot]$ with $\min$ (Def. 3.10); i.e. define $\mu(S) = \{[G]\colon G \in \min(S)\}$.

Lemma 6.2 *Let $Q = \mathsf{Cut}(c, S, \mathbb{A})$; let n_1 be an unsolved minimal node in Q.*

1. *$\{H\colon H$ solves $n_1, Q\}$ is a solution cover.*
2. *If $G \in \mathcal{H}$ and $\mathcal{H}$ is a solution cover for n_1, Q then so is $\mathcal{H} \downarrow G$, where*

$$\mathcal{H} \downarrow G = (\mathcal{H} \setminus \{K\colon G \leq_n K\}) \cup \{G\}.$$

3. *If $\mathcal{H}$ is a solution cover for n_1, Q then $\mu(\mathcal{H})$ is a cohort for n_1, Q.*

Proof: 1. Immediate from Def. 6.1.

2. Clause 1 holds because $\mathcal{H} \downarrow G \subseteq \mathcal{H}$. Clause 2 holds because if J is any solution, then $K \leq_n J$ for some $K \in \mathcal{H}$; but if $K \notin \mathcal{H} \downarrow G$, then $G \leq_n K \leq_n J$.

3. $\mu(\mathcal{H})$ is a solution cover by the preceding clause, since $\mu(\mathcal{H}) = \bigcap_{G \in \mathcal{H}} \mathcal{H} \downarrow G$. It is a cohort since it contains only canonical, $\leq_n$-minimal values. □

Definition 6.3 *When $Q = \mathsf{Cut}(c, S, \mathbb{A})$ and n_1 is an unsolved minimal node in Q, then define* $\mathsf{cohort}(n_1, Q) = \mu\{K\colon K$ *solves* $n_1, Q\}$.

Lemma 6.4 *If $H \in \mathsf{cohort}(n_1, Q)$ and H is a solution:*

by destruction, then *ϕ_H is surjective, $\preceq_{\mathbb{B}_H} = \phi(\preceq_{\mathbb{A}})$, $\mathsf{non}_{\mathbb{B}_H} = \alpha_H(\mathsf{non}_{\mathbb{A}})$, and $\mathsf{unique}_{\mathbb{B}_H} = \alpha_H(\mathsf{unique}_{\mathbb{A}})$.*
Moreover, α_H is $\leq_s$-minimal among β such that $\beta(c) \odot^{\beta(S)} \beta(\mathsf{msg}(n_1))$.

by transmission, then *there is exactly one transmission node $m_1 \preceq_{\mathbb{B}_H} \phi_H(n_1)$ that is minimal in $H(Q)$. Moreover:*

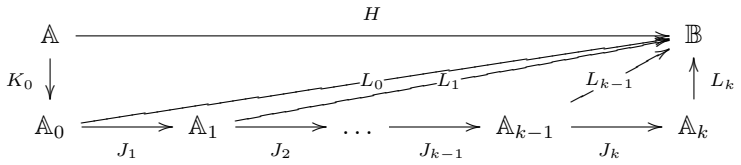


Figure 15. Cohort members J_i and nodewise-injective L_i

1. $\mathsf{nodes}(\mathbb{B}_H) = \mathsf{ran}(\phi_H) \cup \{m_0 \colon m_0 \Rightarrow^* m_1\}$*;*
2. $\preceq_{\mathbb{B}_H}$ *is the least ordering that extends* $\phi_H(\preceq_{\mathbb{A}})$*, satisfies Def. 3.1, Cl. 4b, and in which* $m_1 \preceq_{\mathbb{B}_H} \phi_H(n_1)$*;*
3. $\mathsf{non}_{\mathbb{B}_H} = \alpha_H(\mathsf{non}_{\mathbb{A}})$ *and* $\mathsf{unique}_{\mathbb{B}_H} = \alpha_H(\mathsf{unique}_{\mathbb{A}})$*;*[5]
4. *Letting* $m_1 = \gamma(\rho \downarrow j)$*,* γ *is* $\leq_s$*-minimal among* β *such that, for all* $k < j$*,* $\beta(c) \odot^{\beta(S)} \beta(\mathsf{msg}(\rho \downarrow k))$ *and also* $\alpha_H(c) \dagger^{\alpha_H(S)} \mathsf{msg}(\beta(\rho \downarrow j))$*.*

by listener, then $\mathsf{nodes}(\mathbb{B}_H) = \mathsf{ran}(\phi_H) \cup \{\mathsf{Lsn}[t]\}$ *for some* t *and* t_0 *such that either* $c = \{\!|t_0|\!\}_t$ *or else* $\{\!|t_0|\!\}_{t^{-1}} \in S$*.* $\preceq_{\mathbb{B}_H}$ *is the least ordering that extends* $\phi_H(\preceq_{\mathbb{A}})$*, satisfies Def. 3.1, Cl. 4b, and in which* $m_1 \preceq_{\mathbb{B}_H} \phi_H(n_1)$*. Moreover,* $\alpha_H = \mathsf{Id}$*,* $\mathsf{non}_{\mathbb{B}_H} = \mathsf{non}_{\mathbb{A}}$*, and* $\mathsf{unique}_{\mathbb{B}_H} = \mathsf{unique}_{\mathbb{A}}$*.*

Proof: In each case, if the stated condition is not true, then we can reduce H in the $\leq_n$ ordering by making it true. □

The comment about $\mathbb{B}'_{21}$ after Def. 6.1 points out that there are two possibilities covered in the **by transmission** clause here: m_1 may be a newly added node, or it may be a node already in $\mathbb{A}$, for which the ordering relation $m_1 \prec n_1$ has been newly added. This may also apply in the **by listener** clause.

6.1. Completeness of the Search

Solving tests meets a *progress* or *completeness* condition. Namely—modulo omission of listener strands—by solving tests we can reach all of the shapes. We allow an initial step in which some strands of $\mathbb{A}$ are identified; cf. Fig. 15.

Theorem 6.5 (Search Completeness) *Suppose* $H \colon \mathbb{A} \to \mathbb{B}$*, and every cut in* $\mathbb{B}$ *is solved. Let* $\mathcal{K} = \{K \colon \exists L \,.\, H = L \circ K \wedge \phi_K$ *is surjective* $\wedge\ \phi_L$ *is injective*$\}$*.*

$\mathcal{K} \neq \emptyset$*, and there is a universal* $K_0 \colon \mathbb{A} \to \mathbb{A}_0 \in \mathcal{K}$*.*

Let L_0 *be the node-injective* $\mathbb{A}_0 \rightarrowtail \mathbb{B}$ *with* $H = L_0 \circ K_0$*.*

There exists $k \geq 0$*, and there exist (a) tests* $\langle n_i, Q_i \rangle_{0<i\leq k}$*, (b) node-injective solutions* $\langle J_i \colon \mathbb{A}_{i-1} \rightarrowtail \mathbb{A}_i \rangle_{0<i\leq k}$*, and (c) node-injective* $\langle L_i \colon \mathbb{A}_i \rightarrowtail \mathbb{B} \rangle_{0<i\leq k}$*, such that:*

1. *for each* i *with* $0 < i \leq k$*,* $J_i \in \mathsf{cohort}(n_i, Q_i)$ *and* $L_i \circ J_i = L_{i-1}$*; and*
2. *every cut in* $\mathbb{A}_k$ *is solved.*

[5] When $m_1 = \gamma(\rho \downarrow j)$, and ρ has origination assumptions (see p. 25), we in fact have $\mathsf{non}_{\mathbb{B}_H} = \alpha_H(\mathsf{non}_{\mathbb{A}}) \cup \gamma(\mathsf{role\_non}(\rho \downarrow j))$ and $\mathsf{unique}_{\mathbb{B}_H} = \alpha_H(\mathsf{unique}_{\mathbb{A}}) \cup \gamma(\mathsf{role\_non}(\rho \downarrow j))$.

Proof: Let $K = [\phi_H, \alpha_H] \colon \mathbb{A} \rightarrow (\mathbb{B} \restriction \phi_H(\mathsf{nodes}(\mathbb{A})))$ be H with its target restricted to the image of $\mathbb{A}$. Then K and the embedding $\mathbb{B} \restriction (\phi_H(\mathsf{nodes}(\mathbb{A}))) \hookrightarrow \mathbb{B}$ are respectively node-surjective and node-injective. Thus, $\mathcal{K} \neq \emptyset$.

For each pair n_1, n_2 of nodes of $\mathbb{A}$ such that $\phi_H(n_1) = \phi_H(n_2)$, apply Lemma 3.14, Clause 3; so there is a universal K_0. L_0 is injective, since we have already identified all the pairs that ϕ_H does.

Now suppose that $\mathbb{A}_i, L_i$ has been defined. If all cuts in $\mathbb{A}_i$ are solved, we are done.

Otherwise, let (n_i, Q_i) with $n_i \in \mathbb{A}_i$ and $Q_i = \mathsf{Cut}(c_i, S_i, \mathbb{A}_i)$ be unsolved. Since $L_i(n_i, Q_i)$ is solved, $J_{i+1} \leq_n L_i$ for some $J_{i+1} \in \mathsf{cohort}(n_i, Q_i)$. Any such J_{i+1} must be injective, since L_i is. Define L_{i+1} to be the node-injective homomorphism such that $L_i = L_{i+1} \circ J_{i+1}$, which exists by $J_{i+1} \leq_n L_i$.

This process must terminate with a $\mathbb{A}_k$ in which all tests are solved, because, for each i, $(J_i \circ \ldots \circ J_1 \circ K_0) \leq_n H$, and there are only finitely many non-isomorphic values $\leq_n H$ by Lemma 3.11, Clause 4. □

6.2. Cohorts and Characterizations

CPSA incrementally computes a (node-injective) characterization $\mathcal{C}$ for some starting point $\mathbb{A}$. A cohort is a single information-increasing step in this process.

Although, as Thm. 5.5 suggests, CPSA adds listener nodes to solve tests, as reception nodes they do not increase the information available to the adversary. In a shape $H \colon \mathbb{A} \rightarrow \mathbb{B}$, for each listener node $n \in \mathsf{nodes}(\mathbb{B})$, there is a listener node $m \in \mathsf{nodes}(\mathbb{A})$ such that $n = \phi_H(m)$. Otherwise $\mathbb{B}$ would not be minimal; n could be omitted.

CPSA adds listener nodes for keys K to indicate possible solutions in which K is disclosed. It may later add other regular strands showing how K became disclosed. Thus, CPSA's intermediate steps use listener nodes that it will discard when reporting a shape. We define *fringe* to trim out these listener nodes that will eventually be discarded.

Definition 6.6 *When* $H \colon \mathbb{A} \rightarrow \mathbb{B}$, *define* $\mathsf{trim}(H) = [\phi_H, \alpha_h] \colon \mathbb{A} \rightarrow \mathbb{B}'$ *where*

$$\mathcal{N} = \{n \in \mathsf{nodes}(\mathbb{B}) \colon \exists m \in \mathsf{nodes}(\mathbb{A}) \,.\, n = \phi_H(m) \vee n \textit{ is not a listener node}\}$$

$$\mathbb{B}' = \langle \mathcal{N},\ (\preceq_{\mathbb{B}} \restriction \mathcal{N} \times \mathcal{N}),\ \mathsf{non}_{\mathbb{B}},\ \mathsf{unique}_{\mathbb{B}} \rangle.$$

Define $\mathsf{trim}(\mathcal{F}) = \{\mathsf{trim}(H) \colon H \in \mathcal{F}\}$. *A set* $\mathcal{F}$ *of homomorphisms is a* fringe for $\mathcal{C}$ *iff every* $K \in \mathcal{C}$ *factors through* $\mathsf{trim}(\mathcal{F})$.

Define $\mathsf{Targets}(\mathcal{F}) = \{\mathbb{B} \colon (F \colon \mathbb{A} \rightarrow \mathbb{B}) \in \mathcal{F}\}$.

This theorem follows from the definitions via Lemma 3.11:

Theorem 6.7 *Let* $\mathcal{C}$ *be a characterization for* $\mathbb{A}$. *The singleton* $\{\mathsf{Id}_{\mathbb{A}}\}$ *is a fringe for* $\mathcal{C}$.

Let $\mathcal{F}$ *be a fringe for* $\mathcal{C}$.

1. *If* $F \colon \mathbb{A} \rightarrow \mathbb{B}$ *in* $\mathcal{F}$ *where* n_1, Q *is unsolved, then* $\mathcal{F}'$ *is also a fringe for* $\mathcal{C}$, *where*

$$\mathcal{F}' = (\mathcal{F} \setminus \{F\}) \cup \{H \circ F \colon H \in \mathsf{cohort}(n_1, Q)\}.$$

2. *If every* $\mathbb{B} \in \mathsf{Targets}(\mathcal{F})$ *is realized, then* $\min(\mathcal{C}) \sim \min(\mathsf{trim}(\mathcal{F}))$.

In Cl. 1, when $\mathsf{cohort}(n_1, Q)$ is empty, $\mathbb{B}$ is dead, and we discard $F \colon \mathbb{A} \rightarrow \mathbb{B}$.

Thm. 6.7 provides motivation for CPSA's algorithm. CPSA uses breadth-first search starting with an initial skeleton $\mathbb{A}$. CPSA maintains the fringe $\mathcal{F}$ as a union of two sets, $\mathcal{U} \cup \mathcal{S}$. $\mathcal{U}$ contains the as-yet unexplored part. $\mathcal{S}$ contains the current estimate of the set of shapes. $\mathcal{S}$ contains those homomorphisms $F\colon \mathbb{A} \to \mathbb{B}$ such that $\mathbb{B}$ is realized, and no $G <_n F$ with realized target has yet been seen. Initially, $\mathcal{U} = \{\mathsf{Id}_{\mathbb{A}}\}$, and $\mathcal{S} = \emptyset$.

CPSA halts if $\mathcal{U}$ is empty, returning $\mathcal{S}$.

Otherwise, it selects one $F\colon \mathbb{A} \to \mathbb{B}$ from $\mathcal{U}$. If $\mathbb{B}$ is realized, it compares $\mathsf{trim}(F)$ to the current members of $\mathcal{S}$. If for some $G \in \mathcal{S}$, $G <_n \mathsf{trim}(F)$, F is discarded, and otherwise we add $\mathsf{trim}(F)$ to $\mathcal{S}$. If $\mathsf{trim}(F) <_n G$ for some $G \in \mathcal{S}$, we discard them.

If $\mathbb{B}$ is not realized, CPSA finds an unsolved test $n_1, Q = \mathsf{Cut}(c, S, \mathbb{B})$. CPSA computes $\mathsf{cohort}(n_1, Q)$, and then replaces F by $\{H \circ F\colon H \in \mathsf{cohort}(n_1, Q)\}$.

How does CPSA compute $\mathsf{cohort}(n_1, Q)$? For solutions by **destruction** and by **listener**, it follows the prescriptions given in Lemma 6.4. In finding the solutions by **transmission**, there are two stages. In the first, CPSA considers every $\rho \in \Pi$ and each transmission node $\rho \downarrow j$. For each path p within $\mathsf{msg}(\rho \downarrow j)$, if p traverses no key edge, CPSA attempts to unify c with $p(\mathsf{msg}(\rho \downarrow j))$. If this unification succeeds with some γ_0, CPSA attempts to extend γ_0 to satisfy the condition of Lemma 6.4, Cl. 4. If S is finite, then since there are at most finitely many encryptions along the path to any occurrence in an earlier node m_0, there can be only finitely many successful outcomes. CPSA collects all successful results, as ρ, j, p vary.

This first step could leave out solutions by transmission in which the critical value c is inserted into $\mathsf{msg}(\rho \downarrow j)$ as an ingredient in a larger message t. Suppose, for some path p, $p(\mathsf{msg}(\rho \downarrow j))$ is an indeterminate x. Choose γ_0 with $\gamma_0(x) = t$ for any t where $c \sqsubseteq t$. Consider $m_1 = \gamma_0(\rho \downarrow j)$. Suppose that in $\mathsf{msg}(m_1)$, p traverses neither a key edge nor any $e \in S$, but whenever $m_0 \Rightarrow^+ m_1$, every path p_0 in $\mathsf{msg}(m_0)$ that reaches c traverses either a key edge or a member of S. Then m_1 provides a solution to S, even though the choice of $\gamma_0(x) = t$ was not motivated syntactically by unification in $\mathsf{msg}(\rho \downarrow j)$.

However, by the Indeterminate Acquisition Principle (Section 2.5, p. 16), there is at least one reception node $\rho \downarrow i$ where $i < j$ and $x \sqsubseteq \mathsf{msg}(\rho \downarrow i)$. Thus, there is at least one path p_0 such that p_0 does not traverse a key edge, and $p_0(\mathsf{msg}(\rho \downarrow i)) = x$, or equivalently $p_0(\mathsf{msg}(\gamma_0(\rho \downarrow i))) = t$. Along every such path in $\mathsf{msg}(\gamma_0(\rho \downarrow i))$, we must unify some encryption with a member of S. Since there are only a finite number of ways to succeed, only finitely many t can be successful substitutions for x.[6]

Theorem 6.8 *Let S be finite, and let $\mathcal{H}$ be a cohort for $n_1, Q = \mathsf{Cut}(c, S, \mathbb{A})$. There are only finitely many $H \in \mathcal{H}$.*

Proof: By the finiteness of S, there are only finitely many solutions by **listener**. Moreover, consider a solution by **destruction** H. Since $\alpha_H(c) \odot^{\alpha_H(S)} \alpha_H(\mathsf{msg}(n_1))$, for each path p to c in $\mathsf{msg}(n_1)$, there is some encryption e that p traverses, such that α_H unifies e with some member of S. By the finiteness of S and of $\mathsf{msg}(n_1)$, and the m.g.u. property, there are only finitely many most general ways to do so.

If H is a solution by **transmission**, then it meets the conditions in the discussion above, in each case belonging to a finite set of possibilities. □

[6] Key ideas in the cohort computation are due to John Ramsdell and Paul Rowe.

The CPSA algorithm is thus finitely branching; by Thm. 6.7 it never discards any possibilities, and by Thm. 6.5 it reaches each shape after finitely many steps.

Conclusion. We have presented here a theory explaining CPSA, the Cryptographic Protocol Shape Analyzer. We will write more specifically about the design and implementation of CPSA elsewhere.

We believe that the Authentication Test Principle is central to cryptographic protocols. Indeed, already in our first paper about the tests, we pointed out that they provide very strong heuristics to guide cryptographic protocol design [12]. We have also illustrated a systematic protocol design process, which led to a protocol achieving goals akin to the Secure Electronic Transaction protocol, organized by reference to the tests [9]. Moreover, we have recently used the tests to justify a criterion for when combining a protocol with new behaviors preserves all security goals met by the original protocol [10]. We hope to develop the theory we have just described to provide a systematic set of protocol transformations that preserve security goals. This appears likely to provide rigorous techniques to replace the heuristics that protocol designers currently use.

Acknowledgments. I am enormously grateful to F. Javier Thayer, who through a decade and more has been an integral collaborator. Shaddin Dughmi (a.k.a. Doghmi) and John D. Ramsdell contributed so much to the theory and to making it work. Dughmi wrote the first version of CPSA. Ramsdell is author of the current version.

Jonathan Herzog contributed to the original strand space ideas. Paul Rowe suggested key ideas and puzzles. Moses Liskov and Leonard Monk helped clarify this chapter.

References

[1] Boris Balacheff, Liqun Chen, Siani Pearson, David Plaquin, and Graeme Proudler. *Trusted Computing Platforms: TCPA Technology in Context*. Prentice Hall PTR, Upper Saddle River, NJ, 2003.

[2] Bruno Blanchet. *Vérification automatique de protocoles cryptographiques: modèle formel et modèle calculatoire. Automatic verification of security protocols: formal model and computational model*. Mémoire d'habilitation à diriger des recherches, Université Paris-Dauphine, November 2008.

[3] Edmund Clarke, Somesh Jha, and Will Marrero. Using state space exploration and a natural deduction style message derivation engine to verify security protocols. In *Proceedings, IFIP Working Conference on Programming Concepts and Methods (*PROCOMET*)*, 1998.

[4] Cas J.F. Cremers. Unbounded verification, falsification, and characterization of security protocols by pattern refinement. In *ACM Conference on Computer and Communications Security (CCS)*, pages 119–128, New York, NY, USA, 2008. ACM.

[5] C.J.F. Cremers. *Scyther - Semantics and Verification of Security Protocols*. Ph.D. dissertation, Eindhoven University of Technology, 2006.

[6] Shaddin F. Doghmi, Joshua D. Guttman, and F. Javier Thayer. Completeness of the authentication tests. In J. Biskup and J. Lopez, editors, *European Symposium on Research in Computer Security (ESORICS)*, number 4734 in LNCS, pages 106–121. Springer-Verlag, September 2007.

[7] Shaddin F. Doghmi, Joshua D. Guttman, and F. Javier Thayer. Searching for shapes in cryptographic protocols. In *Tools and Algorithms for Construction and Analysis of Systems (TACAS)*, number 4424 in LNCS, pages 523–538. Springer, March 2007.

[8] Shaddin F. Doghmi, Joshua D. Guttman, and F. Javier Thayer. Skeletons, homomorphisms, and shapes: Characterizing protocol executions. In M. Mislove, editor, *Proceedings, Mathematical Foundations of Program Semantics*, April 2007.

[9] Joshua D. Guttman. Authentication tests and disjoint encryption: a design method for security protocols. *Journal of Computer Security*, 12(3/4):409–433, 2004. Extended version of "Security Protocol Design via Authentication Tests," CSFW 2002.

[10] Joshua D. Guttman. Cryptographic protocol composition via the authentication tests. In Luca de Alfaro, editor, *Foundations of Software Science and Computation Structures (FOSSACS)*, number 5504 in LNCS, pages 303–317. Springer, March 2009.

[11] Joshua D. Guttman. Fair exchange in strand spaces. In M. Boreale and S. Kremer, editors, *SecCo: 7th International Workshop on Security Issues in Concurrency*, EPTCS. Electronic Proceedings in Theoretical Computer Science, Sep 2009.

[12] Joshua D. Guttman and F. Javier Thayer. Authentication tests and the structure of bundles. *Theoretical Computer Science*, 283(2):333–380, June 2002. Conference version appeared in *IEEE Symposium on Security and Privacy*, May 2000.

[13] Jonathan K. Millen and Vitaly Shmatikov. Constraint solving for bounded-process cryptographic protocol analysis. In *8th ACM Conference on Computer and Communications Security (CCS '01)*, pages 166–175. ACM, 2001.

[14] 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.

[15] 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.

[16] Dag Prawitz. *Natural Deduction: A Proof-Theoretic Study*. Almqvist and Wiksel, Stockholm, 1965.

[17] John D. Ramsdell and Joshua D. Guttman. CPSA: A cryptographic protocol shapes analyzer. In *Hackage*. The MITRE Corporation, 2009. `http://hackage.haskell.org/package/cpsa`.

[18] John D. Ramsdell, Joshua D. Guttman, and Paul D. Rowe. *The CPSA Specification: A Reduction System for Searching for Shapes in Cryptographic Protocols*. The MITRE Corporation, 2009. In `http://hackage.haskell.org/package/cpsa` source distribution, `doc` directory.

[19] Dawn Xiaodong Song. Athena: a new efficient automated checker for security protocol analysis. In *Proceedings of the 12th IEEE Computer Security Foundations Workshop*. IEEE Computer Society Press, June 1999.

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