Strand Spaces

Ryan Culpepper

Outline

Strand spaces

- Concepts and intuitions
- Modeling protocols
- Specifying and verifying properties
- Applications

Concepts and intuitions

Traces

Protocols are often modeled with *traces*:

Hamlet	:	The air bites shrewdly; it is very cold.
Horatio	:	It is a nipping and an eager air.
Hamlet	:	What hour now?
Horatio	:	I think it lacks of twelve.
Marcellus	:	No, it is struck.

Strands

- A *strand* is a perspective on a protocol interaction.
- Hamlet's role:

say	:	The air bites shrewdly; it is very cold.
cue	:	It is a nipping and eager air.
say	:	What hour now?

- Marcellus's role:

cue	:	I think it lacks of twelve.
say	:	No, it is struck.

Bundles

- Strands combine to form *bundles*
- Bundles represent actual protocol interactions

Horatio		**Hamlet**		**Marcellus**
cue: The air bites shrewdly; it is very cold.	←	say: The air bites shrewdly; it is very cold.		
say: It is a nipping and an eager air.	→	cue: It is a nipping and an eager air.		
cue: What hour now?	←	say: What hour now?		
say: I think it lacks of twelve.	→		→	cue: I think it lacks of twelve.
				say: No, it is struck.

Strand space

A *strand space* is a set of strands of

- the initiator and responder roles
- the penetrator (attacker)

Protocols

Protocol

$\Downarrow$

Strand space specification

Protocols

Protocol property

$\Downarrow$

Mathematical proposition

about bundles over strand space

Protocols

Verification

$\Downarrow$

Proof of proposition

Formalism

Terms

There is a set $\mathcal{A}$ of terms.

- contains the set $\mathcal{T}$ of atomic terms
- contains the set $\mathcal{K}$ of cryptographic keys
- closed under concatenation
- closed under encryption/decryption
- free algebra

A signed term is a pair of a sign $\sigma \in \{+, -\}$ and a term t, written either $\langle \sigma, t \rangle$ or $+t$ or $-t$.
$(\pm\mathcal{A})^*$ is the set of finite sequences of signed terms.

Strand spaces

- A strand space Σ is a set of strands.
- Each strand has a trace:

$$tr : \Sigma \to (\pm\mathcal{A})^*$$

- Many strands may share the same trace.
- Many traces share the same shape.

Definitions

Let Σ be a strand space.

- A *node* is a pair $\langle s, i \rangle$ of a strand $s \in \Sigma$ and an index i where $1 \leq i \leq length(tr(s))$.
- $\mathcal{N}$ is the set of nodes.
- $term : \mathcal{N} \rightarrow$ Signed terms
- $\rightarrow$ is a relation on nodes where

$$n \rightarrow n' \text{ iff } term(n) = +t \text{ and } term(n') = -t$$

- $\Rightarrow$ is a relation on nodes where

$$\langle s, i \rangle \Rightarrow \langle s, i+1 \rangle$$

Definitions

Let $I \subseteq \mathcal{A}$ be a set of unsigned terms. Then $n \in \mathcal{N}$ is an *entry point* for I iff:

$$t \in I$$

$$term(n) = +t$$

$$\forall n' \Rightarrow^+ n : term(n') \notin I$$

An unsigned term t originates on $n \in \mathcal{N}$ if n is an entry point for the set of all terms containing t.

An unsigned term t is *uniquely originating* if it originates on a unique node.

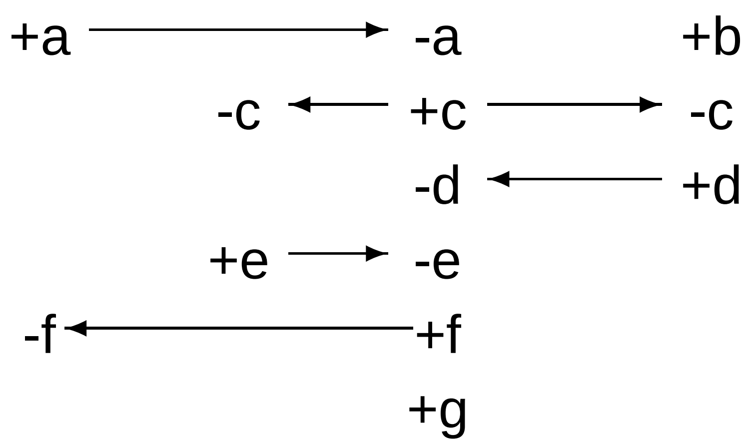
Bundles

A bundle $\mathcal{C}$ is a graph of nodes $\langle \mathcal{N}_\mathcal{C}, \rightarrow_\mathcal{C}, \Rightarrow_\mathcal{C} \rangle$.

- $\mathcal{C}$ is finite and acyclic
- $\mathcal{N}_\mathcal{C} \subseteq \mathcal{N}$
- $\rightarrow_\mathcal{C} \subseteq \rightarrow$
- $\Rightarrow_\mathcal{C} \subseteq \Rightarrow$
- A node with a negative term has a unique $\rightarrow$-edge coming into it
- If $n_2 \in \mathcal{N}_\mathcal{C}$ and $n_1 \Rightarrow n_2$, then $n_1 \Rightarrow_\mathcal{C} n_2$.

Bundles

Here is an example bundle:



Bundles

Causal precedence

- Edges generate partial order $\preceq$
- $n \preceq n'$ means n may influence terms of n'

Induction

- Every non-empty set of nodes has a non-empty $\preceq$-minimal subset
- “Who knew what when?”

Proof tools

Proofs involve arguments about:

- Entry points, origination, and unique origination
- Causality and $\preceq$-minimal nodes
- Case analysis on strand shapes

Modeling Protocols

Needham-Schroeder-Lowe

Needham-Schroeder protocol as fixed by Lowe:

1. $A \longrightarrow B : \{N_a, A\}_{K_B}$
2. $B \longrightarrow A : \{N_a, N_b, B\}_{K_A}$
3. $A \longrightarrow B : \{N_b\}_{K_B}$

Modeling the protocol

Protocols are modeled as strand spaces

An NSL strand space is the union of three kinds of strands:

- Initiator strands
- Responder strands
- Penetrator (attacker) strands

Initiator and responder strands are called “regular strands”, and their nodes are called “regular nodes.”

Initiator strands

$\bigcup\{\mathrm{Init}[A, B, N_a, N_b] \mid A, B \in \mathcal{T}_{\mathrm{names}}, N_a, N_b \in \mathcal{T} - \mathcal{T}_{\mathrm{names}}\}$

Each strand in $\mathrm{Init}[A, B, N_a, N_b]$ has the trace:

$$\begin{aligned}
&+\{N_a, A\}_{K_B} \\
&-\{N_a, N_b, B\}_{K_A} \\
&+\{N_b\}_{K_B}
\end{aligned}$$

Responder strands

$\bigcup\{\mathrm{Resp}[A, B, N_a, N_b] \mid A, B \in \mathcal{T}_{\mathrm{names}}, N_a, N_b \in \mathcal{T} - \mathcal{T}_{\mathrm{names}}\}$

Each strand in $\mathrm{Resp}[A, B, N_a, N_b]$ has the trace:

$$-\{N_a, A\}_{K_B}$$
$$+\{N_a, N_b, B\}_{K_A}$$
$$-\{N_b\}_{K_B}$$

Penetrator strands

Penetrators

- have initial information: compromised keys $\mathcal{K}_p$
- have many capabilities, and they can combine those capabilities in many ways.
- are patient; they can watch through many protocol interactions until they gather enough information.

Penetrator strands sound like they could be complex and arbitrarily long:

- “Our villain watches three protocol interactions, injects a message into a fourth, watches a fifth, initiates an interaction using data from the second, and ...”

Penetrator strands

Characterize penetrator *capabilities* rather than *attacks*.
Model a beaurocracy of penetrators!

- One class of strand per capability
- Many penetrator strands may be combined in a bundle
- Considering “all possible bundles” automatically creates “all possible penetrators”
- Reusable definition: “penetrator standard library”
  Reusable theorems about standard penetrators

Penetrator capabilities

Dolev-Yao attacker:

- $\mathbf{M}: \langle +t \rangle$, where $t \in \mathcal{T}$
- $\mathbf{F}: \langle -g \rangle$, where $g \in Terms$
- $\mathbf{T}: \langle -g, +g, +g \rangle$
- $\mathbf{C}: \langle -g, -h, +gh \rangle$
- $\mathbf{S}: \langle -gh, +g, +h \rangle$
- $\mathbf{E}: \langle -K_0, -h, +\{h\}_{K_0} \rangle$
- $\mathbf{D}: \langle -K_0, -\{h\}_{K_0}, +h \rangle$

Standard penetrators

Standard penetrators have standard limits

- If the penetrator doesn't start out with a key K, and that key never *originates* on a regular node, then K is not a subterm of any penetrator node's term.

  Suppose it does occur in some set of nodes. Take the $\preceq$-minimal base; those must all be penetrator nodes. Do case analysis of penetrator nodes.

Stating and Verifying Protocol Properties

Needham-Schroeder-Lowe properties

- Authentication of initiator to responder
- Authentication of responder to initiator
- Secrecy of nonces

Weak agreement

One form of authentication:

- Whenever B completes a run as responder using N_a, N_b with A as apparent initiator, there is a run of the protocol with A as initiator using N_a, N_b with B as apparent responder.

Weak agreement as proposition

Suppose the following:

- Σ is an NSL space, $\mathcal{C}$ is a bundle in Σ, and $s \in \mathrm{Resp}[A, B, N_a, N_b]$ is a complete responder strand in $\mathcal{C}$.
- $K_A^{-1} \notin \mathcal{K}_p$
- $N_a \neq N_b$ and N_b is uniquely originating in Σ.

Then:

- $\mathcal{C}$ contains a complete initiator's strand in $\mathrm{Init}[A, B, N_a, N_b]$.

Proving weak agreement

A few pages of math.

Secrecy as a proposition

Suppose the following:

- Σ is an NSL space, $\mathcal{C}$ is a bundle in Σ, and $s \in \mathrm{Resp}[A, B, N_a, N_b]$ is a responder strand in $\mathcal{C}$.
- $K_A^{-1} \notin \mathcal{K}_p$
- $K_B^{-1} \notin \mathcal{K}_p$
- $N_a \neq N_b$ and N_b is uniquely originating in Σ.

Then:

- For all nodes $n \in \mathcal{C}$, $term(n) \neq N_b$.

Proving secrecy

Another page or two of math.

Applications

CPPL

- Cryptographic Protocol Programming Language
- Based on strand space semantics
- Compiles domain-specific protocol language via O'Caml

Motivation

- Protocol design isn't "done."
- Different applications have different *agreement* and *commitment* goals.
- Bring implementation and analysis closer together.

Example

A data server based on the Needham-Schroeder (original) protocol:

$$A \longrightarrow B : \{N_a, A, D\}_{K_B}$$
$$B \longrightarrow A : \{N_a, SK\}_{K_A}$$
$$A \longrightarrow B : \{SK\}_{K_B}$$
$$B \longrightarrow A : \{\mathbf{data_is}, V\}_{SK}$$

Relies and guarantees

Idea of CPPL:

- Annotate message sends with *guarantees*
- Annotate message receives with *relies*

Protocol soundness:

- If P receives a message apparently from P' and relies on a formula ϕ, then P' previously sent the message with a formula ψ, where $\psi \Rightarrow \phi$.

NSQ Code

```
proc server (b:text, kb:key) _
  let chan = accept in
  (chan recv {na:nonce, a:text, d:text} kb _
    let sk:symkey = new in
    (send _ chan {na, sk, b} ka
      (chan recv {sk} kb _
        (send _ chan {Data_is v} sk
          return _))))
```

NSQ Code

%

```
proc server (b:text, kb:key) [owns(b,kb)]
  let chan = accept in
  (chan recv {na:nonce, a:text, d:text} kb [true]
    let sk:symkey = new in
    (send [owns(a,ka)] chan {na,sk,b} ka
      (chan recv {sk} kb [says_requests(a,a,b,d)]
        (send [will_pay(a,d); curr_val(d,na,v:text)]
              chan {Data_is v} sk
          return [supplied(a,na,d,v)]))))
```

Semantics

Semantics of CPPL maps processes to sets of strands. Verify resulting strand space, or translate further to other frameworks for verification.

Conclusion

References

- “Strand Spaces: Proving Security Protocols Correct”, Fábrega, Herzog, and Guttman. *Journal of Computer Security, 1999*.
- “Programming Cryptographic Protocols”, Guttman, Herzog, Ramsdell, and Sniffen. *Symposium on Trustworthy Global Computing, 2005*.
- http://www.mitre.org/tech/strands/

The End