Verified Cryptographic Protocol Analysis Vision and Status

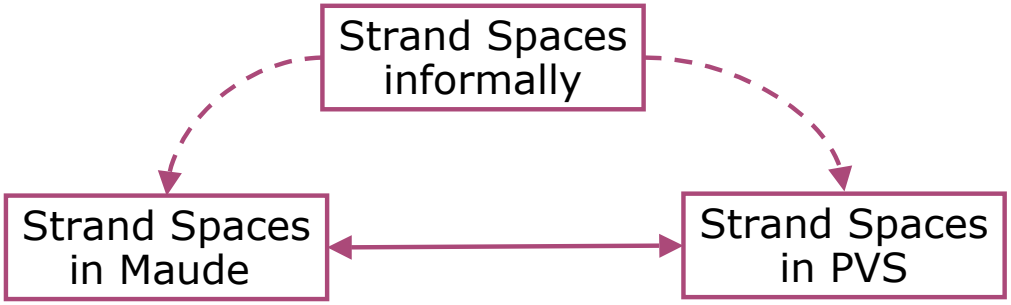
Carolyn Talcott
SRI International
HCSS April 2006

Plan

- Vision then and now
- About Strand Spaces
- Tool Interoperation
  - Capabilities
  - Subtleties
- Towards Formal Semantic foundation

Original Vision
(due to Sylvan Pinsky)

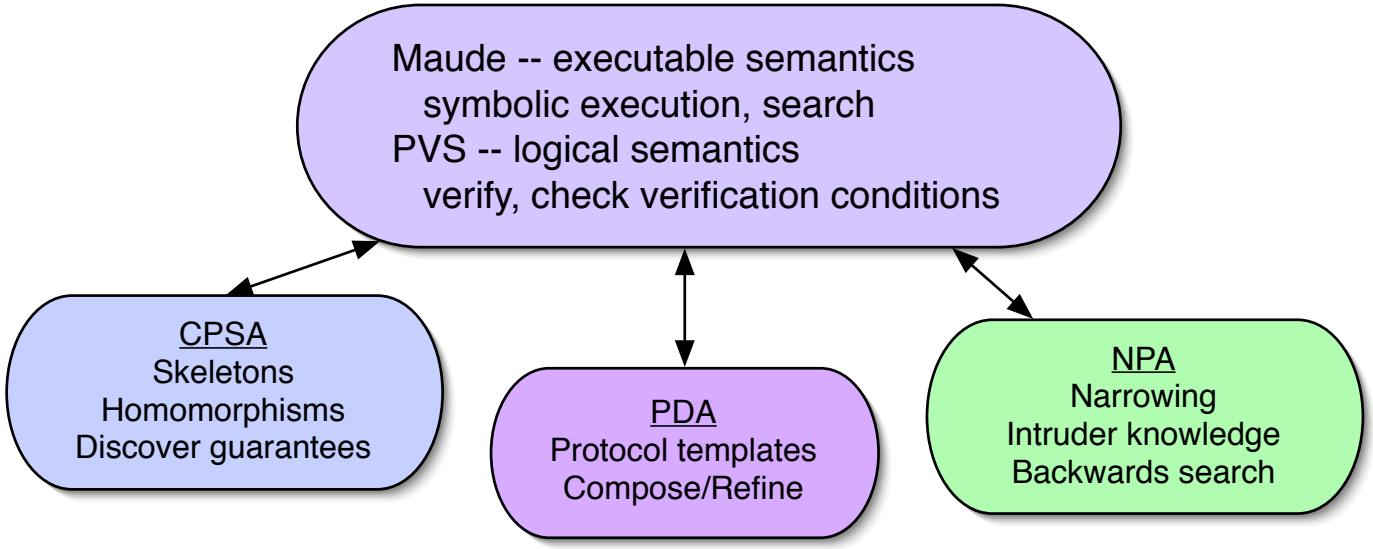
- A foundation for developing cryptographic protocol analysis algorithms
  - Specify and prototype in Maude
  - Verify in PVS



- Formal representations of strand spaces and strand space protocols in Maude and PVS
- Meaning preserving mappings between them

Emerging Vision
(from Protocol eXchange)

Formal framework for semantically sound interoperation of tools for design and analysis of cryptographic protocols (building on the strand space model)



Strand Spaces in a Nutshell

Strand Space Basics

A mathematical model for cryptographic protocols

- a strand is a sequence of events representing an execution of a legitimate participant or penetrator
- a strand space is a collection of strands equipped with a causal order relation on events
- elements of a strand are called nodes, the causal order on nodes gives rise to a graph
- bundles are acyclic subgraphs corresponding to possible runs

Strand Space Protocols

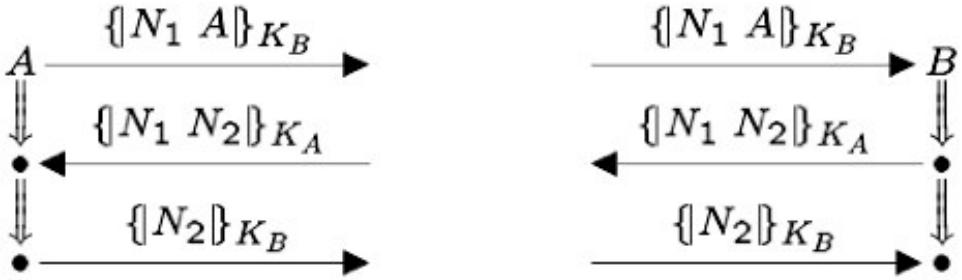
- A protocol is a set of roles---strand templates.
- Penetrator strands correspond to the Dolev-Yao model for attackers
- The strand space for a protocol has all instances of the protocol roles as well as all possible penetrator strands.

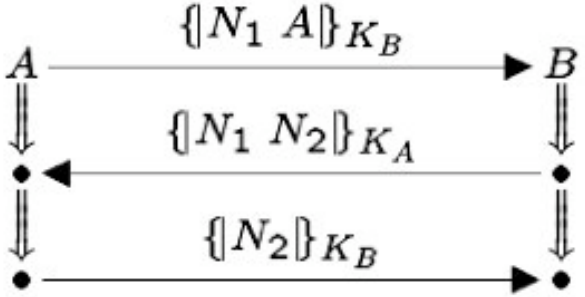
Needhan-Schroeder protocol

- Alice (A) would like to establish a shared secret with Bob (B)
  - A -> B : {| Na ^ A |} Kb
  - B -> A : {| Na ^ Nb |} Ka
  - A -> B : {| Nb |} Kb
- Furthermore, if Bob completes a run, he would like to be assured that Alice was the partner and that Na,Nb is a shared secret

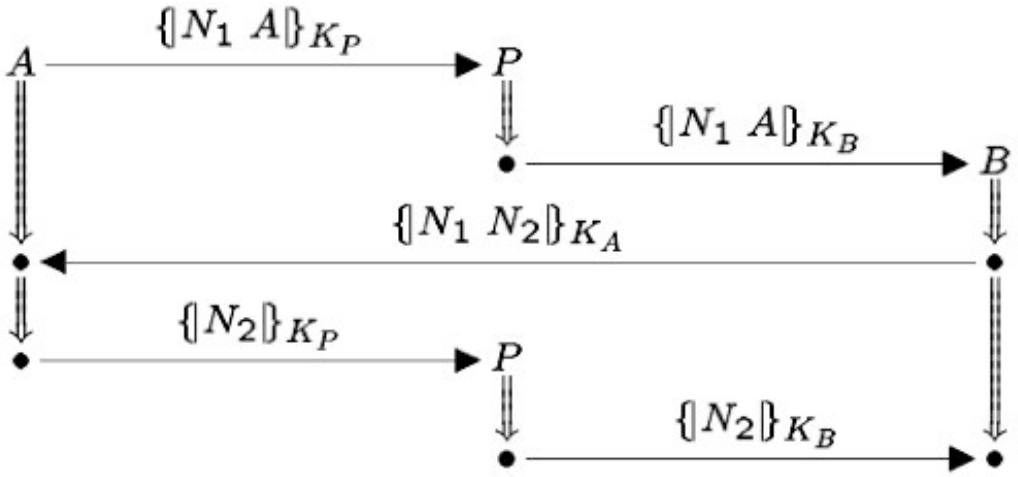
Needhan-Schroeder protocol

Role strands





Bundle for desired run



Bundle of run with penetrator

Security Goals and Tests

- Secrecy --- what the penetrator can see / not see
- Safe keys --- penetrator can not obtain
  - non-originating or only sent protected by safe key
- Authentication
  - what other participants must have done
- Outgoing test --- if A sends a fresh nonce protected by safe key, and receives it back in a new form then a regular strand must have transformed it
- Incoming test --- if A sends a fresh nonce in the clear and received it back protected by a safe key the a regular strand must have transformed it

Tools

Cryptographic Protocol Shape Analyzer (CPSA MITRE)

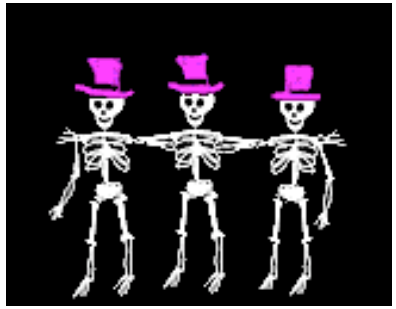
Protocol Derivation Assistant (PDA Kestrel)

NPA-Maude (NRL Protocol Analyzer in Maude
NRL, UIUC, U. Valencia)

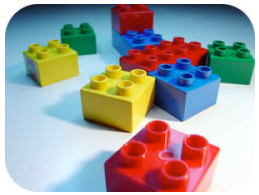
.....

CPSA

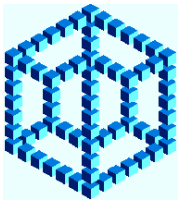
- Reasons about skeletons (regular part of bundle)
- Generate possible shapes by solving authentication tests, what else must have happened.
- Infers safe keys



PDA



- Abstract building blocks
  - basic protocol elements and associated properties
  - schema express constraints on parameters
- Derive more complex protocols and their properties by composition and refinement



- Underlying protocol composition logic

NPA-Maude

- Start with potential bad state and show it is unreachable
  - reason about what the intruder has or has not learned
  - run execution rules backwards
  - use grammars to prune search space

Tool Cooperation

- At some level they are working with the same things
  - analysis of cryptographic protocols
  - strand space-like semantics

- Taking different approaches
  - CPSA: discovers guarantees given protocol and skeleton (run of some role)
  - PDA: properties hold by construction, but you have to figure out how to derive them
  - NPA: user postulate bad states for analysis

Tool Communication

- Shared S-expression grammar
  - captures basics concepts abstractly
  - sublanguage understood by each tool
  - in progress!
- Concepts
  - Term, Action / Signed Message
  - Program / Signed Message List, Agent/Role
  - Protocol
  - Process, Skeleton
  - Annotations

Tool Interoperation

The devil is in the details

- Choice of crypto primitives
  - CPSA currently fixed set of crypto
  - NPA allows user to introduce and define equationally
  - PDA allows arbitrary functions to be introduced, and axiomatized in the PDA logic
- Protocol structure
  - CPSA/NPA linear sequences of signed terms
  - PDA sequential/parallel composition of actions with intended run made explicit

Tool Interoperation II

- Role of intruder
  - CPSA / PDA -- implicit
  - NPA -- explicit

- Matching vs deconstuctors
  - {A,N}K := M, vs
  - A := 1st(decrypt(M,K)), N := 2nd(decrypt(M,K))
- CPSA -- implicit matching
- PDA -- explicit matching
- NPA -- uses deconstructors to analyze, but implicitly

Tool Interoperation III

- Expressing freshness
  - CSPA uses strand annotation
  - PDA has an explicit action
  - NPA-maude uses a `new' abstraction
- Confidentiality
  - CPSA: non-origination annotation, safe induction
  - PDA: specified in logic as assumptions
  - NPA: specified in facts about I (intruder knowledge)

Semantics
Current State

Strand Spaces in PVS

- 1999 Strand spaces paper
  - Mostly a direct formalization of the paper
  - Some reformulation of lemmas for automated use
- Shapes of Strands paper (2004-5)
  - Closely follows paper
  - Small problems unearthed
- Need to develop strategies for properties of specific protocols (tedium elimination)

Basic Semantic Model

- Generalize bundle to event partial order
  - Event: instance of an action by a player
  - Action: send, receive, internal
  - Principals may have multiple, related names
  - Receive has (unique) preceding send

Executable semantics in Maude (exists in 3 flavors)

- Protocol language
  - Terms [parametric in crypto algebra]
  - Actions / Code
  - Roles/agents [parameterized code]
- Execution state
  - players, messages, event history
  - players have knowledge component
- Execution rules -- effect of actions
  - Theorem: event history ~ bundle

Semantics
To Do

Symbolic execution

Extend basic execution rules with rules for

- Adding players and past actions via rules for what must have happened (ala CPSA)
- Adding to player's knowledge (PDA)
- Inferring what intruder must have known

Execution based semantics

- PVS formalization of execution model
- Formal connection to abstract strand model
- Represent protocol composition logic
- Verify symbolic execution and reasoning rules
- Goal: a basis set of verified rules for developing analysis algorithms

Futuristic

- Formalizing use of schema (VC for PVS)
- Formalizing different levels of agreement
- Interleaving reasoning and execution
- Interleaving reasoning and refinement/composition
- Alice can reason about what Bob might infer ...

???