

How to Break a Protocol

Joshua D Guttman

F. Javier Thayer

The MITRE Corporation

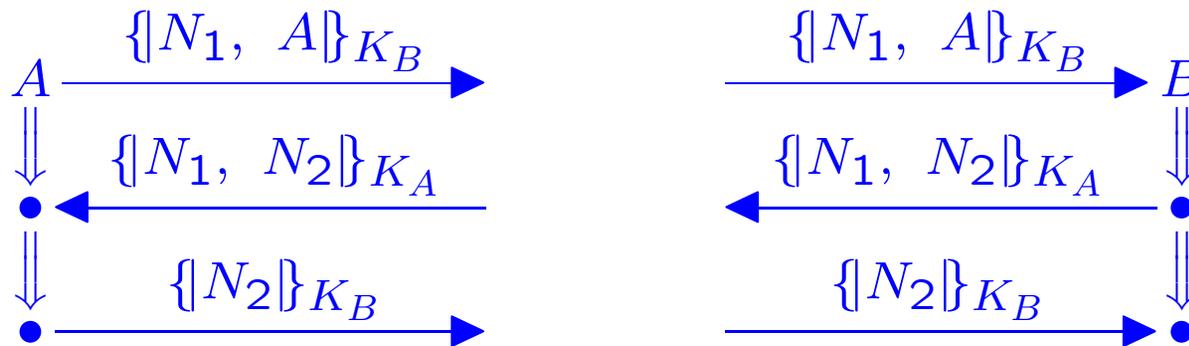
<http://www.ccs.neu.edu/home/guttman>

Thanks to support from: National Security Agency

How to Break a Protocol

- Try to prove it correct
 - Where you get stuck
that's where the flaw is
- Focus on services provided by protocol
 - Actions the protocol requires regular principals to perform
 - Produce values useful to penetrator

Needham-Schroeder



K_A, K_B

N_1, N_2

$\{t\}_K$

$N_1 \oplus N_2$

Public (asymmetric) keys of A, B

Nonces, one-time random bitstrings

Encryption of t with K

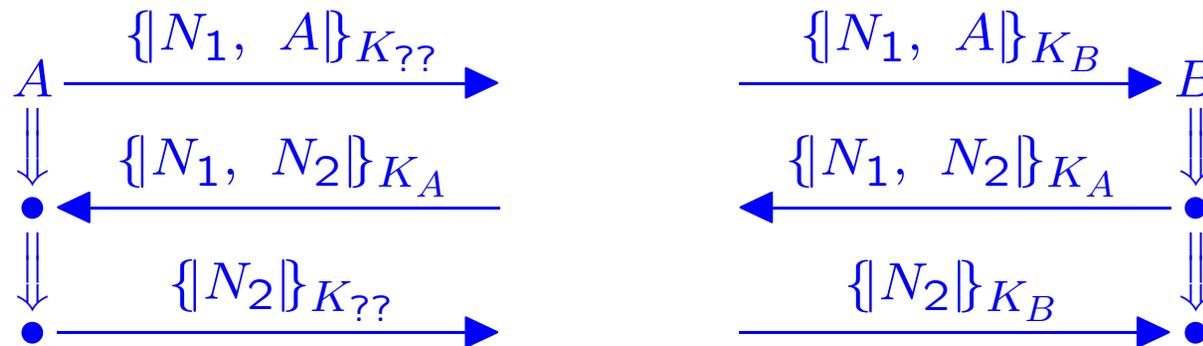
New shared secret

(whitespace)

Essence of Cryptography (for today's lecture)

- Symmetric key cryptography: algorithm using a single value, shared as a secret between sender, receiver
 - Same key makes ciphertext, extracts plaintext
- Public key cryptography: algorithm using two related values, one private, the other public
 - Encryption: Public key makes ciphertext, only private key owner can decrypt
 - Signature: Private key makes ciphertext, anyone can verify signature with public key
- Terminology: A 's public key: K_A A 's private key: K_A^{-1}
In symmetric crypto, $K = K^{-1}$
- Uncompromised key:
 - Key used only in accordance with protocol

Needham-Schroeder: How does it work?



Assume A 's private key K_A^{-1} uncompromised

Public (asymmetric) keys of A, B

Nonces, one-time random bitstrings

Encryption of t with K

New shared secret

K_A, K_B

N_1, N_2

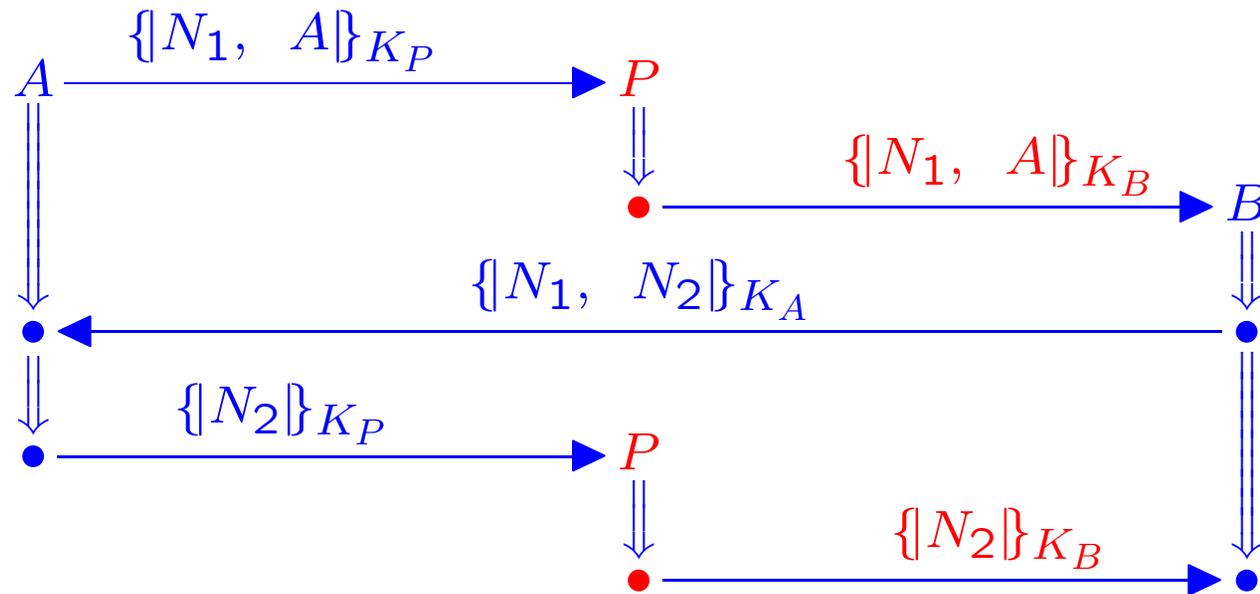
$\{t\}_K$

$N_1 \oplus N_2$

Whoops

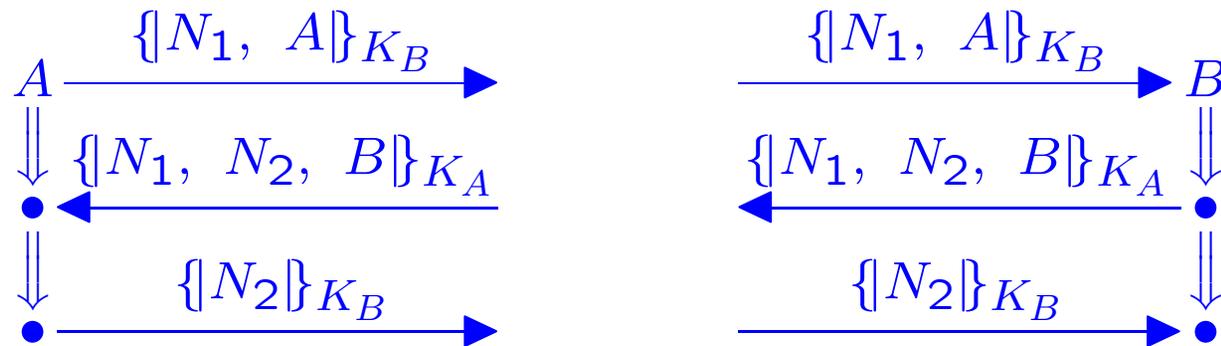
Needham-Schroeder Failure

If $?? = P$,



(Gavin Lowe)

Needham-Schroeder-Lowe



K_A, K_B

Public (asymmetric) keys of A, B

N_1, N_2

Nonces, one-time random bitstrings

$\{t\}_K$

Encryption of t with K

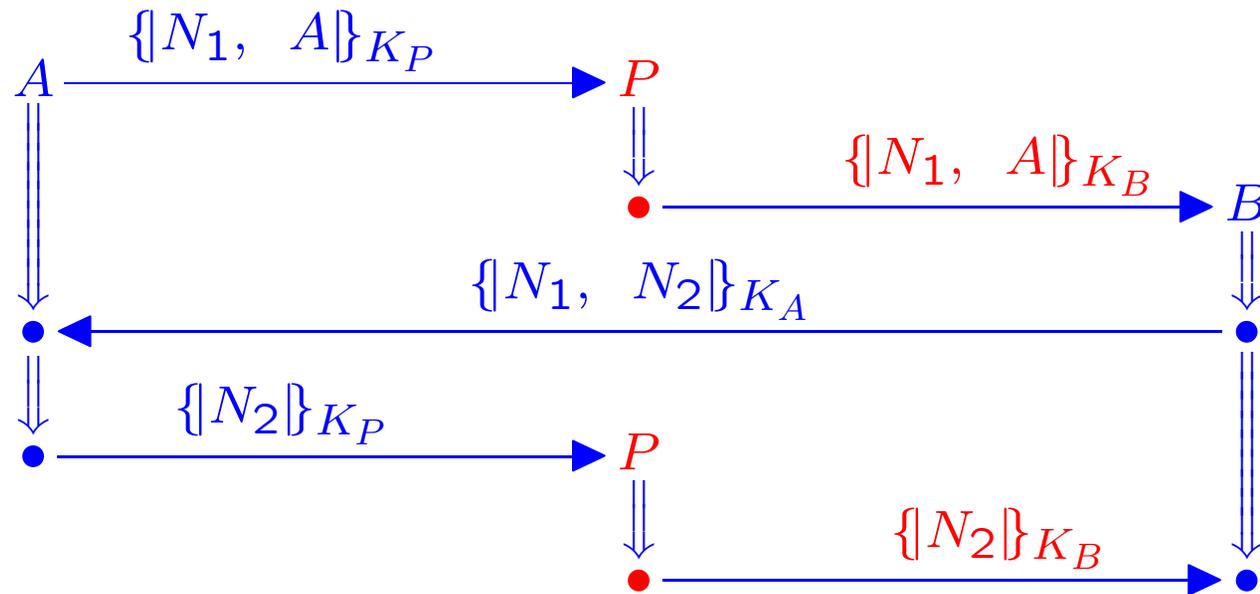
$N_1 \oplus N_2$

New shared secret

**How to Break Protocols:
Unintended Services
and
Junk Terms**

Needham-Schroeder Failure

If $?? = P$,

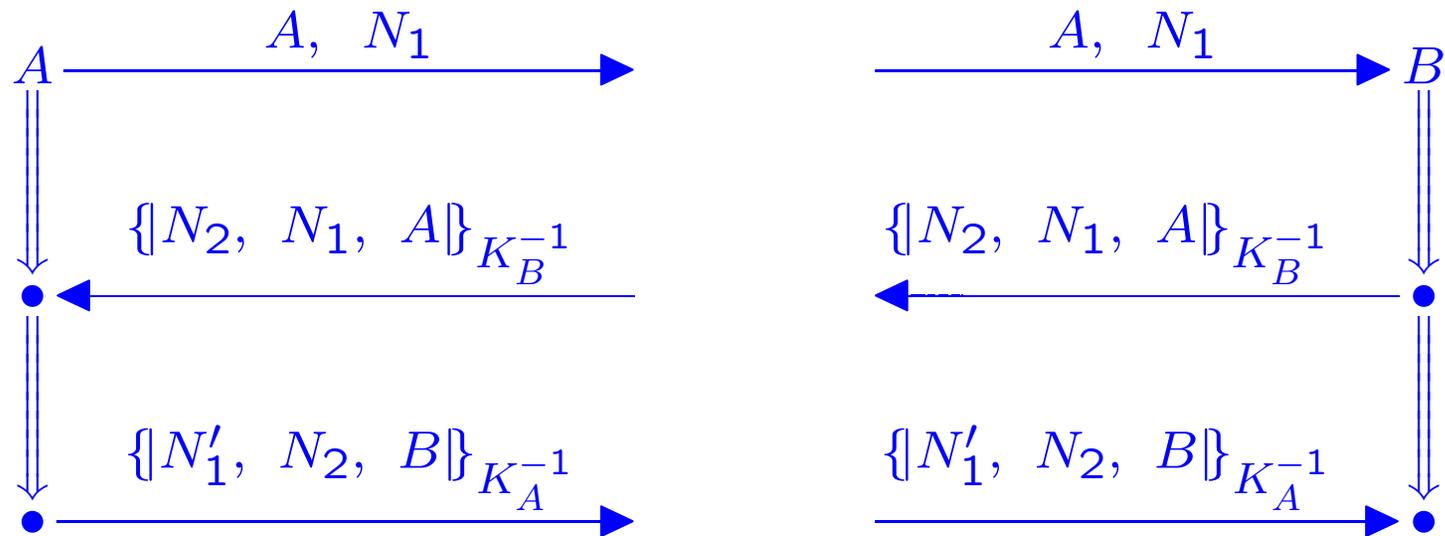


(Gavin Lowe)

Diagnosis of a Failure

- Who was duped?
 - Not A : Meant to share N_1, N_2 with P
 - B : Thinks he shares N_1, N_2 only with A
 - Secrecy failed: P knows values
 - Authentication failed:
 A had no run with B
- How? A offered P a service:
 - Gave P nonce N_1
 - Promised to translate $\{N_1, N\}_{K_A}$ to $\{N\}_{K_P}$
- An “unintended service”
 - Attacker needs to compute some value
 - N_2 in this case
 - But legitimate party creates such a value

Another Example: ISO Reject



Signatures only

Mere authentication

Diagnosis of ISO

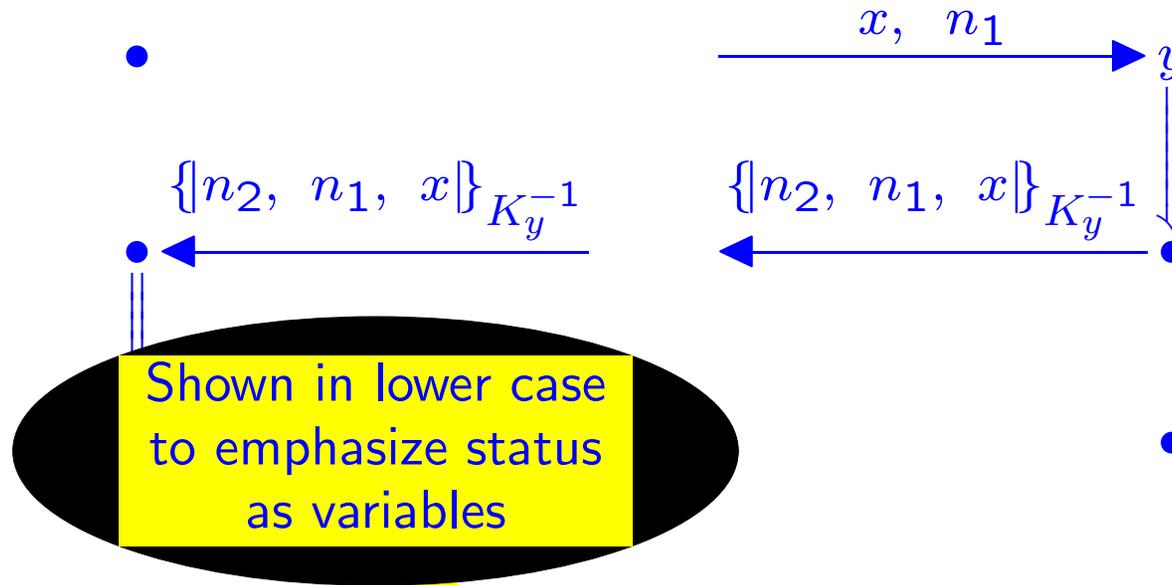
- Respondent B gets only two messages
 - Clearly A , N_1 is “junk”
 - It has no authenticating force
 - Other term received is the only challenge
- Attacker needs to create

$$\{N'_1, N_2, B\}_{K_A^{-1}}$$

Only $\{N'_1, N_2, B\}_{K_A^{-1}}$ requires work

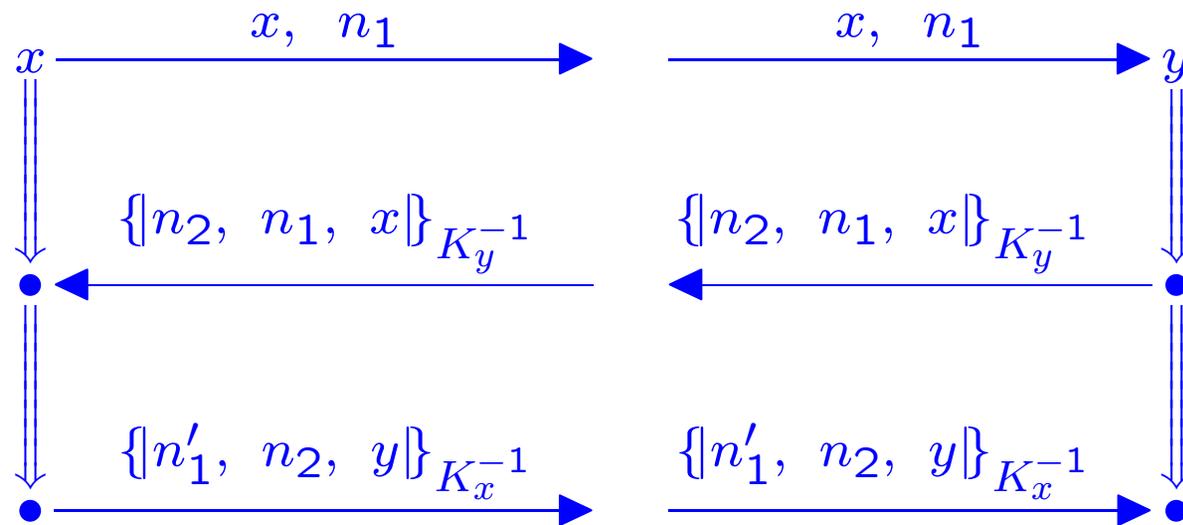
- What services are useful?

The Available Services



- May rename **in-bound** variables
- Want to produce $\{N'_1, N_2, B\}_{K_A^{-1}}$
for some N'_1
- Can use A as respondent, B, N_2 in-bound
i.e. use substitution $[A/y, B/x, N_2/n_1]$

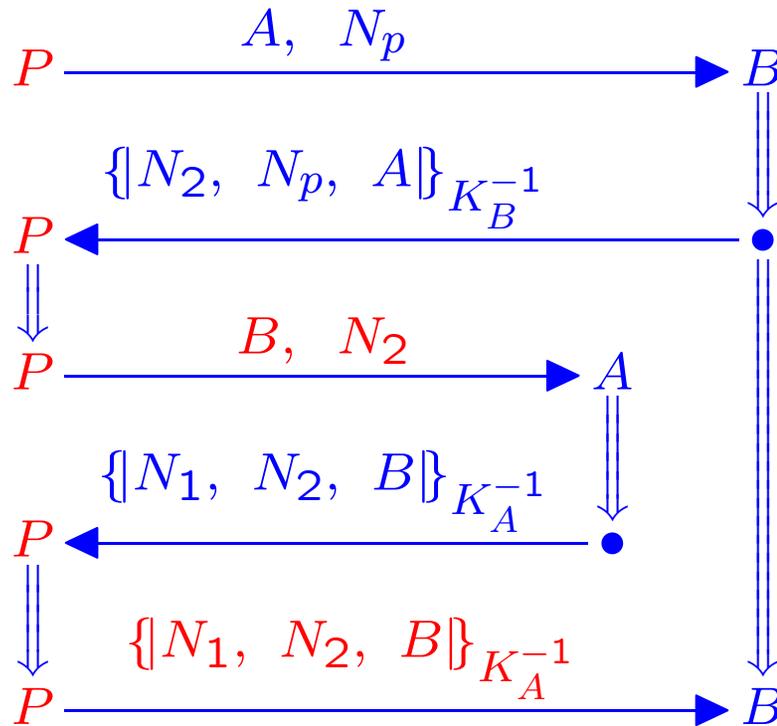
Behaviors are Parametric



x, y, n_1, n_2, n'_1 are variables

Possible behaviors are all substitution instances

Counterexample to One Security Goal



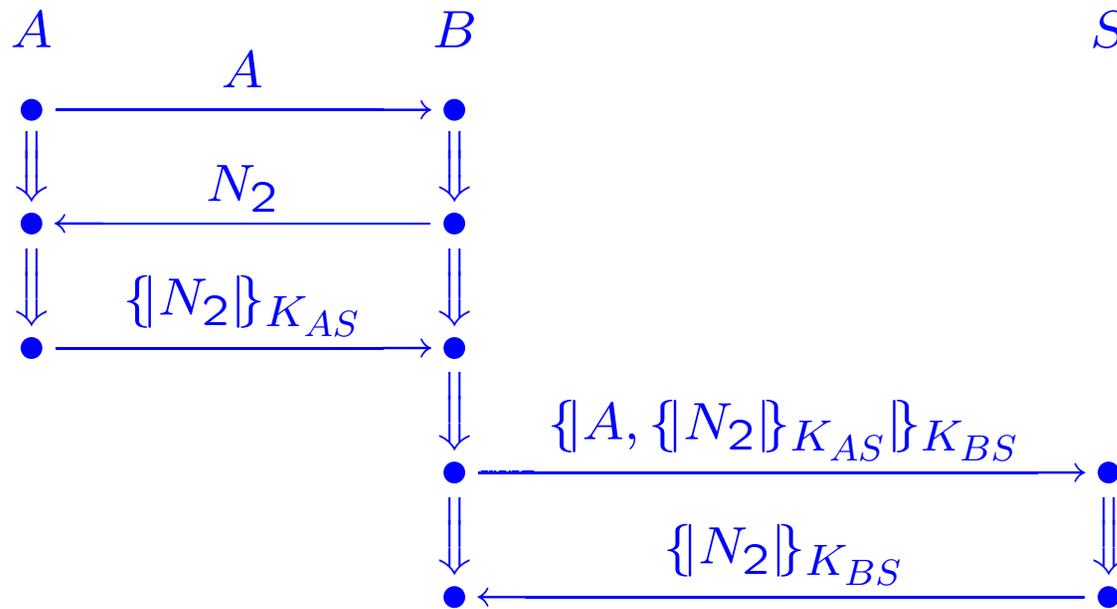
What Goal is Refuted?

- A executed a signature
 - “Entity authentication” for A may hold depending what that means
- But A was not initiator in any run with B

Dolev-Yao Attacks: A Recipe

- Identify and discard “junk” messages
 - They don’t contribute to authentication
 - Remaining incoming messages: “Challenge”
 - Adversary needs to synthesize them
- Look for unintended services
 - Criterion: Can they build challenge messages?
- Combine unintended services

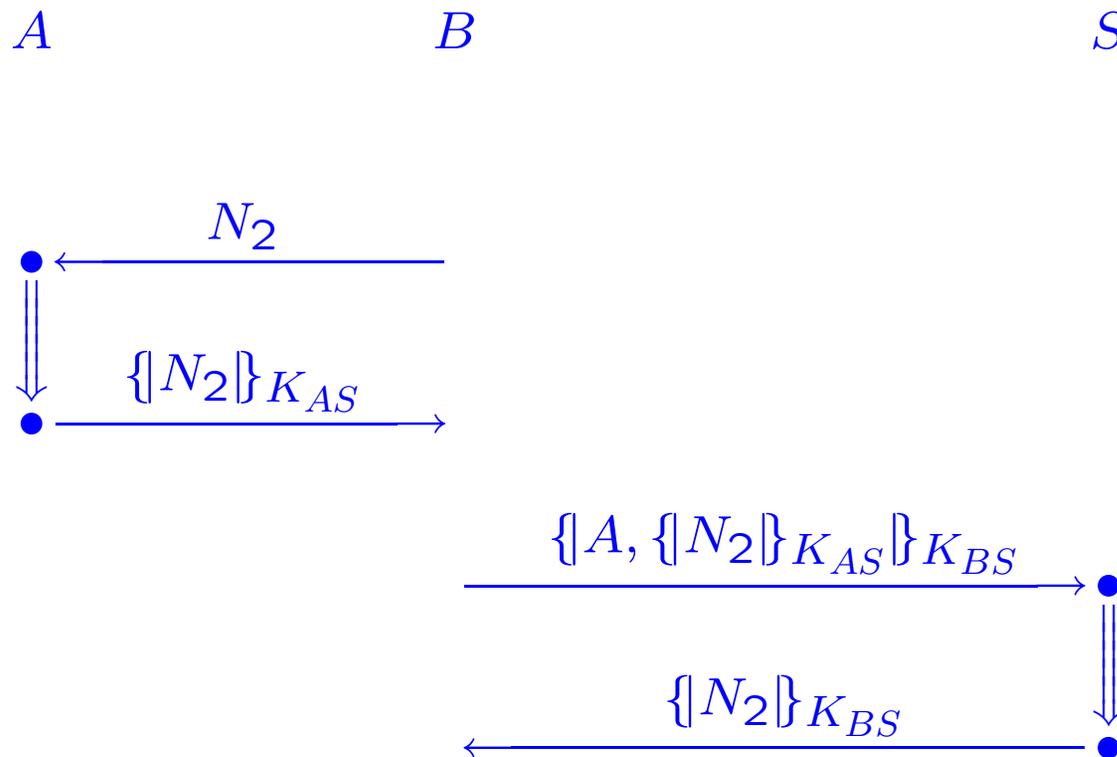
Example with Symmetric Crypto



Woo-Lam protocol

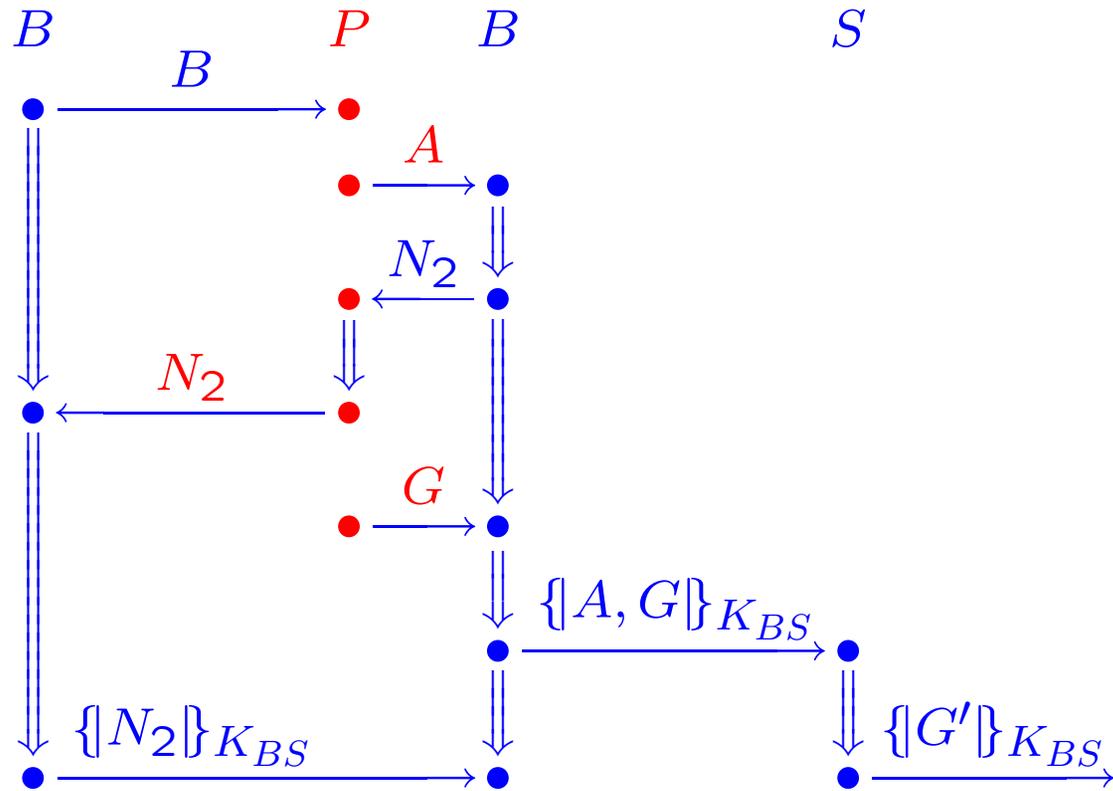
A and $\{N_2\}_{K_{AS}}$ are junk terms to B
 $\{N_2\}_{K_{BS}}$ only non-junk term

Woo-Lam Unintended Services

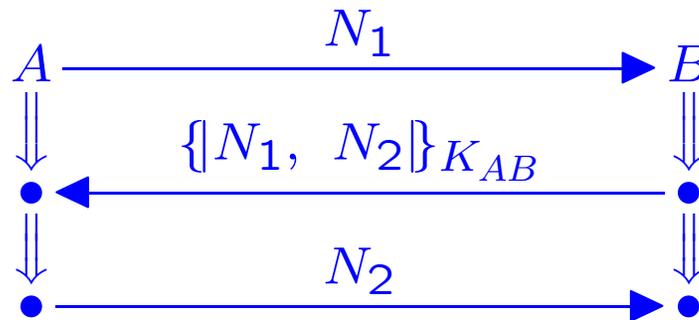


Both could produce $\{\text{nonce}\}_{\text{key}}$

Woo-Lam Infiltrated, I



Exercise (due to Song/Perrig)



What are the junk terms for B ?

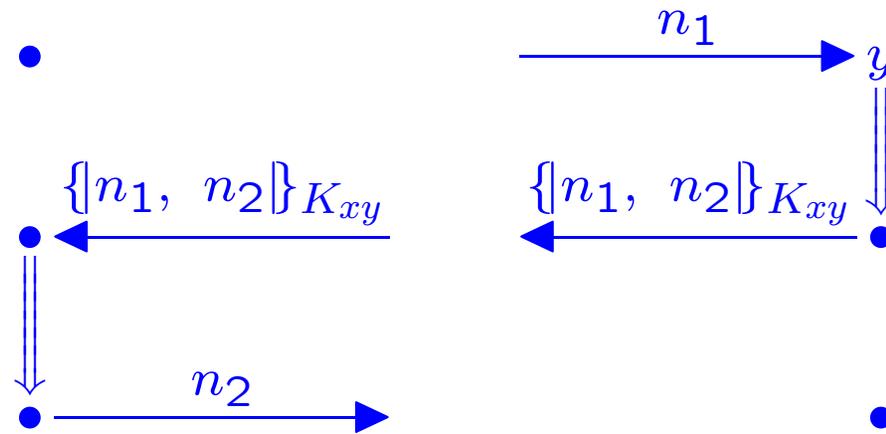
For A ?

Which terms have the “authenticating force”?

What are the services?

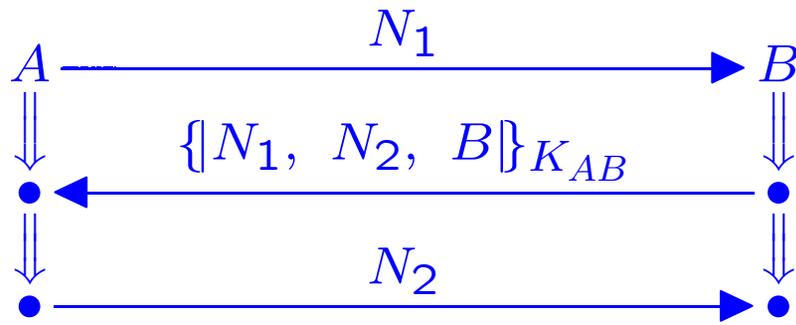
Is there an attack?

Exercise: Available Services



- To dupe initiator A , send back nonce N_1 to A as respondent
- I.e. use substitution $[A/y, N_1/n_1, K_{AB}/K_{xy}]$
- Resulting term $\{N_1, N_2\}_{K_{AB}}$ tricks A as initiator

Correction (due to Song/Perrig)



What Unintended Services Occur?

Signature: $N_a \mapsto \{ N_a \}_{K^{-1}}$
Encryption: $N_a \mapsto \{ N_a \}_K$
Decryption: $\{ N_a \}_K \mapsto N_a$
Translation: $\{ N_a \}_K \mapsto \{ N_a \}_{K'}$

- Examples:

- Signature service: ISO reject protocol
- Encryption service: Woo-Lam
- Decryption service: None
(too obvious?)
- Key-translation service: NS PK

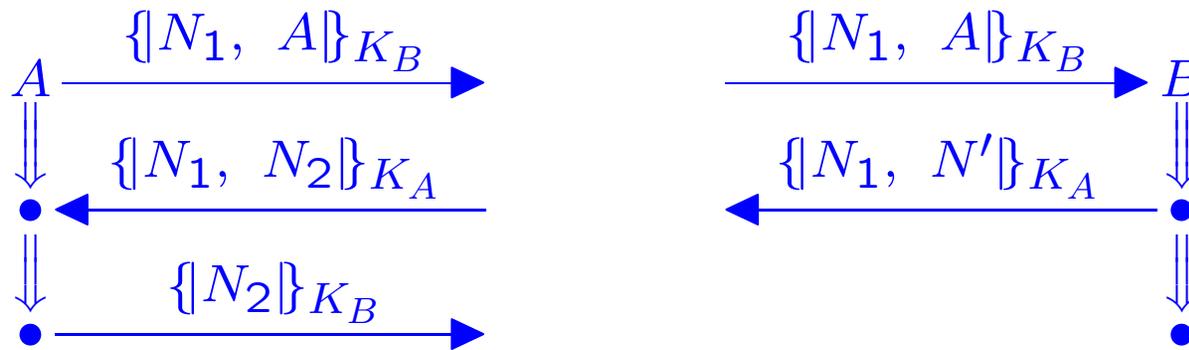
The Dolev-Yao Problem

- Given a protocol, and assuming all cryptography perfect, find
 - What **secrecy** properties
 - What **authentication** propertiesthe protocol achieves
- Find counterexamples to other properties
 - Unintended services useful
- What does perfect cryptography mean?
 - No collisions
 - Need key to make encrypted value
 - Need key to decrypt and recover plaintext

How to Prove a Protocol Correct

- Try to break it
 - When you get stuck you'll see why it's right

Needham-Schroeder: Initiator's View



Assume A, B 's private keys K_A^{-1}, K_B^{-1} uncompromised

K_A, K_B

Public (asymmetric) keys of A, B

N_1, N_2

Nonces, one-time random bitstrings

$\{t\}_K$

Encryption of t with K

$N_1 \oplus N_2$

New shared secret

Does $N' = N_2$? Yes, there are no available services!

Summary

- How to break a protocol
 - Try to prove it correct
 - Where you get stuck, look for trouble
 - Specifically, look for unintended services to produce non-junk terms expected by regular principals
- How to prove a protocol correct
 - Try to break it
 - See what unintended services must be used
 - “Read off” authentication properties
- Strand spaces: make these ideas precise, justify method

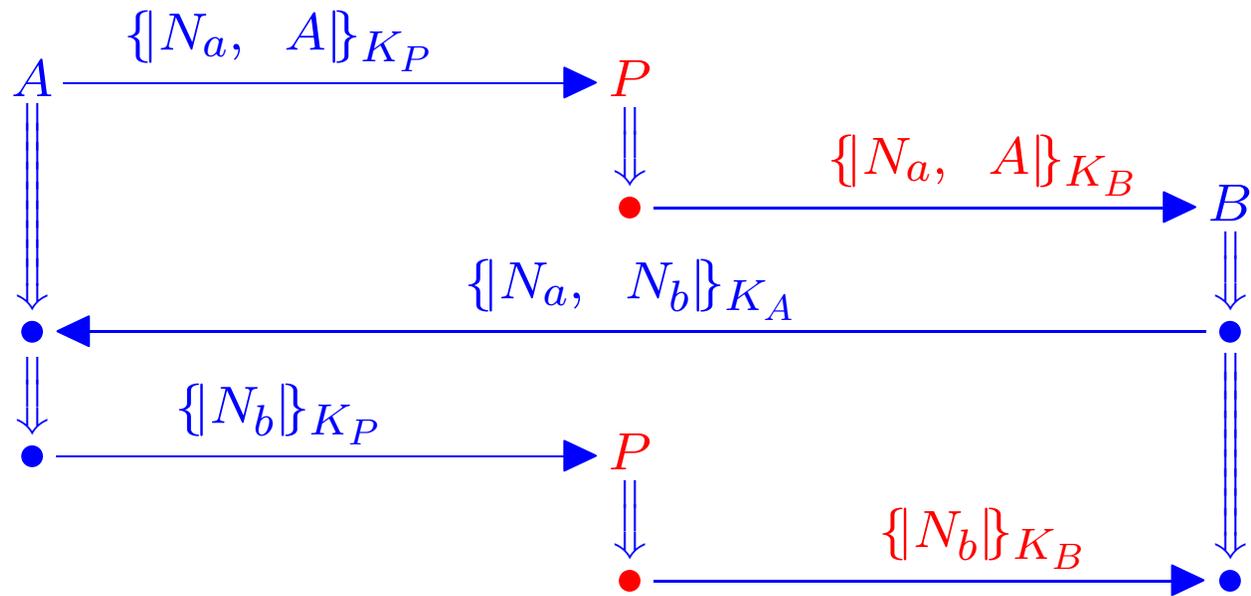
Strand Spaces

**work done jointly with
Javier Thayer and Jonathan Herzog**

Protocol Executions are Bundles

- Send, receive events on strands called “nodes”
 - Positive for send
 - Negative for receive
- Bundle \mathcal{B} : Finite graph of nodes and edges representing causally well-founded execution; Edges are arrows \rightarrow, \Rightarrow
 - For every reception $-t$ in \mathcal{B} , there’s a unique transmission $+t$ where $+t \rightarrow -t$
 - When nodes $n_i \Rightarrow n_{i+1}$ on same strand, if n_{i+1} in \mathcal{B} , then n_i in \mathcal{B}
 - \mathcal{B} is acyclic

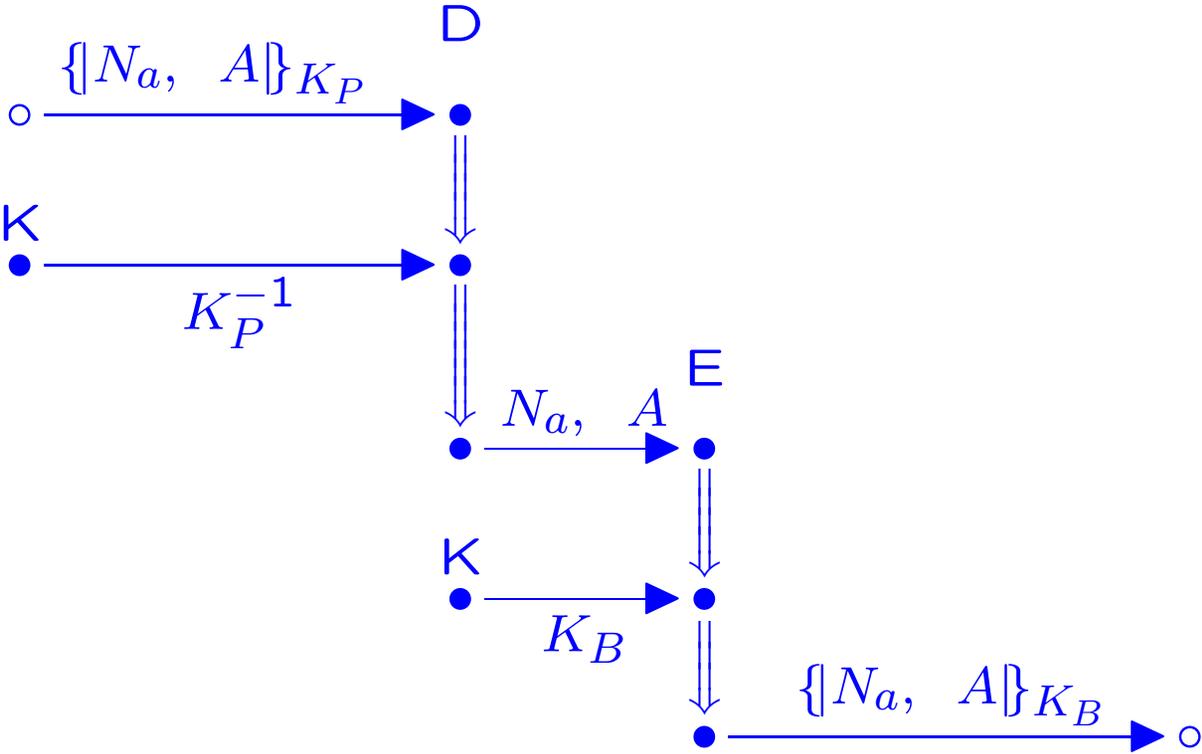
A Bundle



Precedence within a Bundle

- Bundle precedence ordering $\preceq_{\mathcal{B}}$
 - $n \preceq_{\mathcal{B}} n'$ means sequence of 0 or more arrows \rightarrow, \Rightarrow lead from n to n'
 - $\preceq_{\mathcal{B}}$ is a partial order by acyclicity
 - $\preceq_{\mathcal{B}}$ is well-founded by finiteness
- Bundle induction: Every non-empty subset of \mathcal{B} has $\preceq_{\mathcal{B}}$ -minimal members
- Reasoning about protocols combines
 - Bundle induction
 - Induction on message structure

NS Attack: Adversary Activity



Messages

- Terms freely generated from
 - Names, texts
 - Nonces
 - Keys

using the operators:

- Concatenation t_0, t_1
- Encryption with a key $\{t_0\}_K$
- Other algebras also interesting
but today we'll use the free one

Subterms and Origination

- Subterm relation \sqsubset
least transitive, reflexive relation with

$$g \sqsubset g, \quad h$$

$$h \sqsubset g, \quad h$$

$$h \sqsubset \{h\}_K$$

N.B. $K \sqsubset \{h\}_K$ implies $K \sqsubset h$

- Represents *contents* of message, not how it's constructed
- t **originates** at n_1 means

n_1 is a transmission (+)

$$t \sqsubset \text{term}(n_1)$$

if $n_0 \Rightarrow \dots \Rightarrow n_1$, then $t \not\sqsubset \text{term}(n_0)$

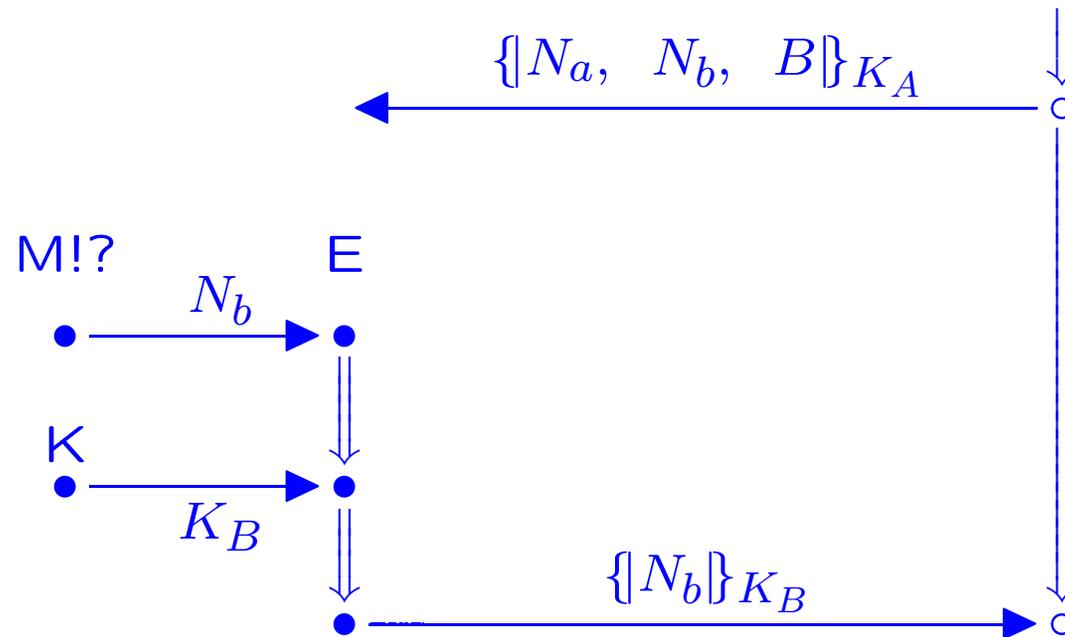
- Unique origination, non-origination formalize a probabilistic assumption

An Authentication Goal

- Suppose:
 - Bundle \mathcal{B} contains a strand $\text{Resp}[A, B, N_a, N_b]$
 - K_A^{-1} non-originating
 - N_b originates uniquely in \mathcal{B}
 - $N_b \neq N_a$
- Then:
 - There is a strand $\text{Init}[A, B, N_a, N_b]$ in \mathcal{B}

Authentication: correspondence assertions (of form $\forall\exists$)
(This is false for NS)

Guessing a Nonce



Guessing a private key (e.g. K_A^{-1})
similarly improbable

A Secrecy Goal

- Suppose:
 - Bundle \mathcal{B} contains a strand $\text{Resp}[A, B, N_a, N_b]$
 - K_A^{-1}, K_B^{-1} non-originating
 - N_b originates uniquely in \mathcal{B}
- Then:
 - There is no node $n \in \mathcal{B}$ with $\text{term}(n) = N_b$

Form: \forall

This also is false for NS

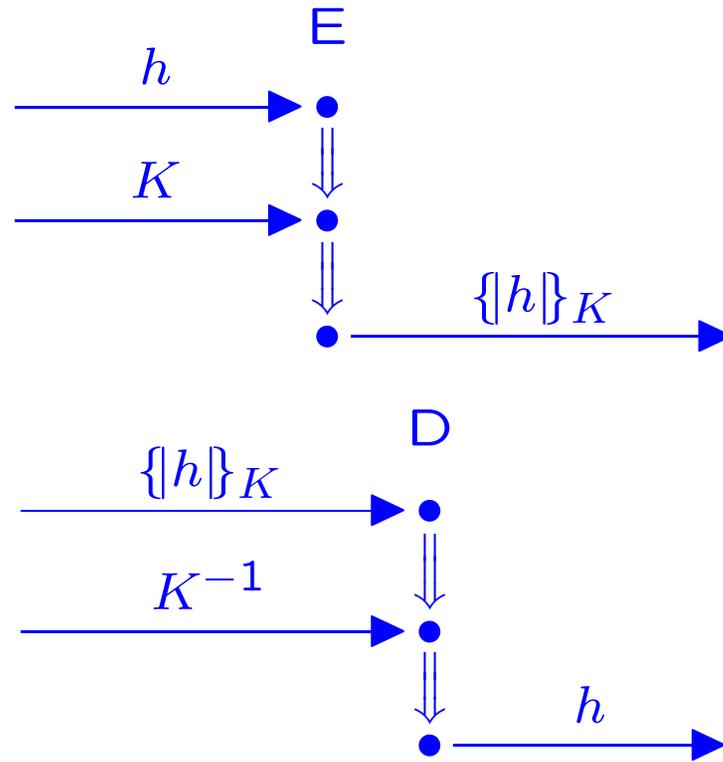
Summary: Breaking Protocols, Strand Spaces

- To break a protocol, you
 - Discard junk terms
 - Identify unintended services
 - Match services against non-junk goals
- Core strand space ideas:
 - Behaviors (regular or adversary) are strands
 - Executions are bundles
 - Unique origination and non-origination
- Security goals:
 - Authentication asserts existence of matching strand
 - Secrecy asserts non-existence of “disclosing” nodes
 - Premises concern n.o., u.o., existence of strands, inequalities
- Tomorrow: How would you prove these goals?

Adversary Strands, I: Initiating Values



Adversary Strands, II: Encrypt, Decrypt



Formalizes notion of ideal cryptography

Adversary Strands, III: Concatenate, Separate

