Security Analysis of Network Protocols

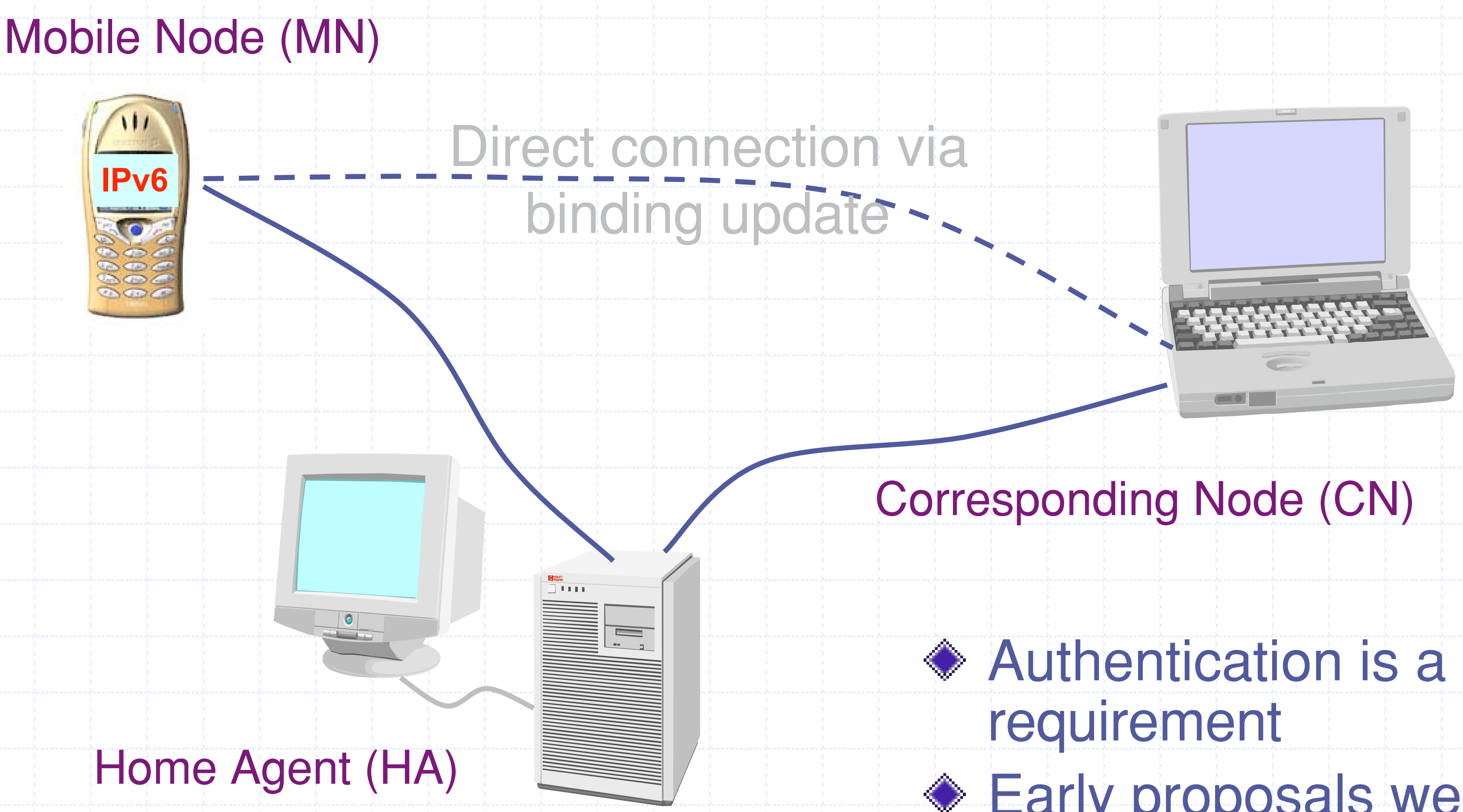
John Mitchell
Stanford University

Usenix Security Symposium, 2008

Many Protocols

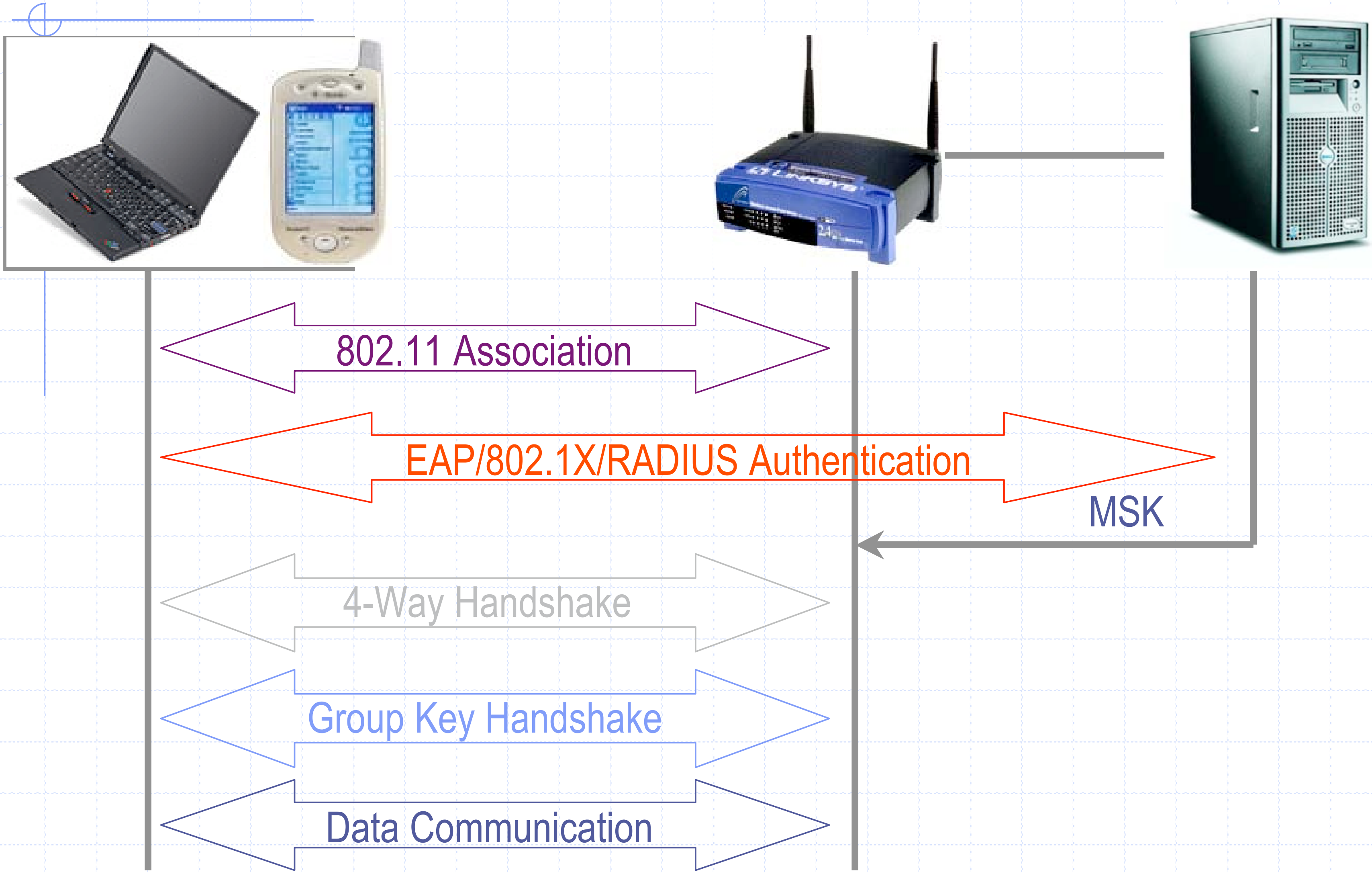
- Authentication and key exchange
  - SSL/TLS, Kerberos, IKE, JFK, IKEv2,
- Wireless and mobile computing
  - Mobile IP, WEP, 802.11i
- Electronic commerce
  - Contract signing, SET, electronic cash, …
- And more
  - Web services, …

Mobile IPv6 Architecture

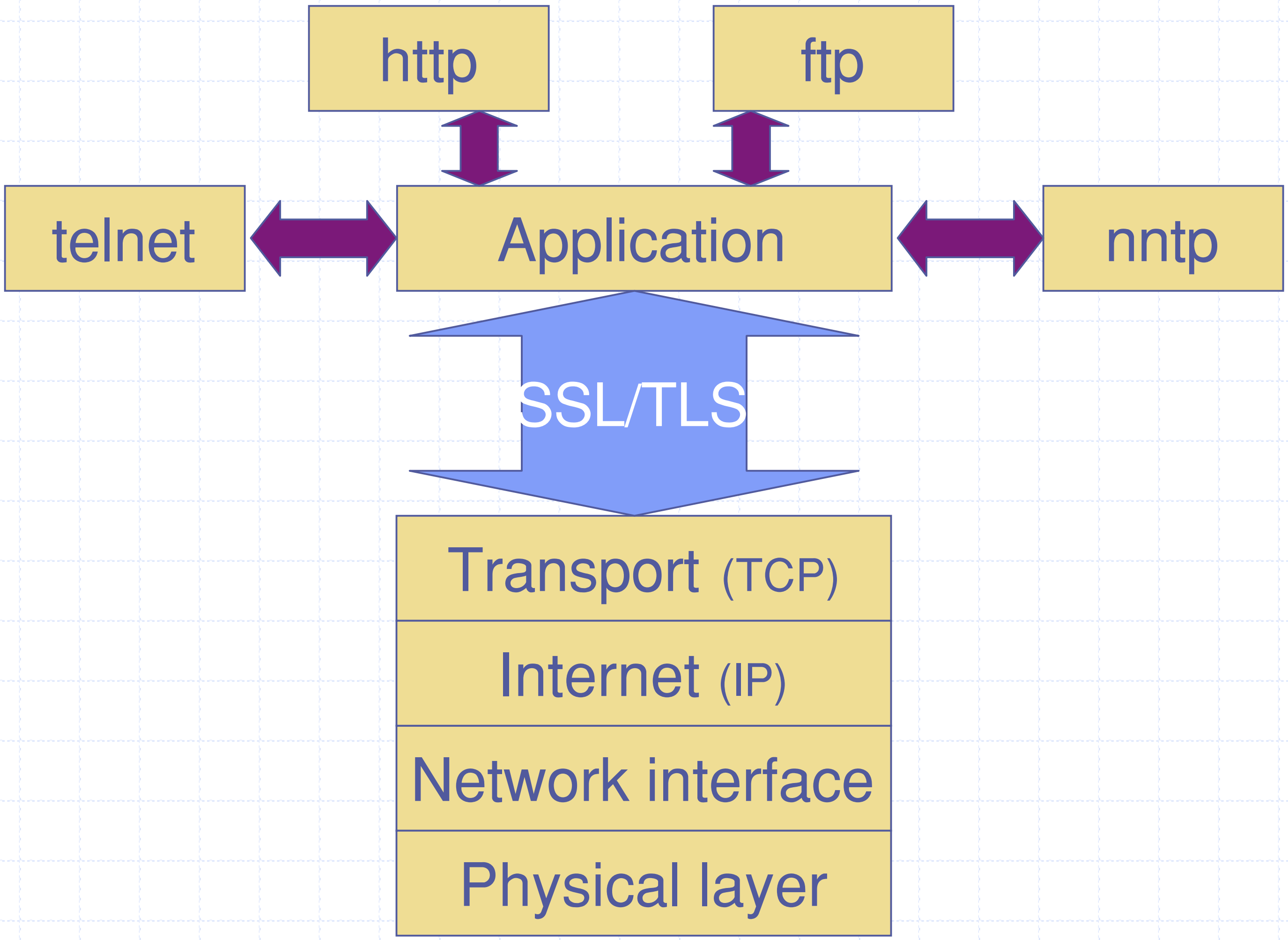


- Authentication is a requirement
- Early proposals weak

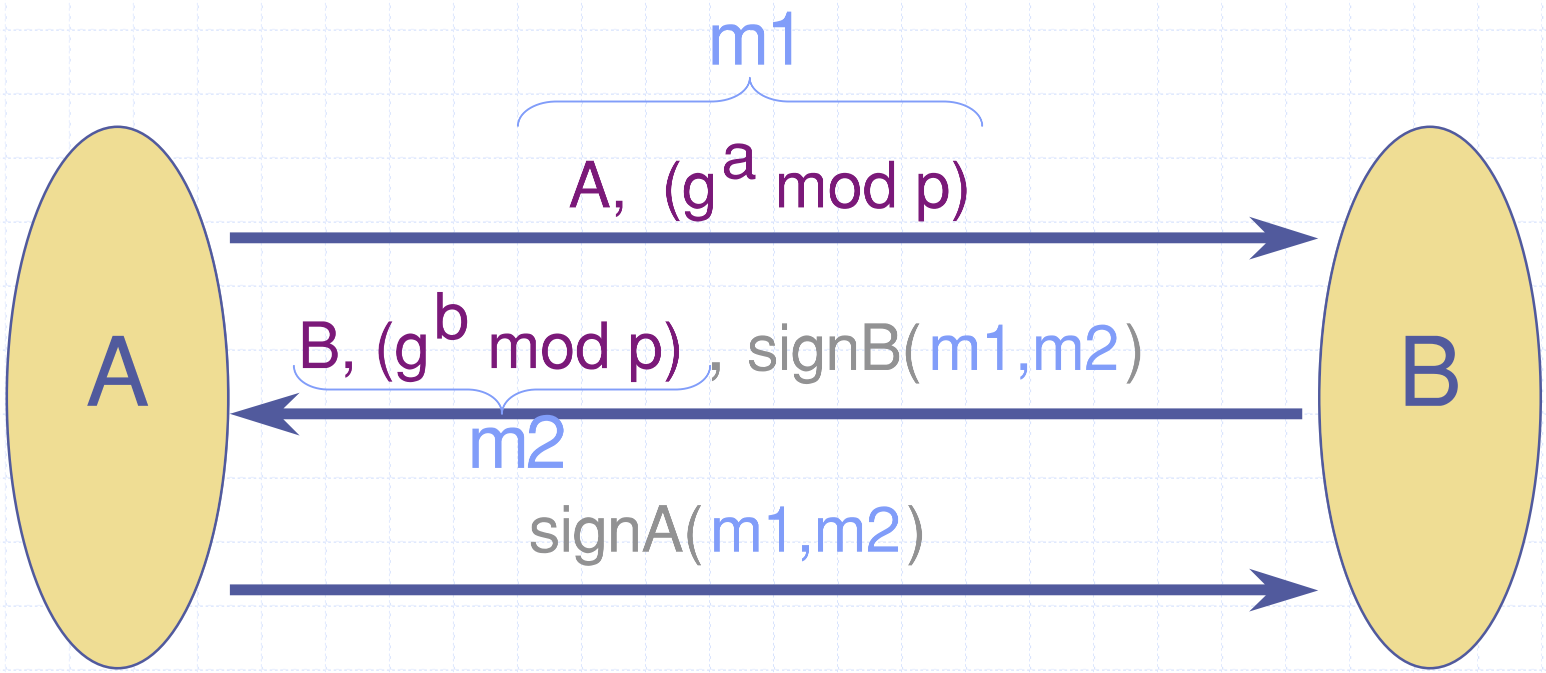
802.11i Wireless Authentication



TLS protocol layer over TCP/IP



IKE subprotocol from IPSEC



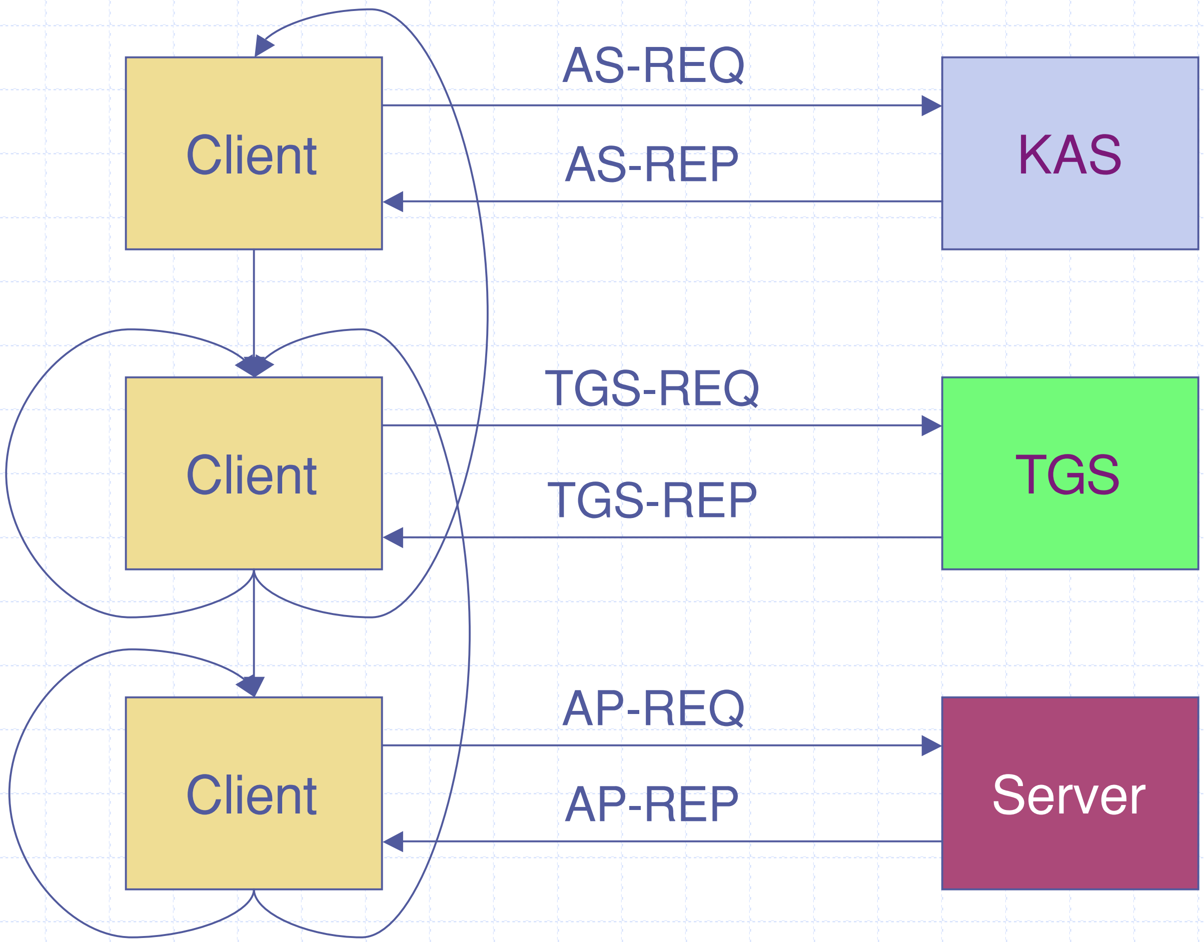
Result: A and B share secret $g^{ab} \bmod p$

Analysis involves probability, modular exponentiation, complexity, digital signatures, communication networks

Protocol Attacks

- Kerberos [Scederov et. Al.]
  - Public key version - lack of identity in message causes authentication failure
- WLAN 802.11i [He , Mitchell]
  - Lack of authentication in msg causes dos vulnerability
  - Proved correct using PCL [Datta , Derek, Sundararajan]
- GDOI [meadows – Pavlovic]
  - Authorization failure
- SSL [Mitchell – Shmatikov]
  - Version roll-back attack, authenticator confusion between main and resumption protocol
- Needham-Schroeder [Lowe]
  - We will look at this today

Kerberos Protocol



Used for network authentication

Microsoft TechNet

Microsoft Security Bulletin MS05-042

Vulnerabilities in Kerberos Could Allow Denial of Service, Information Disclosure, and Spoofing (899587)

Published: August 9, 2005

Affected Software:

- Microsoft Windows 2000 Service Pack 4
- Microsoft Windows XP Service Pack 1 and Microsoft Windows XP Service Pack 2
- Microsoft Windows XP Professional x64 Edition
- Microsoft Windows Server 2003 and Microsoft Windows Server 2003 Service Pack 1
- Microsoft Windows Server 2003 for Itanium-based Systems and Microsoft Windows Server 2003 with SP1 for Itanium-based Systems
- Microsoft Windows Server 2003 x64 Edition

Credit: Cervesato, Jaggard, Scedrov, Tsay, Wals*t*ad

- Attack found in PKINIT-25; fixed in PKINIT-27
- Used in Windows and Linux (called Heimdal)
- Also in implementation by CableLabs (for cable boxes)

Attack on PKINIT (basic idea)

$C \xrightarrow{Cert_C,\ [t_C,\ n_2]_{skC},\ C,\ T,\ n_1} I$

$I \xrightarrow{Cert_I,\ [t_C,\ n_2]_{skI},\ I,\ T,\ n_1} K$

$I \xleftarrow{\{[k,\ n_2]_{skK}\}_{pkI},\ I,\ TGT,\ \{AK,\ \ldots\}_k} K$

$C \xleftarrow{\{[k,\ n_2]_{skK}\}_{pkC},\ C,\ TGT,\ \{AK,\ \ldots\}_k} I$

Data signed by KDC does not include name of client

Repair

C $\xrightarrow{Cert_C,\ [t_C,\ n_2]_{skC},\ C,\ T,\ n_1}$ I

I $\xrightarrow{Cert_I,\ [t_C,\ n_2]_{skI},\ I,\ T,\ n_1}$ K

K signs k, cksum instead of k, n_2

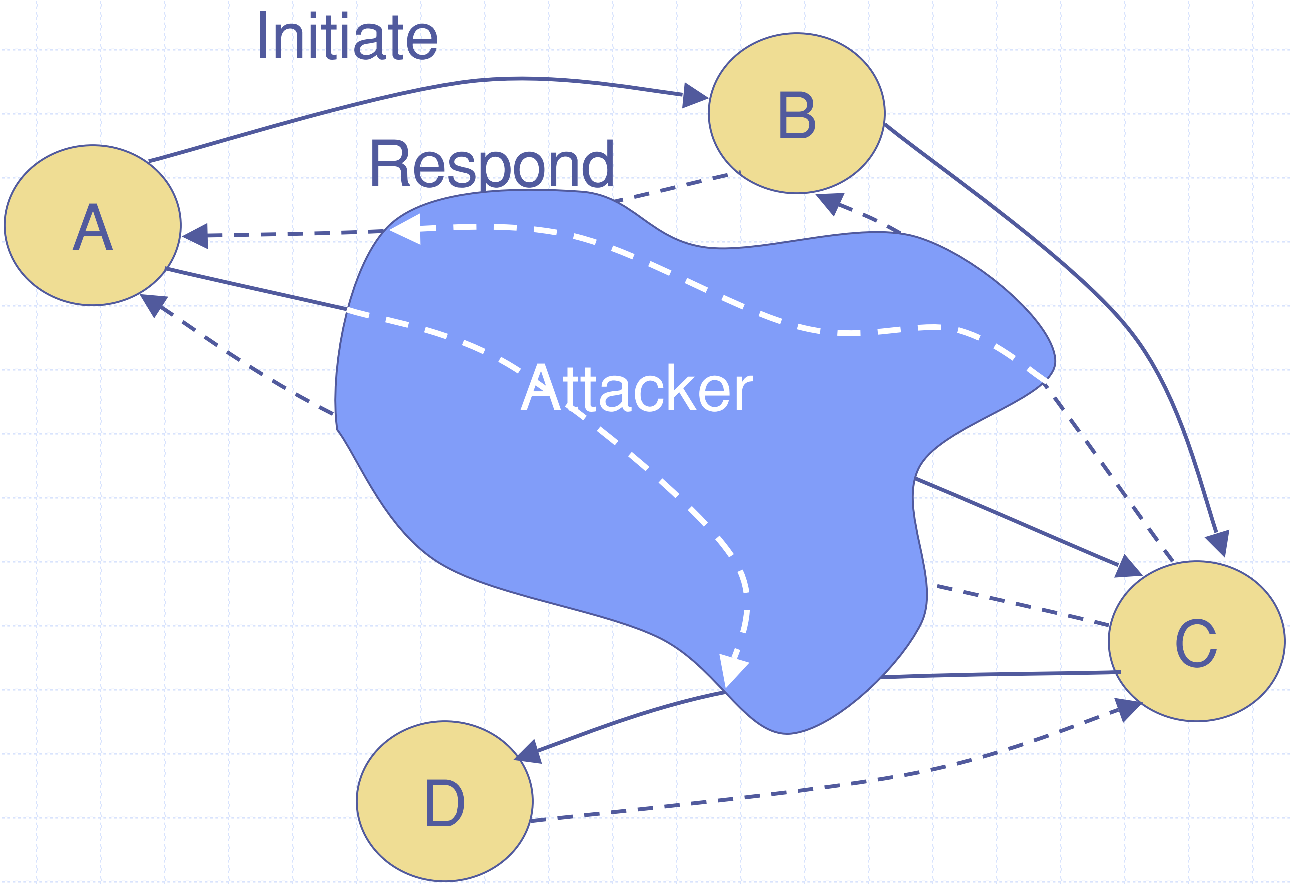
I $\xleftarrow{\{[k,\ n_2]_{skK}\}_{pkI},\ I,\ TGT,\ \{AK,\ \ldots\}_k}$ K

C $\xleftarrow{\{[k,\ n_2]_{skK}\}_{pkC},\ C,\ TGT,\ \{AK,\ \ldots\}_k}$ I

Main points of this talk

- Widely used protocols central to security
  - Worth designing correctly
  - Worth analyzing for bugs
  - Worth proving them correct
    - All methods use some simplifying assumptions
    - Diversity and overlap of methods is a good thing
- Develop basic science and engineering
  - New protocols are being developed
  - Methods can be used for other systems

Run of a protocol

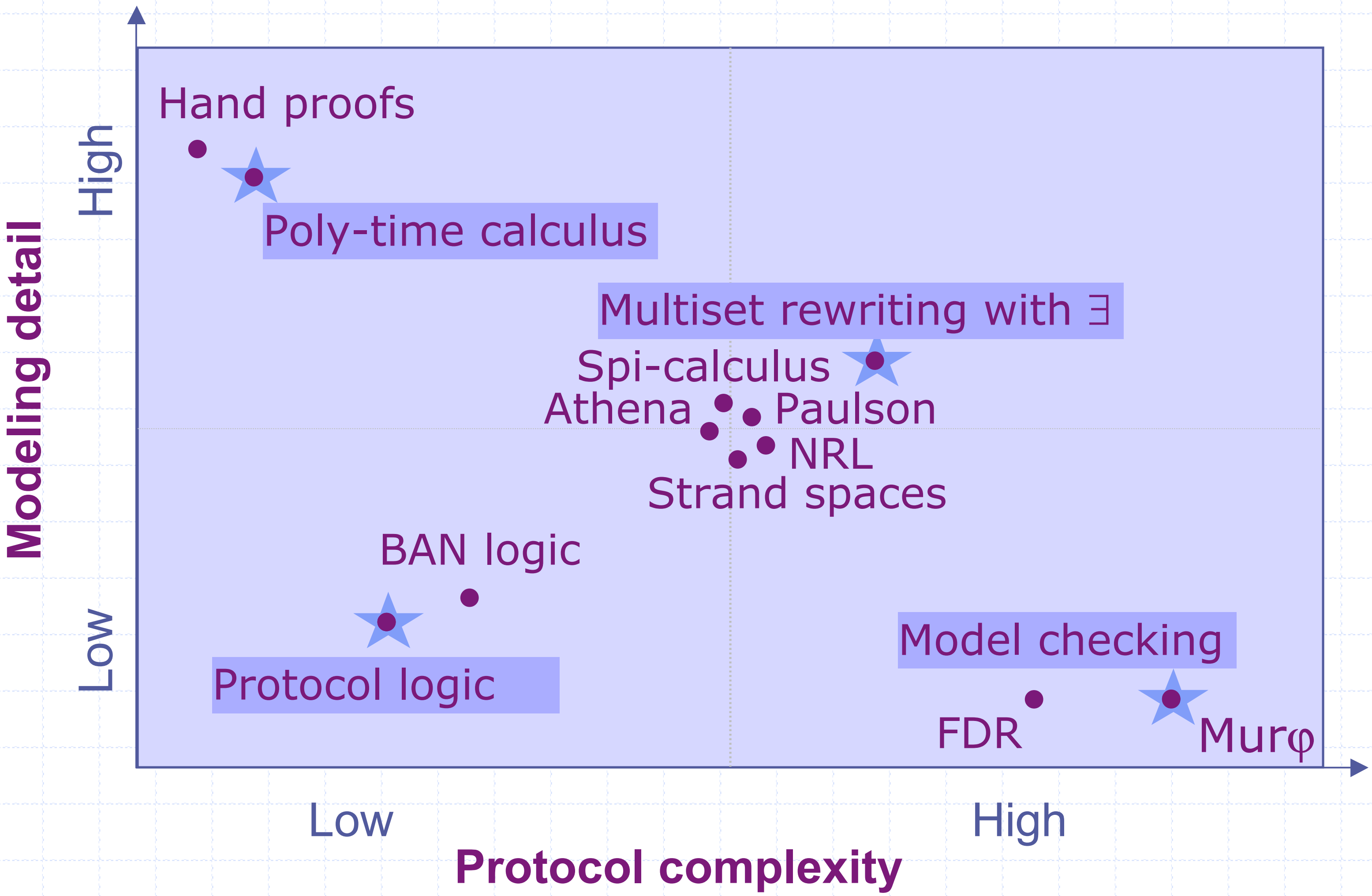


Correct if no security violation in any run

Protocol analysis methods

- Cryptographic reductions
  - Bellare-Rogaway, Shoup, many others
  - UC [Canetti et al], Simulatability [BPW]
  - Prob poly-time process calculus [LMRST…]
- Symbolic methods
  - Model checking
    - FDR [Lowe, Roscoe, …], Murphi [M, Shmatikov, …], …
  - Symbolic search
    - NRL protocol analyzer [Meadows], …
  - Theorem proving
    - Isabelle [Paulson …], Specialized logics [BAN, …]

Protocol analysis spectrum



“The” Symbolic Model

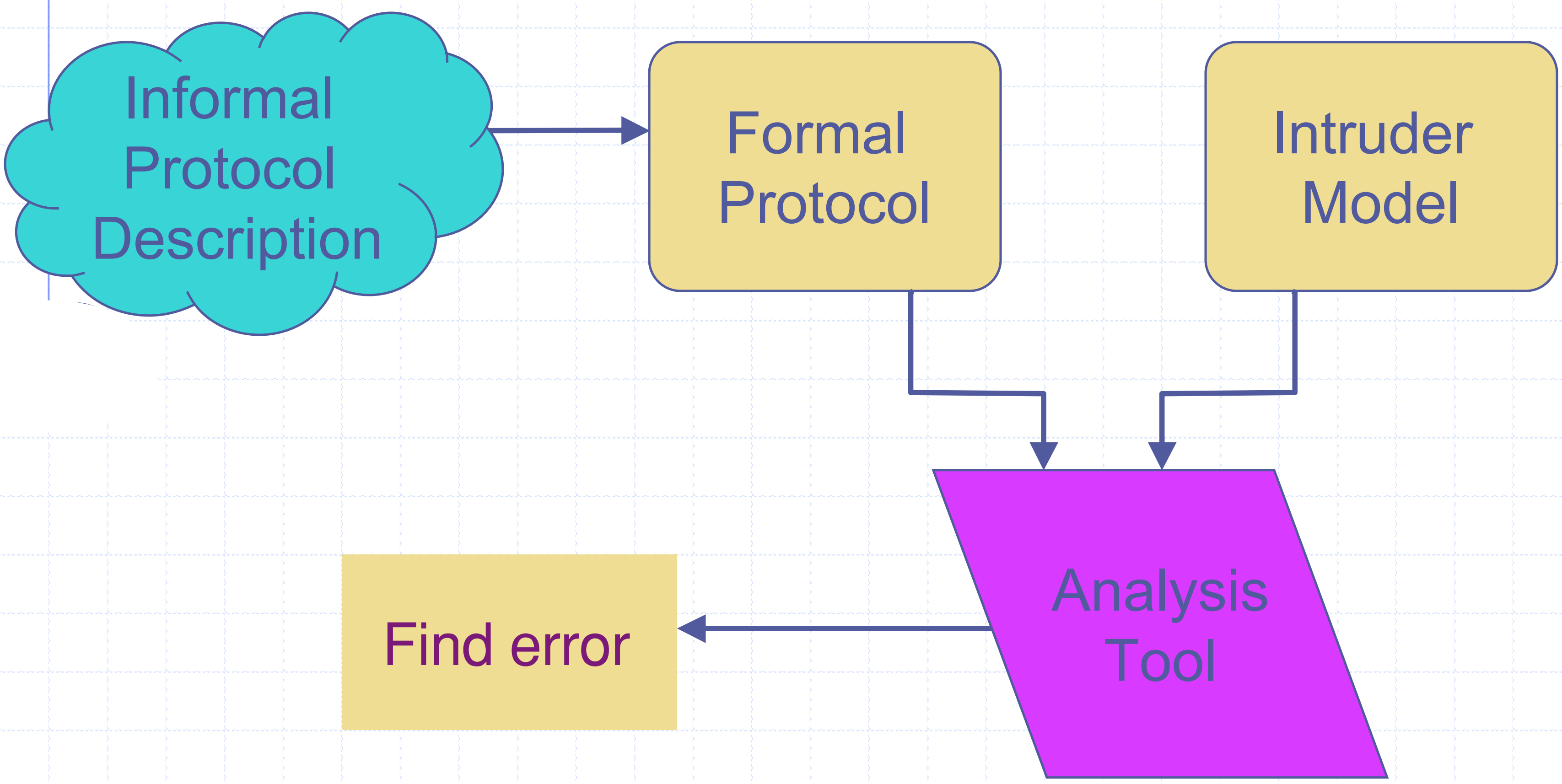
- Messages are algebraic expressions
  - Nonce, Encrypt(K,M), Sign(K,M), …
- Adversary
  - Nondeterministic
  - Observe, store, direct all communication
    - Break messages into parts
    - Encrypt, decrypt, sign only if it has the key
      - Example: $\langle$K1, Encrypt(K1, “hi”) $\rangle$
        $\Rightarrow$ K1, Encrypt(K1, “hi”) $\Rightarrow$ “hi”
    - Send messages derivable from stored parts

Many formulations

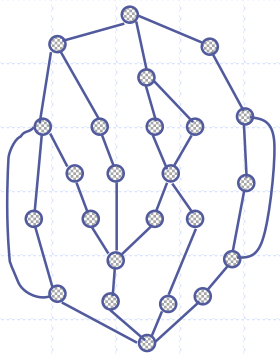
- Word problems [Dolev-Yao, Dolev-Even-Karp, …]
  - Protocol step is symbolic function from input message to output
- Rewrite systems [CDLMS, …]
  - Protocol step is symbolic function from state and input message to state and output message
- Logic programming [Meadows NRL Analyzer]
  - Each protocol step can be defined by logical clauses
  - Resolution used to perform reachability search
- Constraint solving [Amadio-Lugiez, …]
  - Write set constraints defining messages known at step i
- Strand space model [MITRE]
  - Partial order (Lamport causality), reasoning methods
- Process calculus [CSP, Spi-calculus, applied π, …)
  - Each protocol step is process that reads, writes on channel
  - Spi-calculus: use ν for new values, private channels, simulate crypto

Automated tools based on the symbolic model detect important, nontrivial bugs in practical, deployed, and standardized protocols

Explicit Intruder Method



Automated Finite-State Analysis



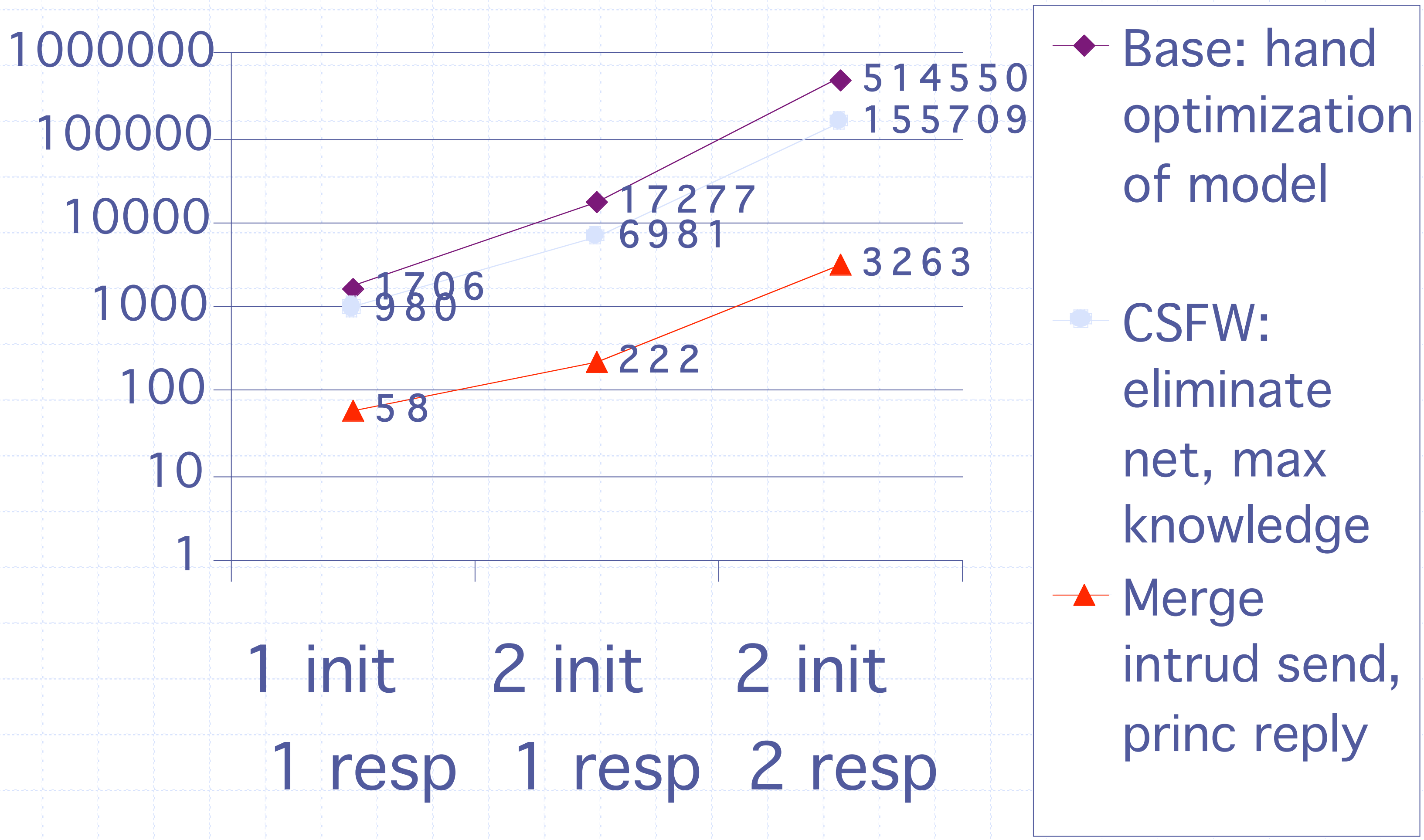
- Define finite-state system
  - Bound on number of steps
  - Finite number of participants
  - Nondeterministic adversary with finite options
- Pose correctness condition
  - Can be simple: authentication and secrecy
  - Can be complex: contract signing
- Exhaustive search using “verification” tool
  - Error in finite approximation $\Rightarrow$ Error in protocol
  - No error in finite approximation $\Rightarrow$???

Limitations

- System size with current methods
  - 2-6 participants

    Kerberos: 2 clients, 2 servers, 1 KDC, 1 TGS
  - 3-6 steps in protocol
  - May need to optimize adversary
- Adversary model
  - Cannot model randomized attack
  - Do not model adversary running time

State Reduction on N-S Protocol



Security Protocols in Murφ

- Standard “benchmark” protocols
  - Needham-Schroeder, TMN, …
  - Kerberos
- Study of Secure Sockets Layer (SSL)
  - Versions 2.0 and 3.0 of handshake protocol
  - Include protocol resumption
- Tool optimization
- Additional protocols
  - Contract-signing
  - Wireless networking

  … ADD YOUR PROJECT HERE …

Tool by Dill et al.

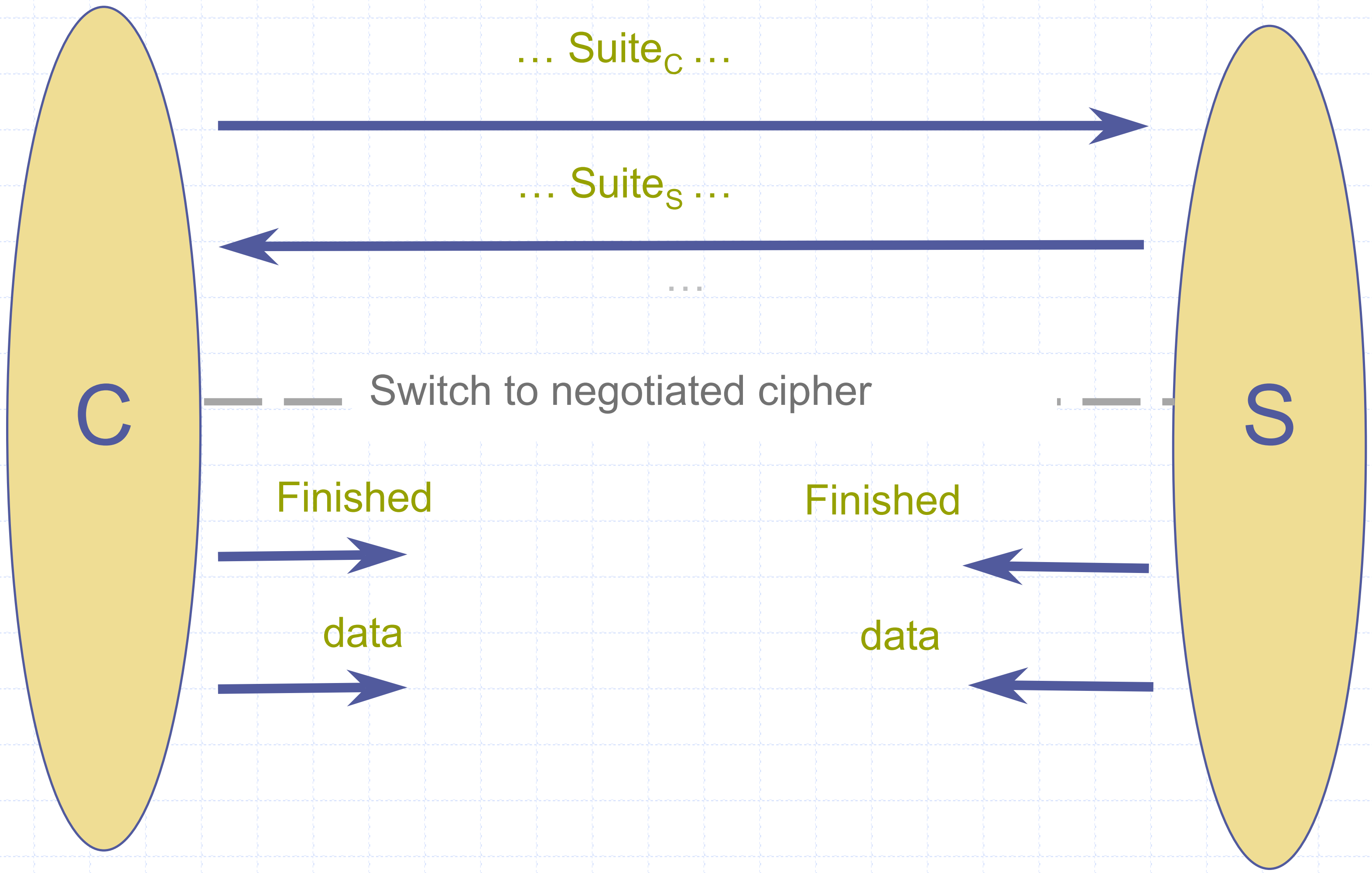
Rational Reconstruction (TLS)

- Begin with simple, intuitive protocol
  - Ignore client authentication
  - Ignore verification messages at the end of the handshake protocol
  - Model only essential parts of messages (e.g., ignore padding)
- Execute the model checker and find a bug
- Add a piece of TLS to fix the bug and repeat
  - Better understand the design of the protocol

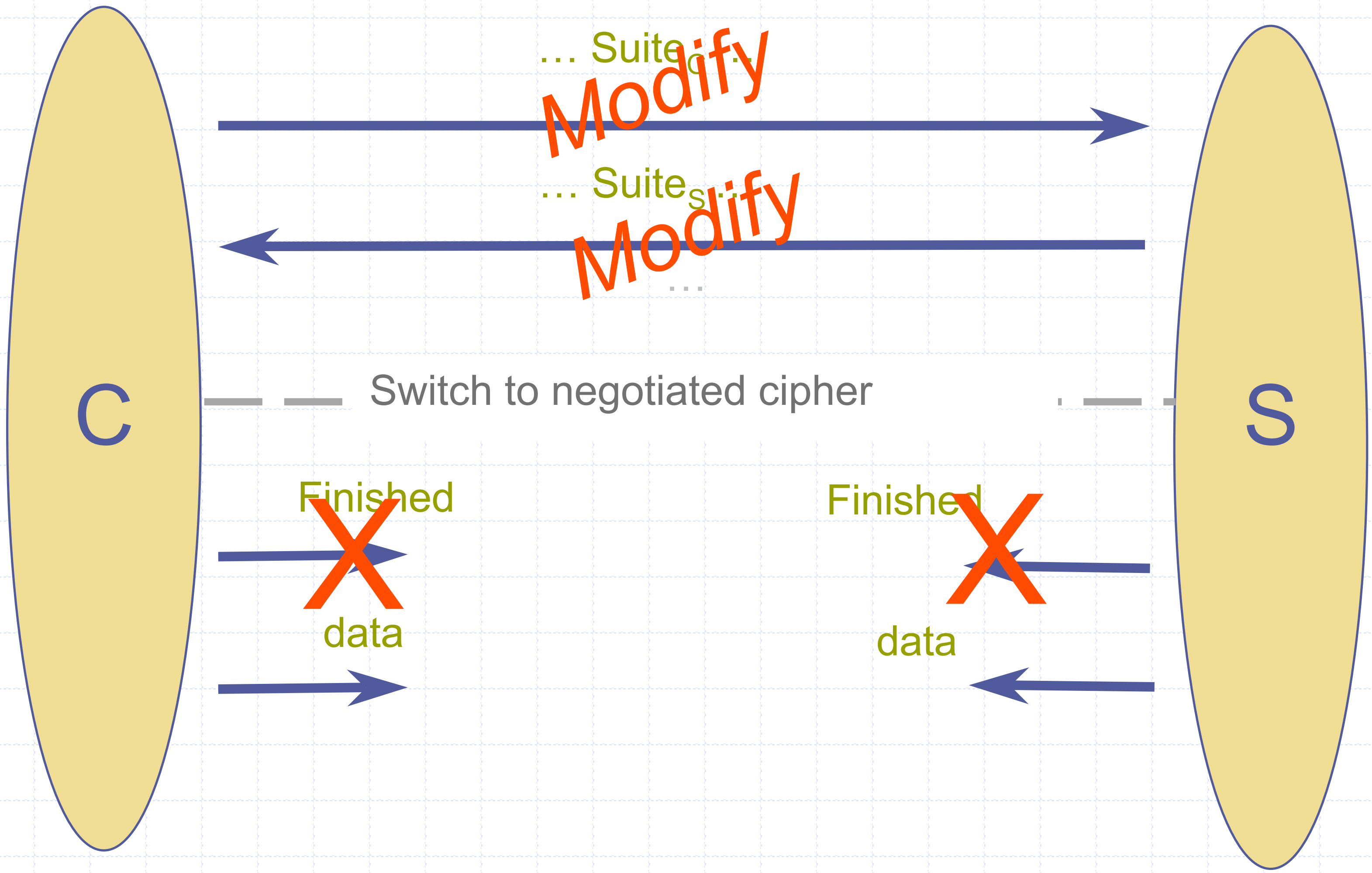
Summary of Incremental Protocols

- A = Basic protocol
- B = A + version consistency check
- D = B + certificates for both public keys
  - Authentication for client + Authentication for server
- E = D + verification (Finished) messages
  - Prevention of version and crypto suite attacks
- F = E + nonces
  - Prevention of replay attacks
- G = “Correct” subset of SSL
  - Additional crypto considerations (black art) give SSL 3.0

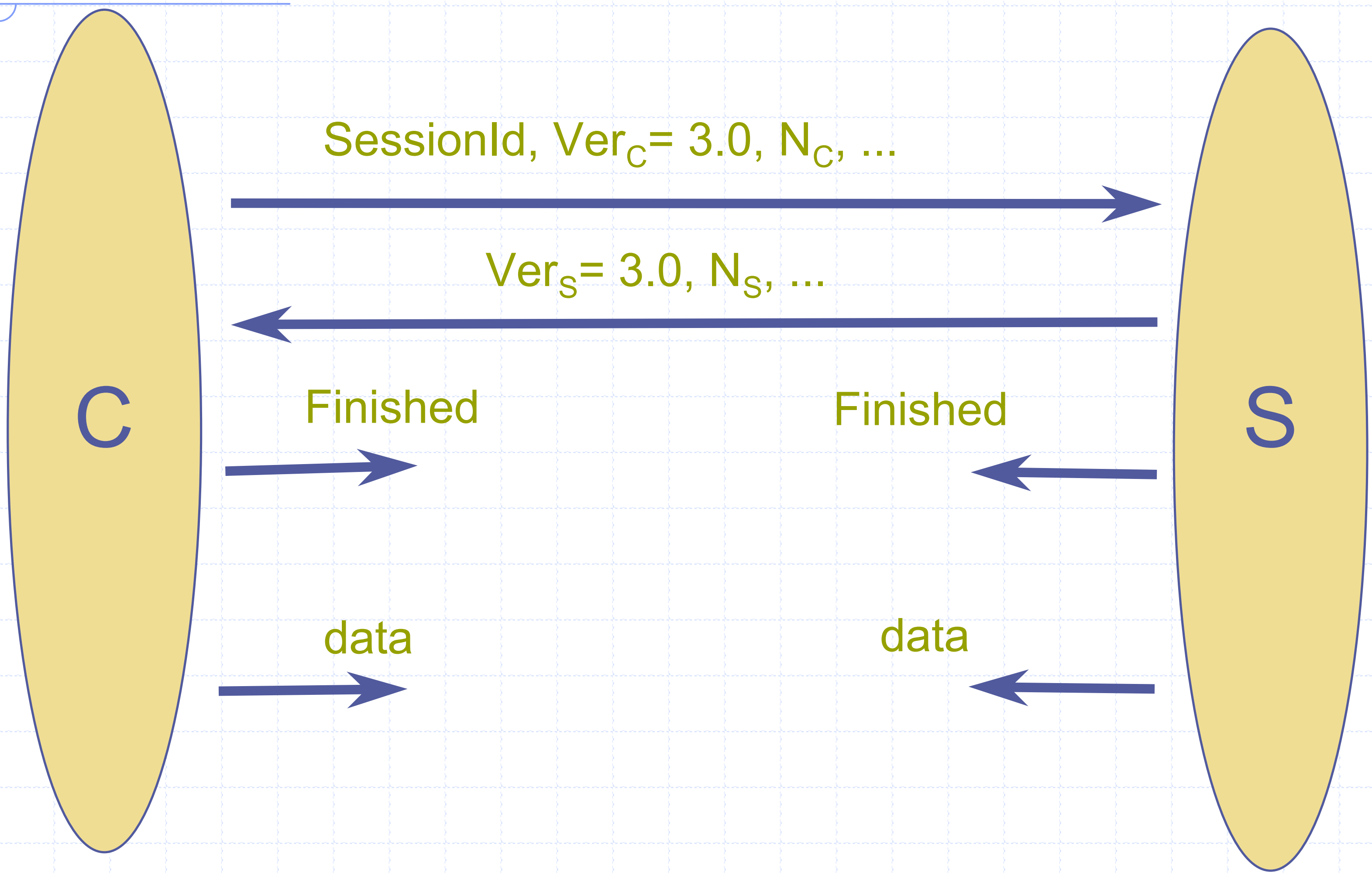
Anomaly (Protocol F)



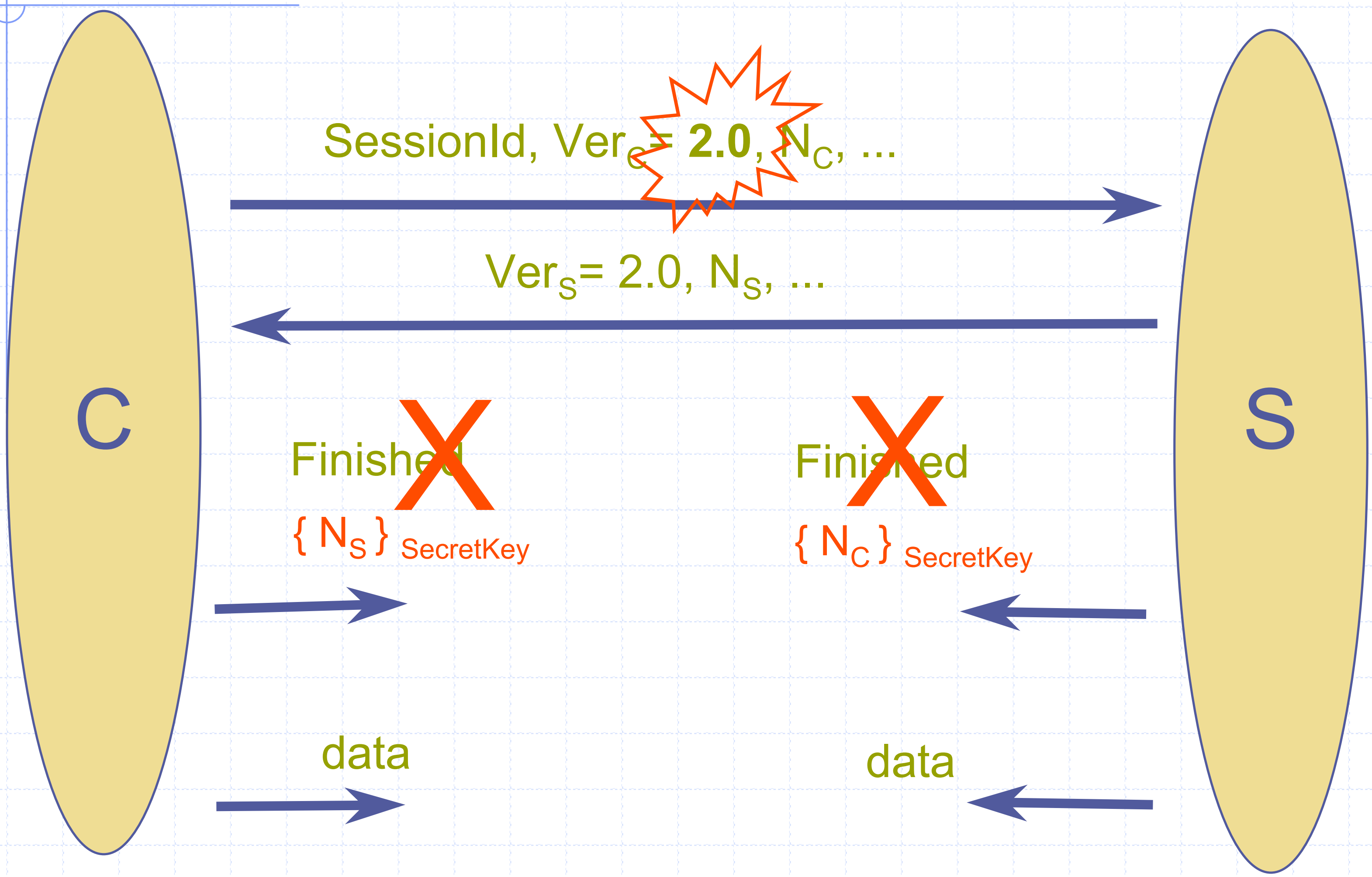
Anomaly (Protocol F)



Protocol Resumption



Version Rollback Attack



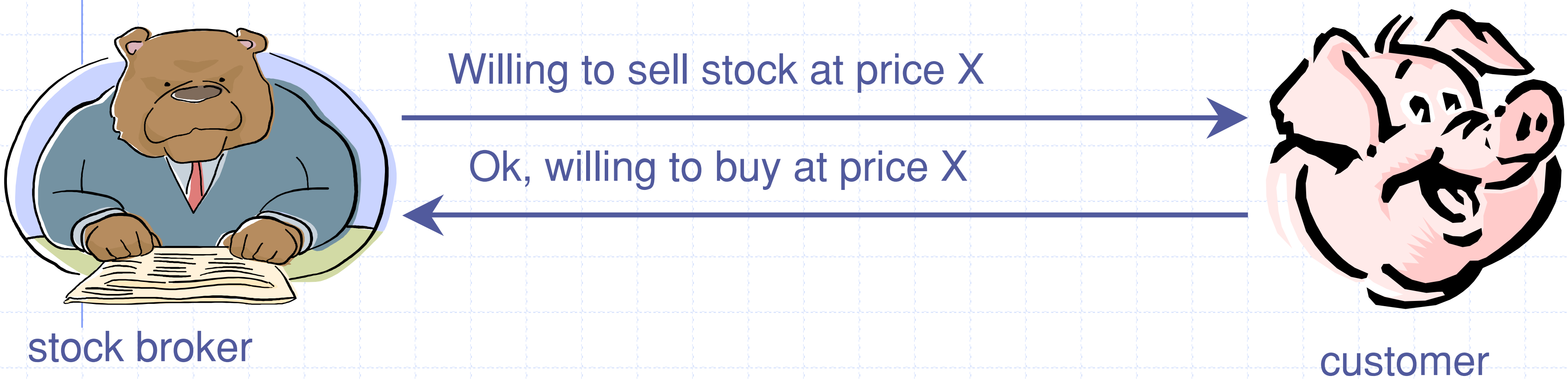
SSL 2.0 Finished messages do not include version numbers or cryptosuites

Contract Signing

Seller advertises and receives bids
Buyer may have several choices

- Both parties want to sign a contract
- Neither wants to commit first

Another example: stock trading

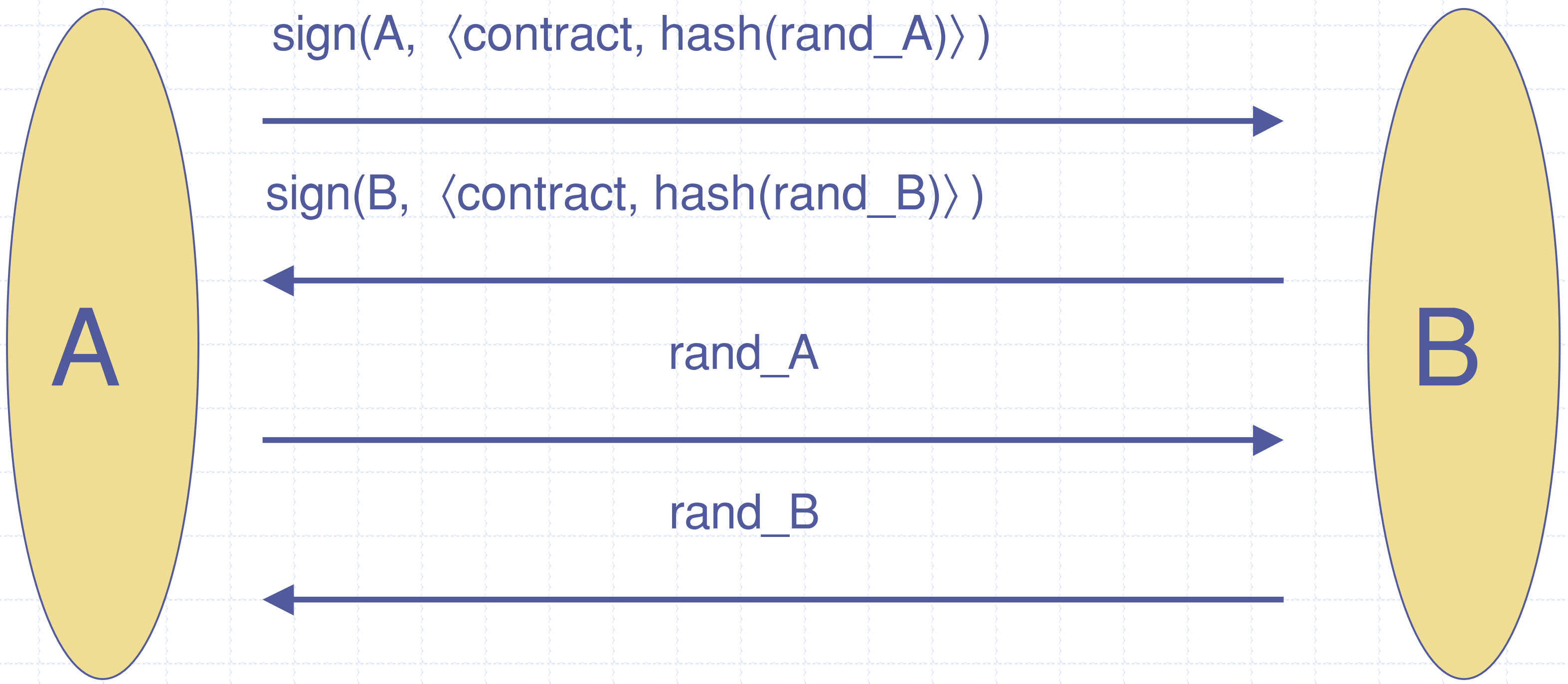


- Why signed contract?
  - Suppose market price changes
  - Buyer or seller may want proof of agreement

A general protocol outline

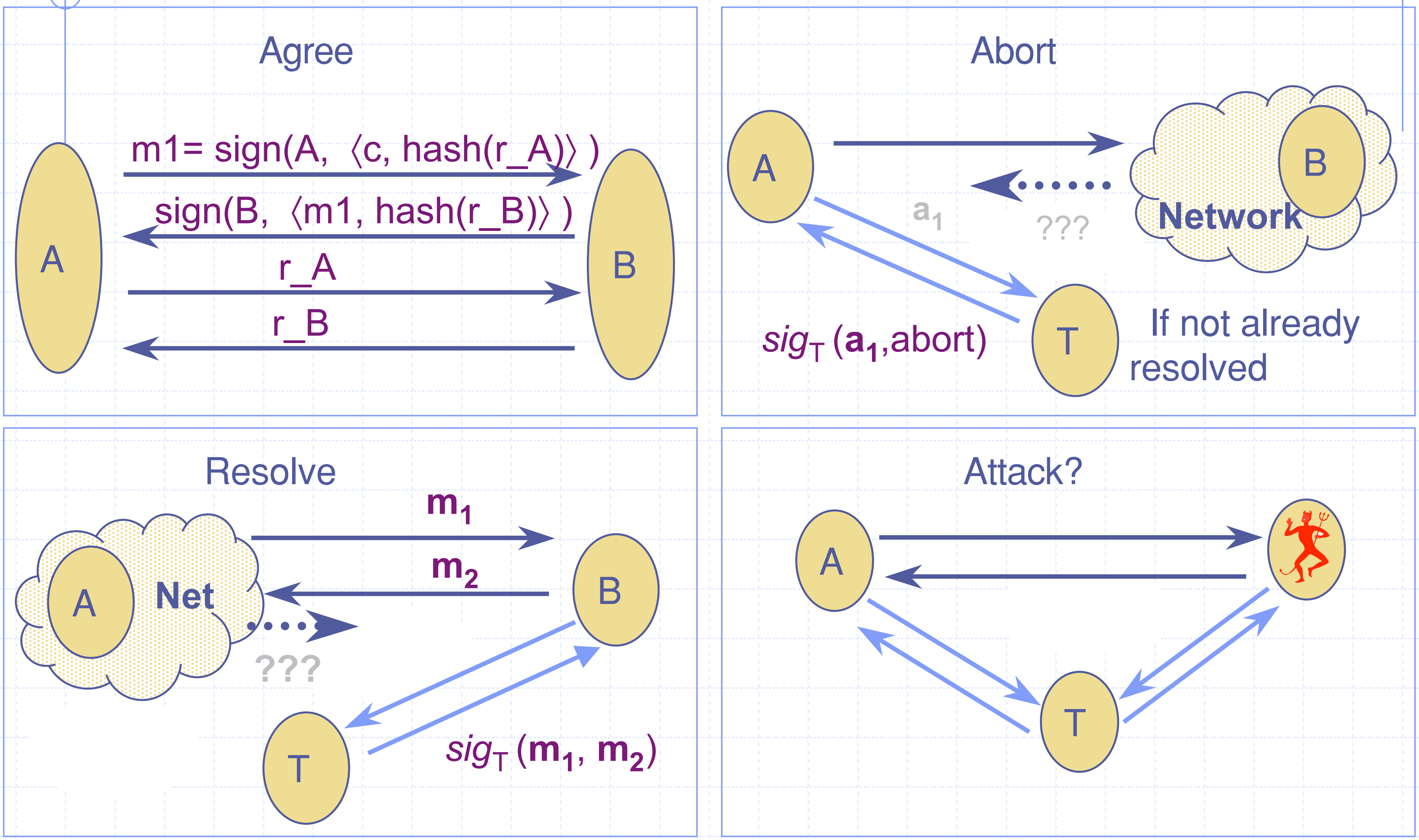
- Trusted third party can force contract
  - Third party can declare contract binding if presented with first two messages.

Refined protocol outline

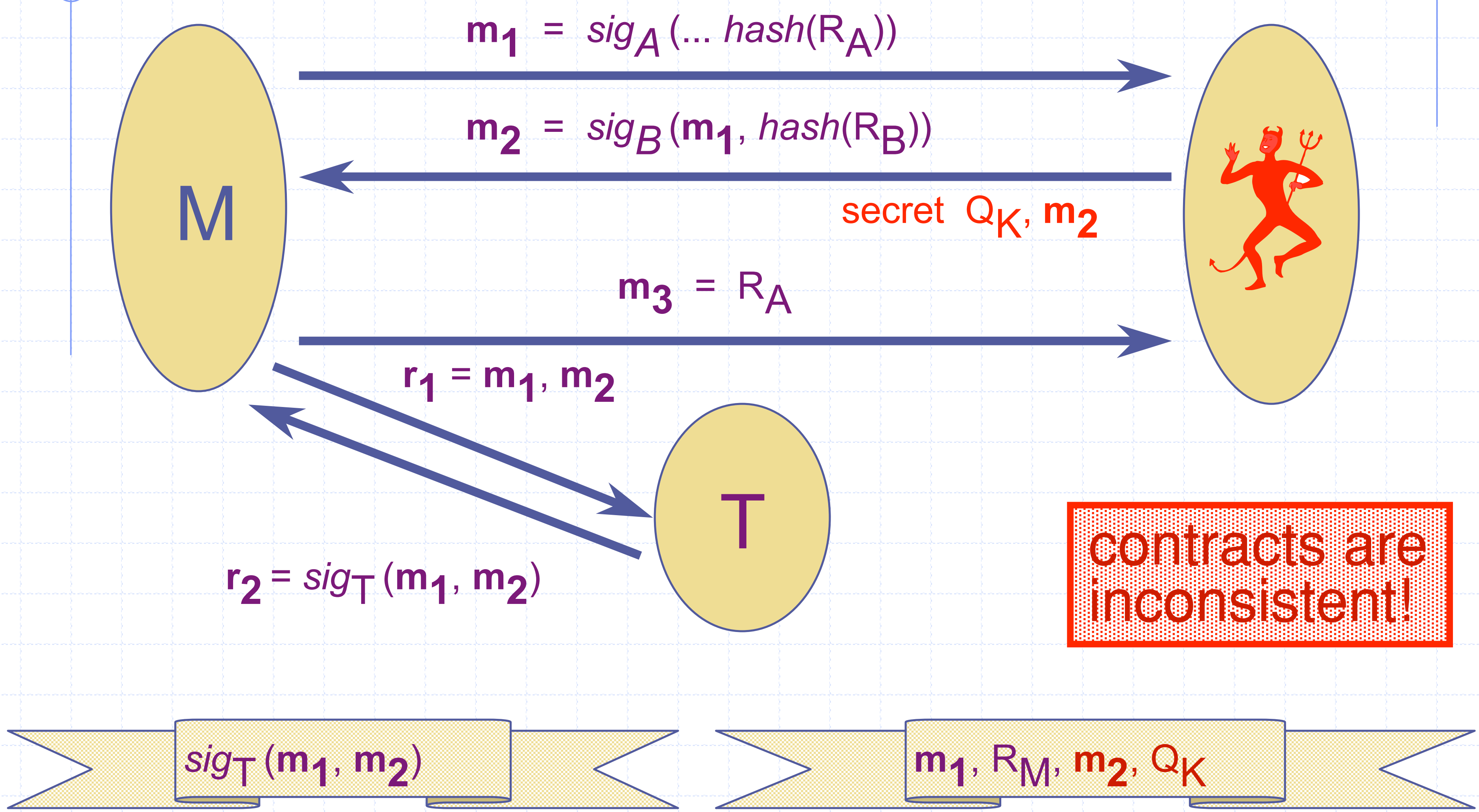


- Trusted third party can force contract
  - Third party can declare contract binding by signing first two messages.

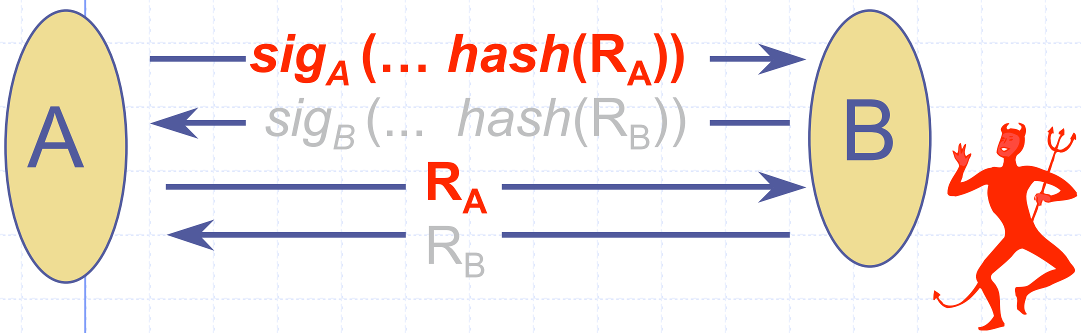
Asokan-Shoup-Waidner protocol



Contract Consistency Attack

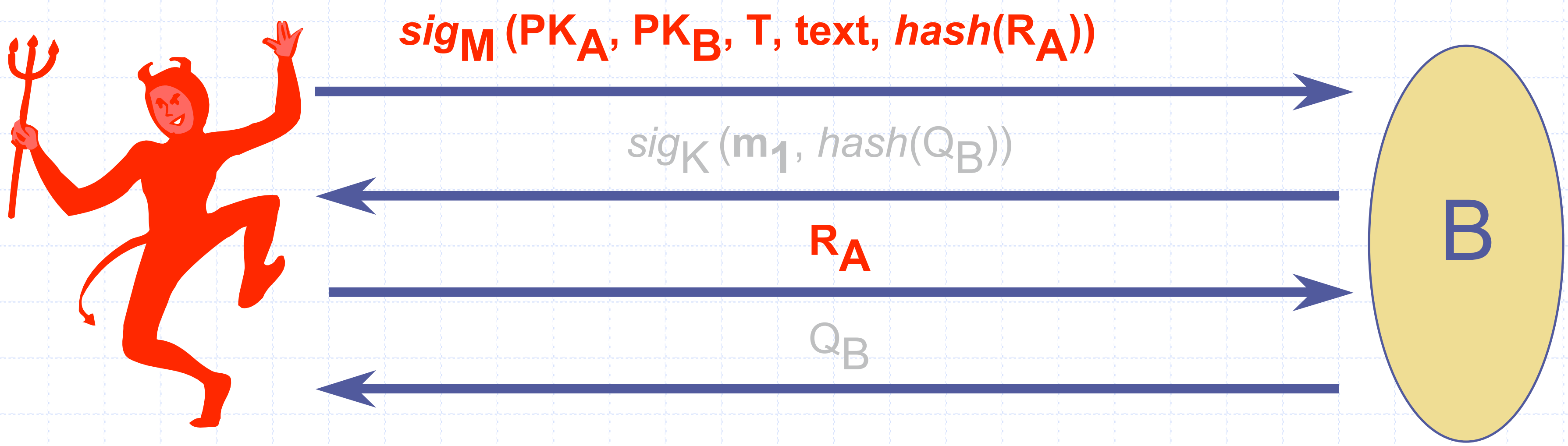


Replay Attack

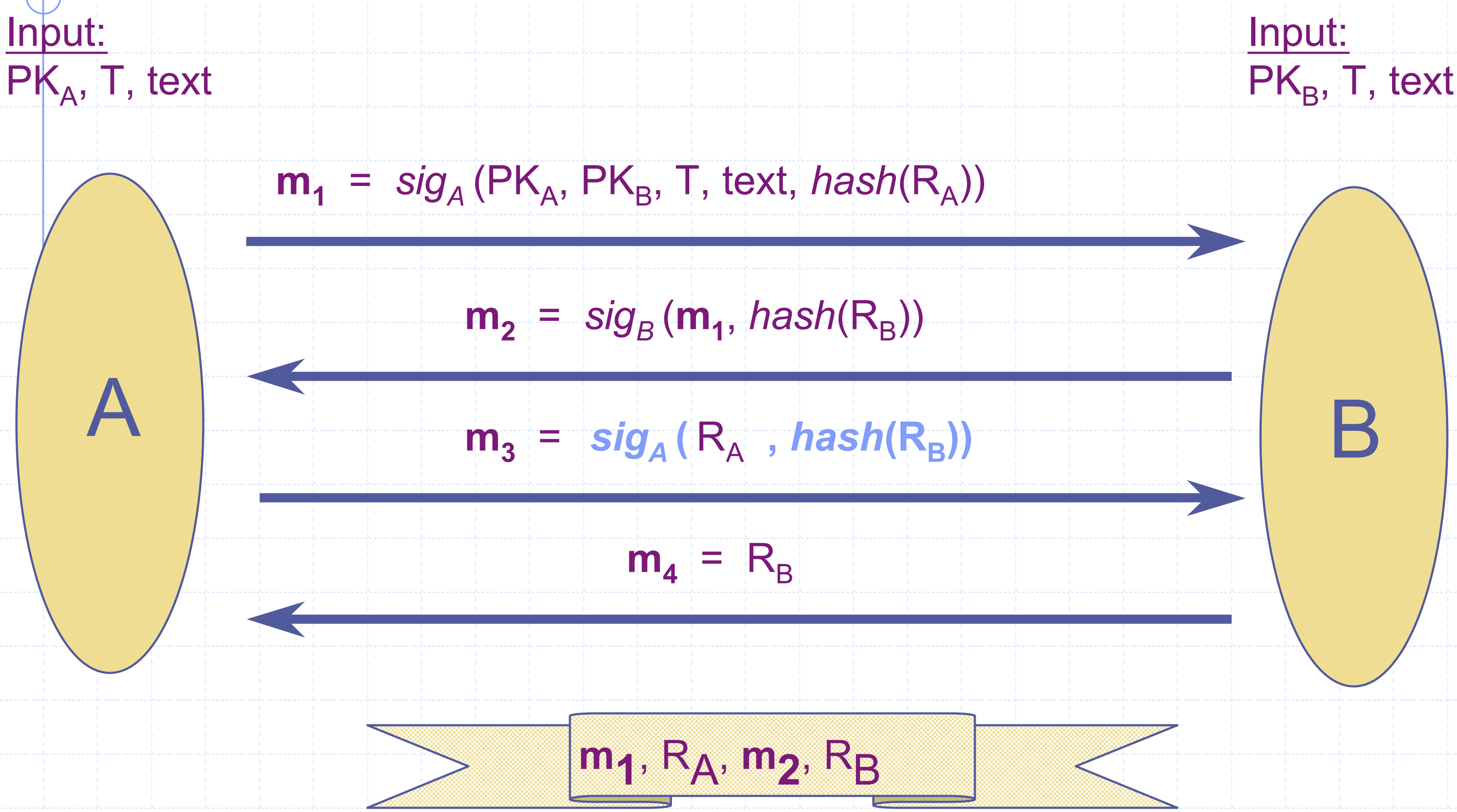


Intruder causes B to commit to old contract with A

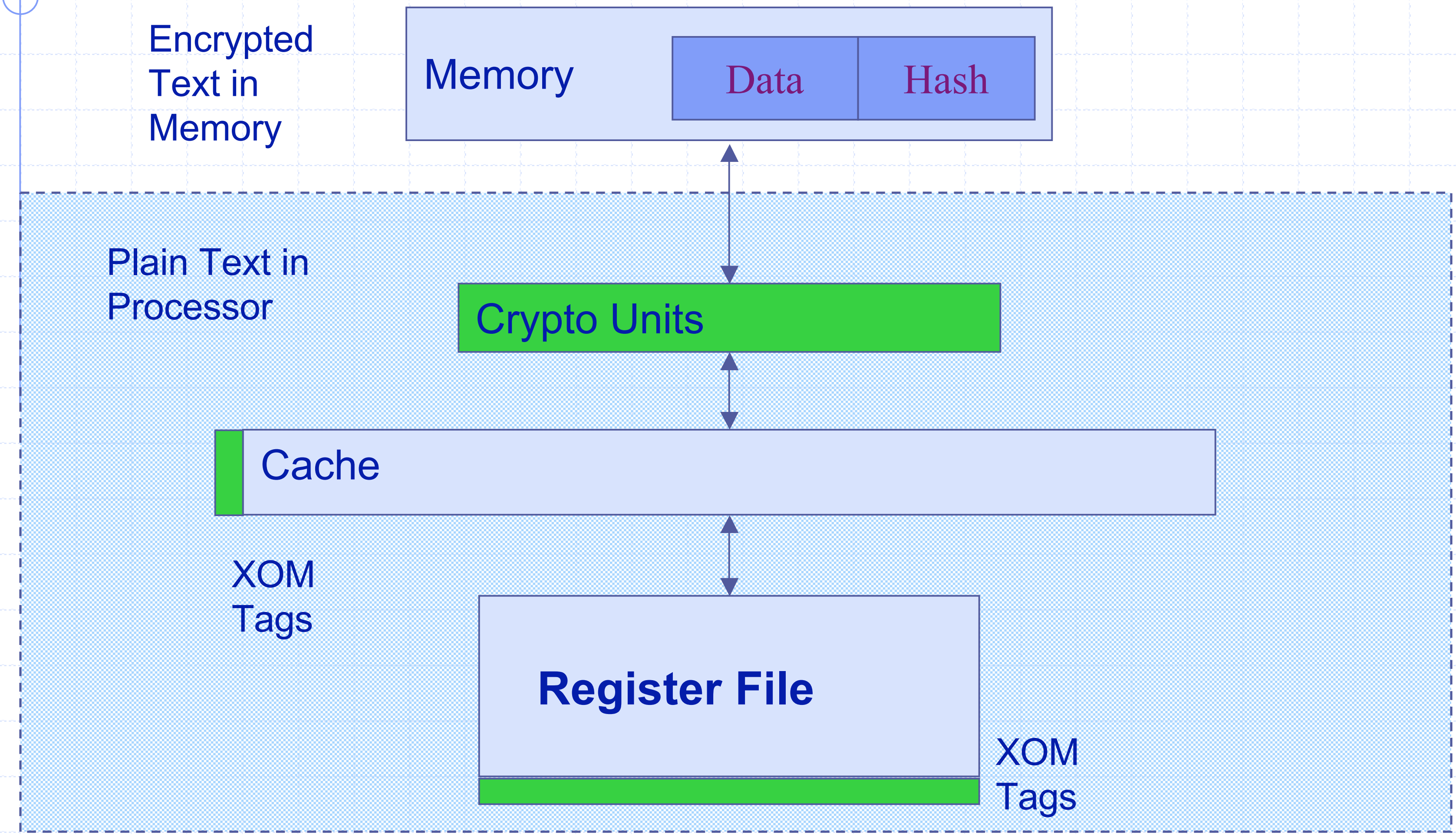
Later ...



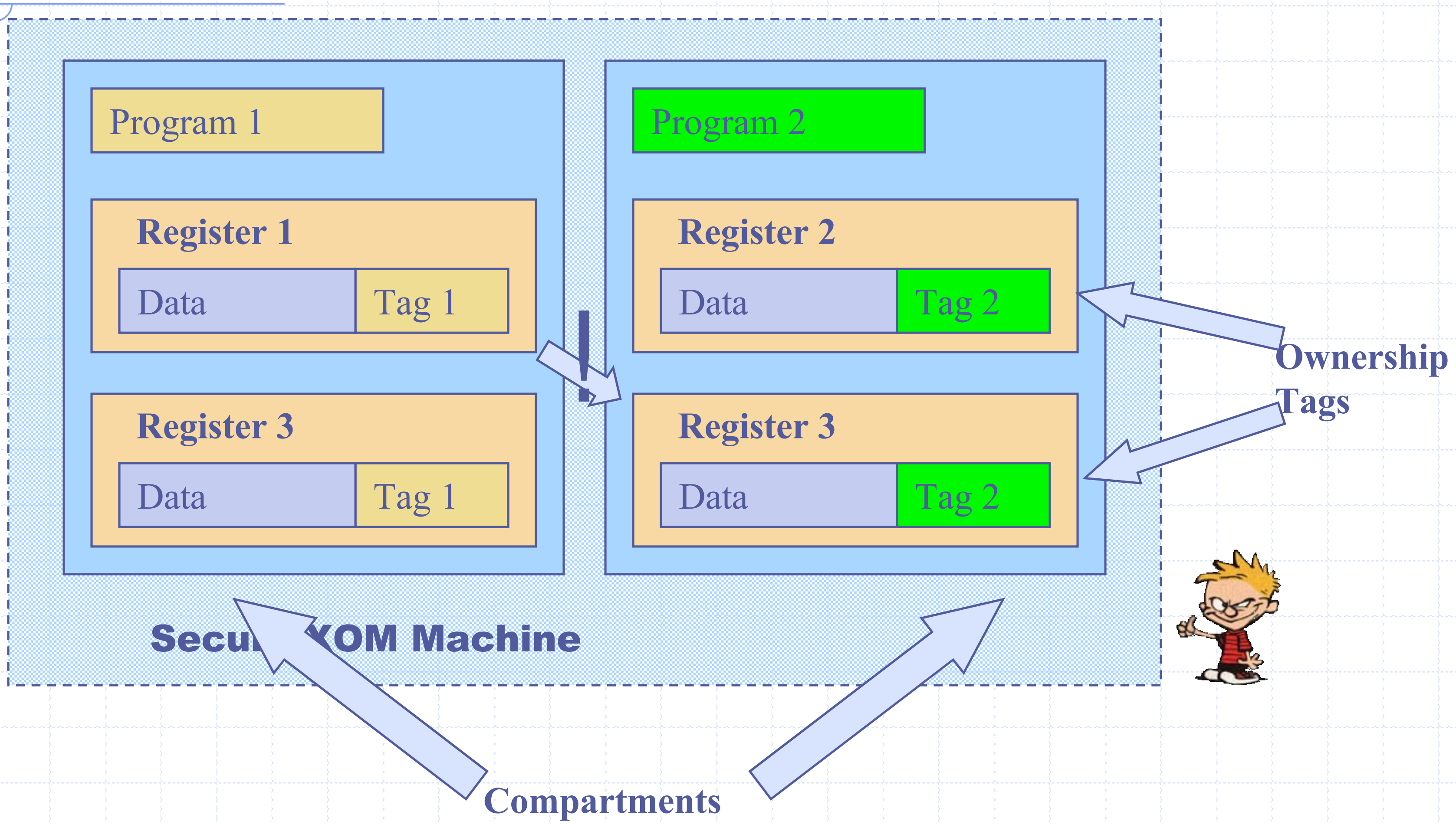
Fixing the Protocol



XOM Processor Hardware



XOM Provides Isolation

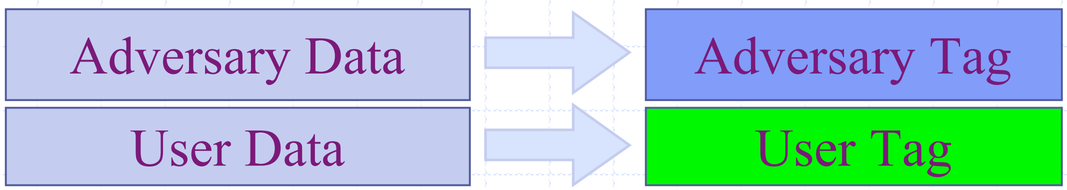


Model Checking XOM

- XOM Model is a state machine:
  - State Vector
    - A set of all things on the chip that can hold state
    - Based on the Processor Hardware
  - Next-State Functions
    - A set of state transitions that the hardware can have
    - Derived from the instructions that can be executed on the processor
  - Invariants
    - Define the correct operation of the XOM processor
    - Two Goals: Prevent observation and modification

No Observation Invariant

1. **Program data cannot be read by adversary**
   - XOM machine performs tag check on every access
   - Make sure that owner of data always matches the tag



if r_i.data is user data then

  check that: r_i.tag = user

else

  check that: r_i.tag = adversary

No Modification Invariant

2. Adversary cannot modify the program without detection

- Adversary may modify state by copying or moving user data
- Need a “ideal” correct model to check against

Adversary

XOM Model

=

Ideal Model

Program

For Memory:

if $xom.m_i.data$ = user data then

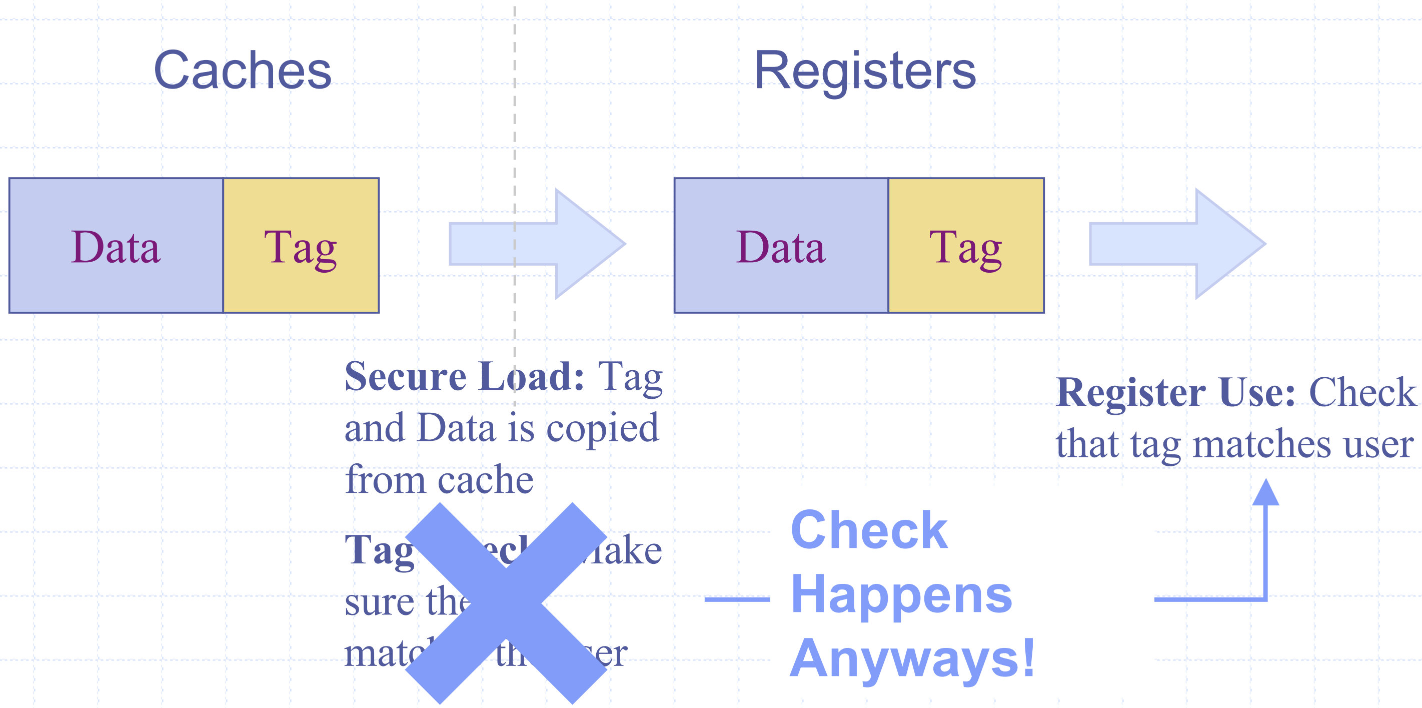
check that: $ideal.m_i.data = xom.m_i.data$

Checking for Correctness

- Model checker helped us find bugs and correct them
  - 2 old errors were found
  - 2 new errors were found and corrected
- Example:
  - Case where it’s possible to replay a memory location
  - This was due to the write to memory and hash of the memory location not being atomic

Reducing Complexity

- Fewer operations makes logic simpler
- Exhaustively remove actions from the next-state functions
  - If a removed action does not result in a violation of an invariant then the action is *extraneous*
  - Example:



Basic Pattern for Doing Your Project

- Read and understand protocol specification
  - Typically an RFC or a research paper
  - We’ll put a few on the website: take a look!
- Choose a tool
  - Murj or other tool
- Start with a simple (possibly flawed) model
  - Rational reconstruction to understand how protocol works and why
- Give careful thought to security conditions
  - This is often the most interesting part!

CS259 Term Projects - 2008

Mobile IPv6 Binding Update	*Fast Handover Key Distribution Using SEND in Mobile IPv6*	Bluetooth v2.1 + EDR Pairing Authentication Protocol
BitTorrent	OpenID 2.0	Handoffs in 802.16g
HIPAA online compliance auditor	*TCG Remote Attestation*	*Direct Anonymous Attestation (DAA) Protocol*
Pynchon Gate Protocol:		

http://www.stanford.edu/class/cs259/

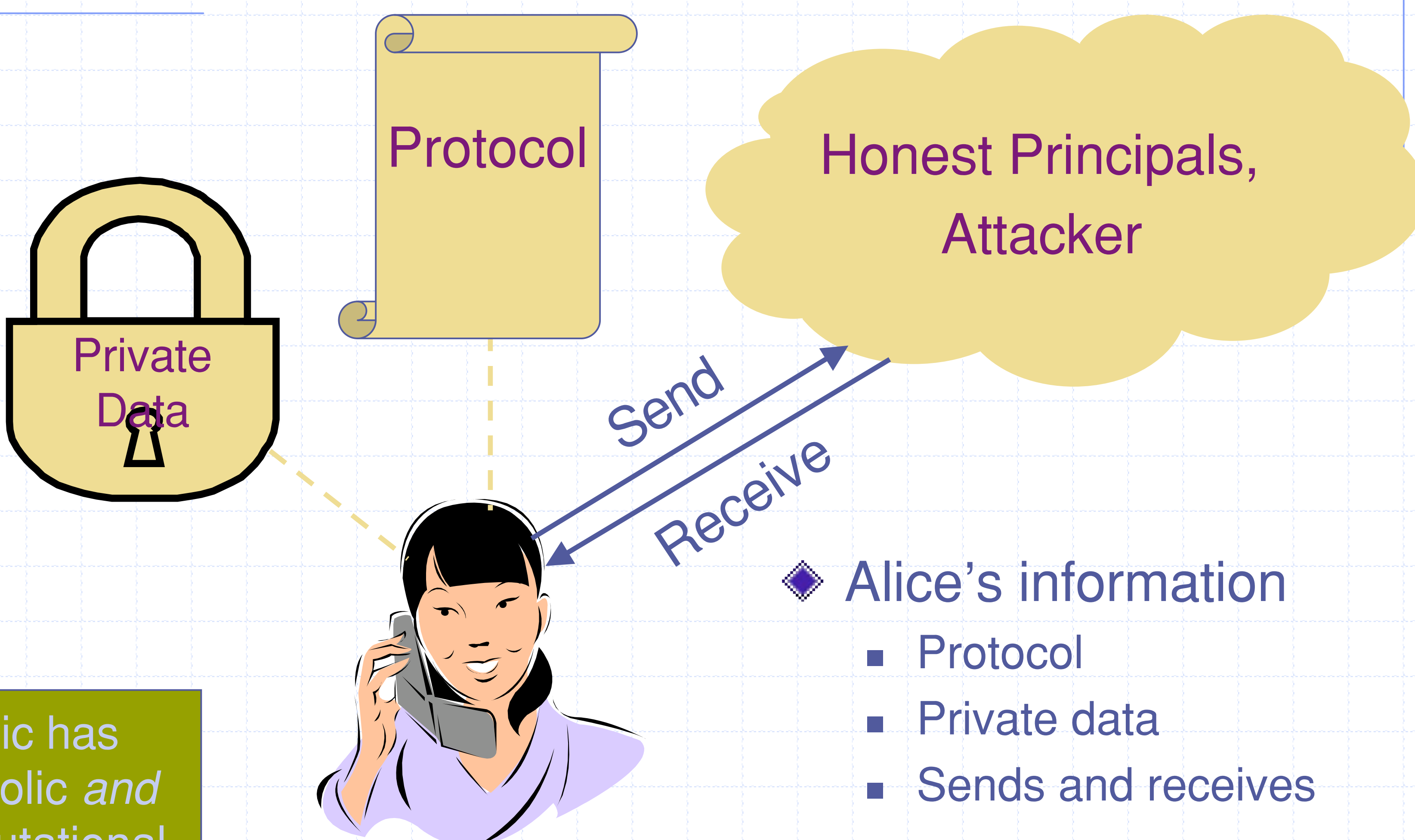
CS259 Term Projects - 2006

Security Analysis of OTRv2	*Formalization of HIPAA*	Security analysis of SIP
Onion Routing	*Analysis of ZRTP*	MOBIKE - IKEv2 Mobility and Multihoming Protocol
802.16e Multicast-Broadcast Key Distribution Protocols	*Short-Password Key Exchange Protocol*	*Analysis of the IEEE 802.16e 3-way handshake*
Analysis of Octopus and Related Protocols		

CS259 Term Projects - 2004

iKP protocol family	*Electronic voting*	*XML Security*
IEEE 802.11i wireless handshake protocol	*Onion Routing*	*Electronic Voting*
Secure Ad-Hoc Distance Vector Routing	*An Anonymous Fair Exchange E-commerce Protocol*	*Key Infrastructure*
Secure Internet Live Conferencing	*Windows file-sharing protocols*	

Protocol composition logic



- Alice's information
  - Protocol
  - Private data
  - Sends and receives

Logic has symbolic *and* computational semantics

Goals of PCL

- PCL is an evolving research framework for approaching this basic question:
  - Is it possible to prove security properties of current practical protocols using compositional, direct reasoning that does not mention the actions of the attacker?

- Example
  - If the client and server exchange hashes of the messages they have seen, encrypted under a shared key, then they must have matching conversations

Can such ordinary conversational arguments be proved sound and used for practical examples?

Logic: Background

- Logic
  - Syntax — Formulas
    - p, $p \vee q$, $(p \vee q)$, $p \Rightarrow q$
  - Semantics — Truth
    - Model, $M = \{p = \text{true}, q = \text{false}\}$

      $M \models p \vee q$
- Proof System
  - Axioms and proof rules — Provability

    ω $p \Rightarrow (q \Rightarrow p)$

    $$\frac{p \quad p \Rightarrow q}{q}$$
  - Soundness Theorem
    - Provability implies truth
    - Axioms and proof rules hold in all "relevant" models

Protocol logic: Actions

send t;	send a term t
receive x;	receive a term into variable x
new n;	generate nonce n

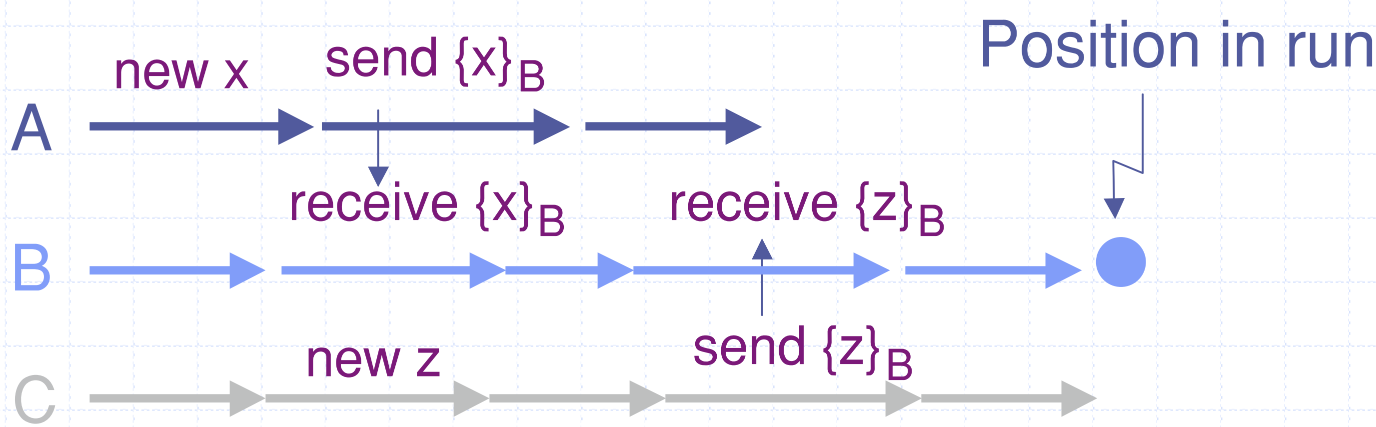
- A program is a sequence of actions

InitCR(A, X) = [
new m;
send A, X, {m, A};
receive X, A, {x, sig_X{"r", m, x, A}};
send A, X, sig_A{"i", m, x, X}};
$]_A$

RespCR(B) = [
receive Y, B, {y, Y};
new n;
send B, Y, {n, sig_B{"r", y, n, Y}};
receive Y, B, sig_Y{"i", y, n, B}};
$]_B$

Execution Model

- Initial Configuration, IC
  - Set of principals and keys
  - Assignment of ≥ 1 role to each principal
- Run
  - Interleaving of actions of honest principals and attacker starting from IC



Formulas true at a position in run

- Action formulas

  a ::= Send(P,t) | Receive (P,t) | New(P,t)

  | Decrypt (P,t) | Verify (P,t)

- Formulas

  φ ::= a | Has(P,t) | Fresh(P,t) | Honest(N)

  | Contains(t_1, t_2) | φ | $\varphi_1 \wedge \varphi_2$ | $\exists x\ \varphi$

  | a < a

- Modal formula

  φ [*actions*]$_P$ φ

- Example

  Has(X, secret)   (X = A $\vee$ X = B)

Specifies secrecy

Challenge-Response Property

Specifying authentication for Responder

CR |= true [RespCR(A)]$_B$ Honest(A) (

Send(A, {A,B,m}) < Receive(B, {A,B,m}) $\wedge$

Receive(B, {A,B,m}) < Send(B, {B,A,{n, sig_B {"r",m, n, A}}}) $\wedge$

Send(B, {B,A,{n, sig_B {"r",m, n, A}}}) < Receive(A, {B,A,{n, sig_B {"r",m, n, A}}}) $\wedge$

Receive(A, {B,A,{n, sig_B {"r",m, n, A}}}) < Send(A, {A,B,{sig_A{"i",m,n,B}}}) $\wedge$

Send(A, {A,B,{sig_A{"i",m,n,B}}} < Receive(B, {A,B,{sig_A{"i",m,n,B}}}))

)

Authentication as "matching conversations" [Bellare-Rogaway93]

Proof System

- Goal: Formally prove security properties
- Axioms
  - Simple formulas about actions, etc.
- Inference rules
  - Proof steps
- Theorem
  - Formula obtained from axioms by application of inference rules

Sample axioms

- Actions

  true [send m $]_P$ Send(P,m)

- Public key encryption

  Honest(X) $\wedge$ Decrypt(Y, enc_X\{m\}) X=Y

- Signature

  Honest(X) $\wedge$ Verify(Y, sig_X\{m\})

  Sign(X, sig_X\{m\})

Correctness of CR – step 1

InitCR(A, X) = [
new m;
send A, X, {m, A};
receive X, A, {x, sig$_X${"r", m, x, A}};
send A, X, sig$_A${"i", m, x, X}};
]$_A$

RespCR(B) = [
receive Y, B, {y, Y};
new n;
send B, Y, {n, sig$_B${"r", y, n, Y}};
receive Y, B, sig$_Y${"i", y, n, B}};
]$_B$

1. B reasons about his own action

CR |- true [RespCR(B)]$_B$ Verify(B, sig$_A$ {"i", m, n, A})

2. Use signature axiom

CR |- true [RespCR(B)]$_B$ Sign(A, sig$_A${"i", m, n, A})

Proving Invariants

- We want to prove
  - $\Gamma \int$ Honest(X) $\rightarrow$ φ,

    where $\varphi \int$

    (Sign(X, sig_X("i", m, n, Y) $\rightarrow$ Receive(Y, n, sig_Y("r", m, n, X)))

- "φ holds at all pausing states of all traces"
  - Since the fragment of honest party action between pausing states is a protocol segment, the propagation of φ looks like:
  - φ --- actions of A --- φ ---- actions of B --- φ --- attacker actions -- φ ---- actions of B --- φ -- …

Proving Invariants (2)

- Rule for establishing Γ:
  - Prove φ holds when threads have started
  - Prove, for all *protocol segments*, if φ held at the beginning, it holds at the end

Correctness of CR – step 2

- So far
  - CR |- true [RespCR(B) $]_B$ Sign(A, sig_A{"i", m, n, A})
- Use invariant Γ to prove:
  - CR |- true [RespCR(B) $]_B$ Receive(A, n, sigB{"r", m, n, A})
- Reason from B's point of view to prove:
  - CR |- true [RespCR(B) $]_B$ FirstSend(B, n, (n, sigB{"r", m, n, A}))
- Apply Nonce freshness axiom to prove:
  - CR |- true [RespCR(B) $]_B$ Receive(A, (n, sigB{"r", m, n, A})) < Send(B, sigB{"r", m, n, A})
- A few similar steps leads to the full proof!

Sample PCL studies

- Wireless 802.11i
  - Model checking to find errors, improve
  - PCL proof of correctness, including TLS
- Kerberos
  - Including variants “PK-Init” and “DH-init”
- Extensible Authentication Protocol (EAP)
  - Model check to find errors, improve
  - PCL proof of correctness, identify subtleties
- Mesh Security Architecture (IEEE 802.11s)
  - *Motorola group* added some axioms, found problems, identified invariants, proved correctness

Composing protocols

$$\mathrm{DH} \blacktriangleright \underbrace{\mathrm{Honest}(X) \quad \ldots}_{\Gamma} \qquad \mathrm{CR} \blacktriangleright \underbrace{\mathrm{Honest}(X) \quad \ldots}_{\Gamma'}$$

Γ |- Secrecy

Γ' |- Authentication

$\Gamma \cup \Gamma'$ |- Secrecy

$\Gamma \cup \Gamma'$ |- Authentication

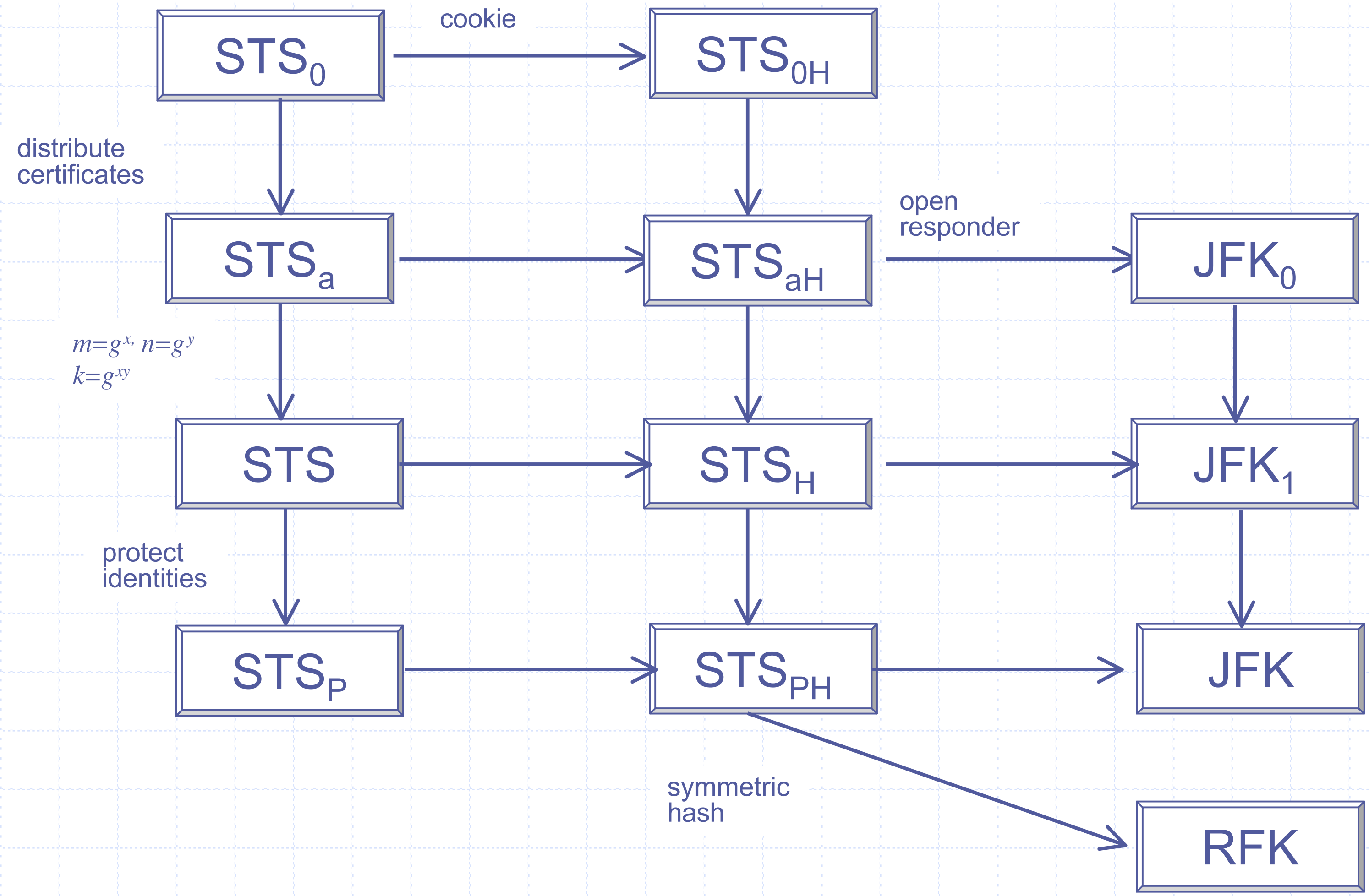
$\Gamma \cup \Gamma'$ |- Secrecy $\wedge$ Authentication [additive]

DH CR $\blacktriangleright$ $\Gamma \cup \Gamma'$ [nondestructive]

||

ISO $\blacktriangleright$ Secrecy $\wedge$ Authentication

STS family



STS_0
cookie
STS_{0H}
distribute
certificates
STS_a
STS_{aH}
open
responder
JFK_0
$m=g^x, n=g^y$
$k=g^{xy}$
STS
STS_H
JFK_1
protect
identities
STS_P
STS_{PH}
JFK
symmetric
hash
RFK

PCL → Computational PCL

PCL

Syntax

Proof System

Computational PCL

Syntax ± Δ

Proof System ± Δ

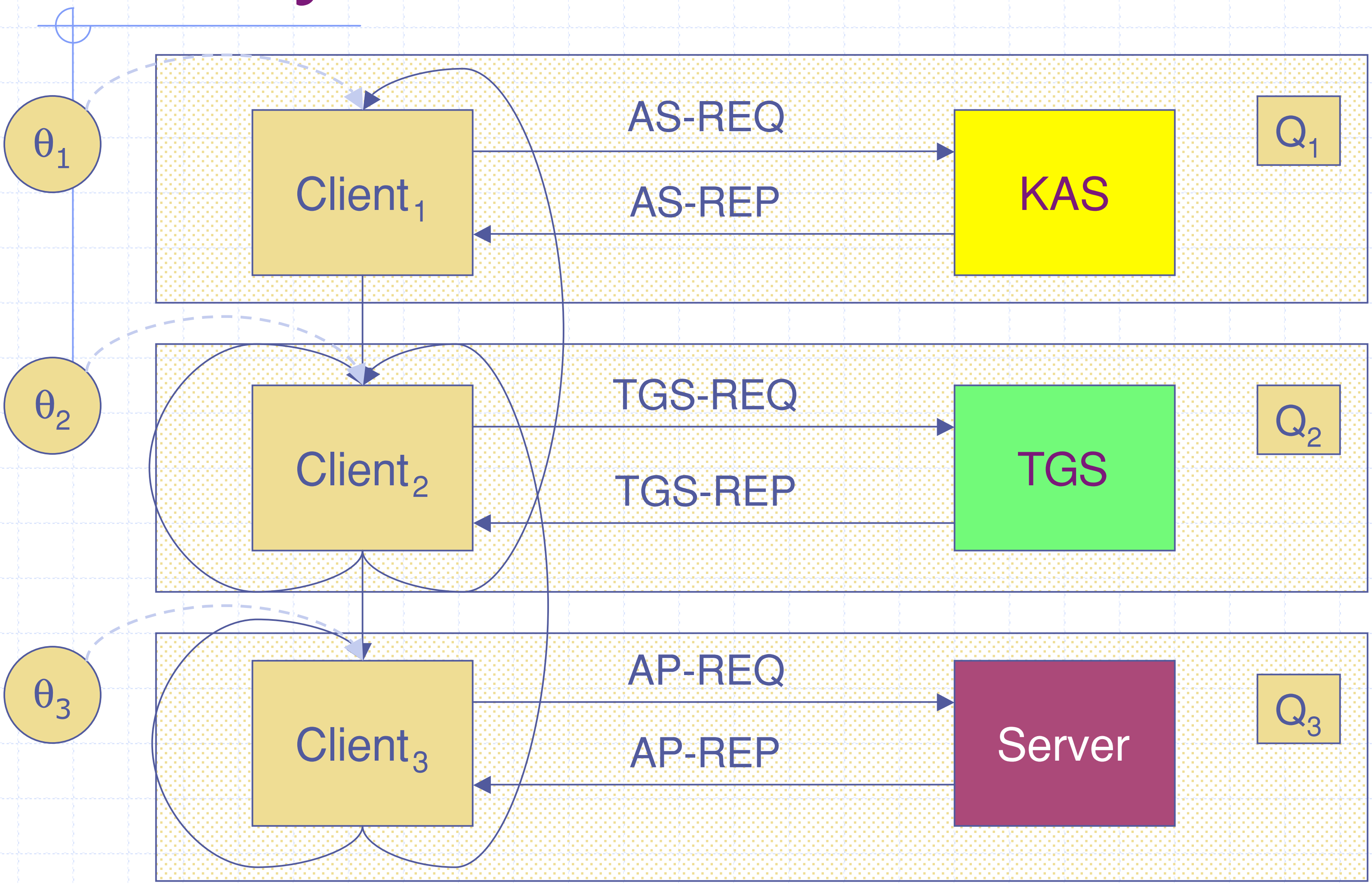
Symbolic model

Semantics

Complexity-theoretic model

Semantics

Analysis of Kerberos



CPCL analysis of Kerberos V5

- Kerberos has a staged architecture
  - 1: generate nonce and send it encrypted
  - 2: use as key to encrypt another nonce
  - 3: use 2nd nonce to encrypt other msgs
- Our proof shows “GoodKey”-ness of both the nonces
- Authentication properties are proved assuming that the encryption scheme provides ciphertext integrity

Foundational results

- Computational PCL
  - Symbolic logic for proving security properties of network protocols using public-key encryption
- Soundness Theorem:
  - If a property is provable in CPCL, then property holds in computational model with overwhelming asymptotic probability.
- Benefits
  - Symbolic proofs about computational model
  - Computational reasoning in soundness proof (only!)
  - Different axioms rely on different crypto assumptions

Challenges for computational reasoning

- More complicated adversary
  - Actions of computational adversary do not have a simple inductive characterization
- More complicated messages
  - Computational messages are arbitrary sequences of bits, without an inductively defined syntactic structure
- Different scheduler
  - Simpler “non-preemptive” scheduling is typically used in computational models (change symbolic model for equiv)
- Power of induction ?
  - Indistinguishability, other non-trace-based properties appear unsuitable as inductive hypotheses
  - Solution: prove trace property inductively and derive secrecy

Automation

- Prolog-based method for checking sufficient conditions for provability of invariants

Conclusion

- Practical protocols may contain errors
  - Tools find bugs, reveal req., guarantees
- Variety of tools
  - Model checking can find errors
  - Proof method can show correctness

    for specific model of execution and attack
- Closing gap between logic and crypto
  - Symbolic reasoning sound for statements about probability, complexity
  - Does not require strong crypto assumptions

Credits

- Collaborators
  - M. Backes, A. Datta, A. Derek, N. Durgin, C. He, R. Kuesters, D. Pavlovic, A. Ramanathan, A. Roy, A. Scedrov, V. Shmatikov, M. Sundararajan, V. Teague, M. Turuani, B. Warinschi, …
- More information
  - Protocol model-checking web page
    - http://crypto.stanford.edu/protocols/mc.html
  - Protocol Composition Logic
    - http://crypto.stanford.edu/protocols/

Science is a social process