

APPLYING FORMAL METHODS TO CRYPTOGRAPHIC PROTOCOL ANALYSIS

Catherine Meadows

Code 5543

Center for High Assurance Computer Systems

US Naval Research Laboratory

Washington, DC 20375

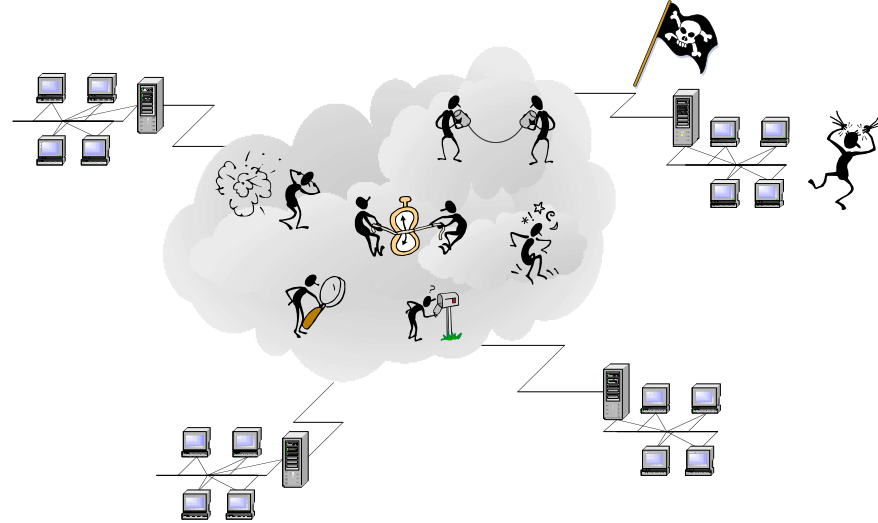
meadows@itd.nrl.navy.mil

<http://chacs.nrl.navy.mil>

OVERVIEW OF TALK

- **Background: What are cryptographic protocols, and why should we be interested in them?**
- **Short history of application of formal methods to cryptographic protocol analysis**
- **Discussion of what I see as emerging issues**
 - **Will concentrate on issues raised by applications rather than the theoretical issues**

WHAT IS A CRYPTOGRAPHIC PROTOCOL?

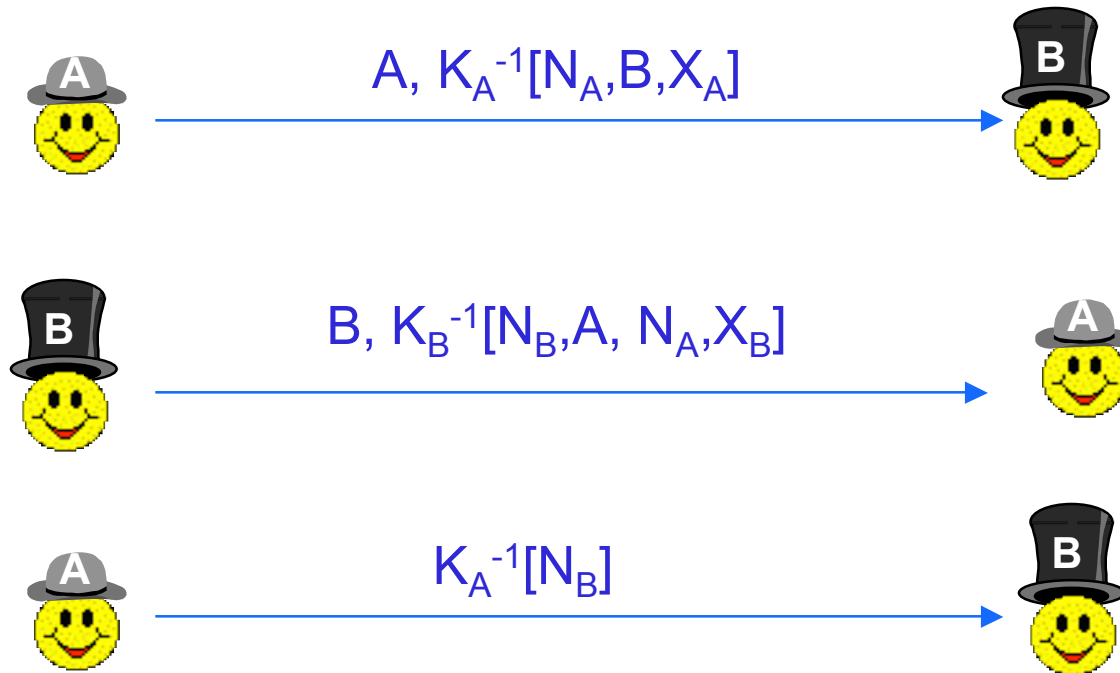


- Communication protocol that uses encryption to:
 - **Distribute keys**
 - **Authenticate principals**
 - **Process transactions securely**
- Must operate in hostile environment in which traffic may be intercepted, altered or destroyed

EXAMPLE: CCITT DRAFT STANDARD X.509 (1987)

- A and B want to verify origin and recency of messages
- Protocol uses public key crypto
 - A and B possess public keys K_A and K_B , private keys K_A^{-1} and K_B^{-1}
 - Anyone can send $K_A[X]$ to A, only A can read X
 - If A sends $K_A^{-1}[X]$, anyone can compute $K_A[K_A^{-1}[X]] = X$ and verify X came from A
- A and B both have the capacity to generate nonces
 - If B receives $K_A^{-1}[X, N]$, where N is a nonce previously sent by B, B knows A sent message after B sent nonce

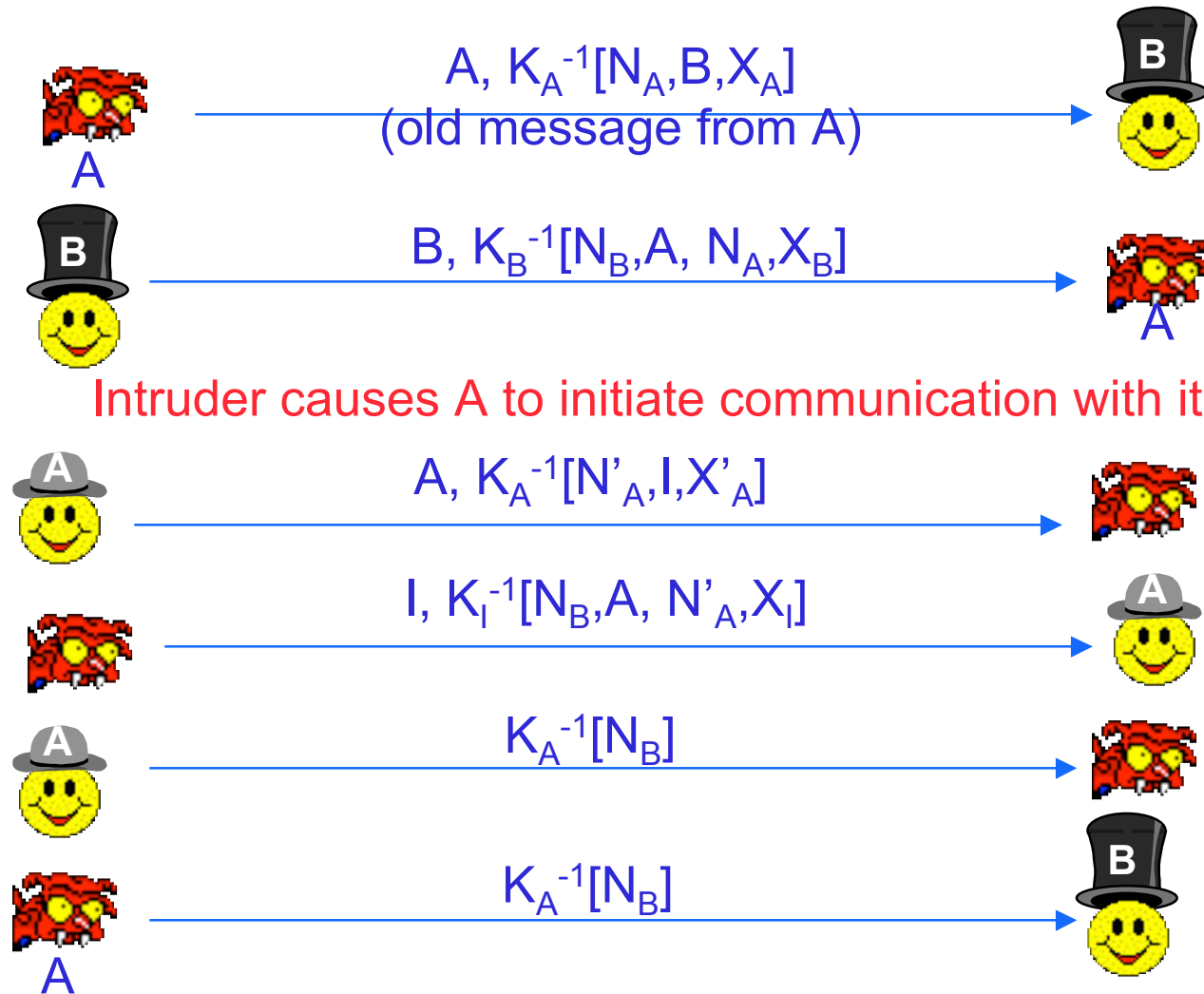
THE PROTOCOL (simplified)



Third message appears to be linked to second by N_B
Second message appears to be linked to first by N_A

Is this enough?

NO! (Burrows, Abadi, Needham, 1989)



A HIERARCHY OF CRYPTO PROTOCOL MODELS

Logics of knowledge and belief

- No concrete model of intruder
- Describes what can be deduced by honest principals from a successful protocol run

“Dolev-Yao” model

- Concrete model of adversary
 - Adversary restricted to (possibly arbitrary set of) fixed set of operations
 - Adversary able to intercept traffic, etc.

Complexity-theoretic models

- Reduce breaking the protocol to solving a computationally hard problem
 - May or may not be able to intercept traffic, etc.
- Adversary restricted to computations done in polynomial time
- Random oracle model special case - appears to integrate well with Dolev-Yao model

Information-theoretic models

- Adversary assumed to have infinite computational power
 - May or may not be able to intercept traffic, etc.

FIRST MENTION OF FORMAL METHODS FOR CRYPTO PROTOCOL ANALYSIS - 1978

- **Needham & Schroeder -- “Using encryption for authentication in large networks of computers,” CACM, 1978**
 - **Early protocols for key distribution and authentication**
 - **Mention as an aside that formal methods could be useful for assuring correctness**

DOLEV-YAO (IEEE Trans Info Theory, 1983)

DOLEV-EVEN & KARP (Info & Control, 1982)

- Assume that cryptoalgorithms obey a certain set of algebraic identities
- Assume an intruder who can
 - read all traffic
 - modify, delete and create traffic
 - perform cryptographic operations available to legitimate users of the system
 - is in league with a subset of “corrupt” nodes
- Assume an arbitrary number of nodes
- Assume an arbitrary number of protocol executions
- Assume that protocol executions may be interleaved
- Allows us to consider the protocol as an algebraic system operated by the intruder

With some modifications, this is the most commonly used model today for formal methods analyses of crypto protocols

PING-PONG PROTOCOLS

- **Small number of operations**
 - **Public key encryption and decryption**
 - **Namestamps**
- **Action of a party on receiving a message is to apply some sequence of operations to it and send it back out**
- **Message sent back and forth like a ping-pong ball, hence the name**
- **Dolev-Yao, Dolev, Even & Karp, and later others, found algorithms for determining whether a protocol would reveal secrets to an intruder**
 - **Found to be related to problem of finding intersection of two formal languages**
 - **Freshness issues (e.g. replay) were not considered**

WHAT HAVE WE LEARNED HERE?

- Possible to abstract away from original protocol model to one that treats protocol as matching operated by intruder
- Possible to reason about machine in exhaustive way so that even nonintuitive attacks ruled out

WHAT'S MISSING FROM THE MODEL

- Other operators, such as
 - Concatenation
 - single-key encryption
 - Digital signatures
- Other data types, such as
 - Nonces
 - timestamps
- Requirements other than secrecy, such as
 - Authentication
 - Replay prevention
- Other intruder capabilities, such as
 - Compromise of keys, etc.
- Unfortunately, problem becomes undecideable quickly

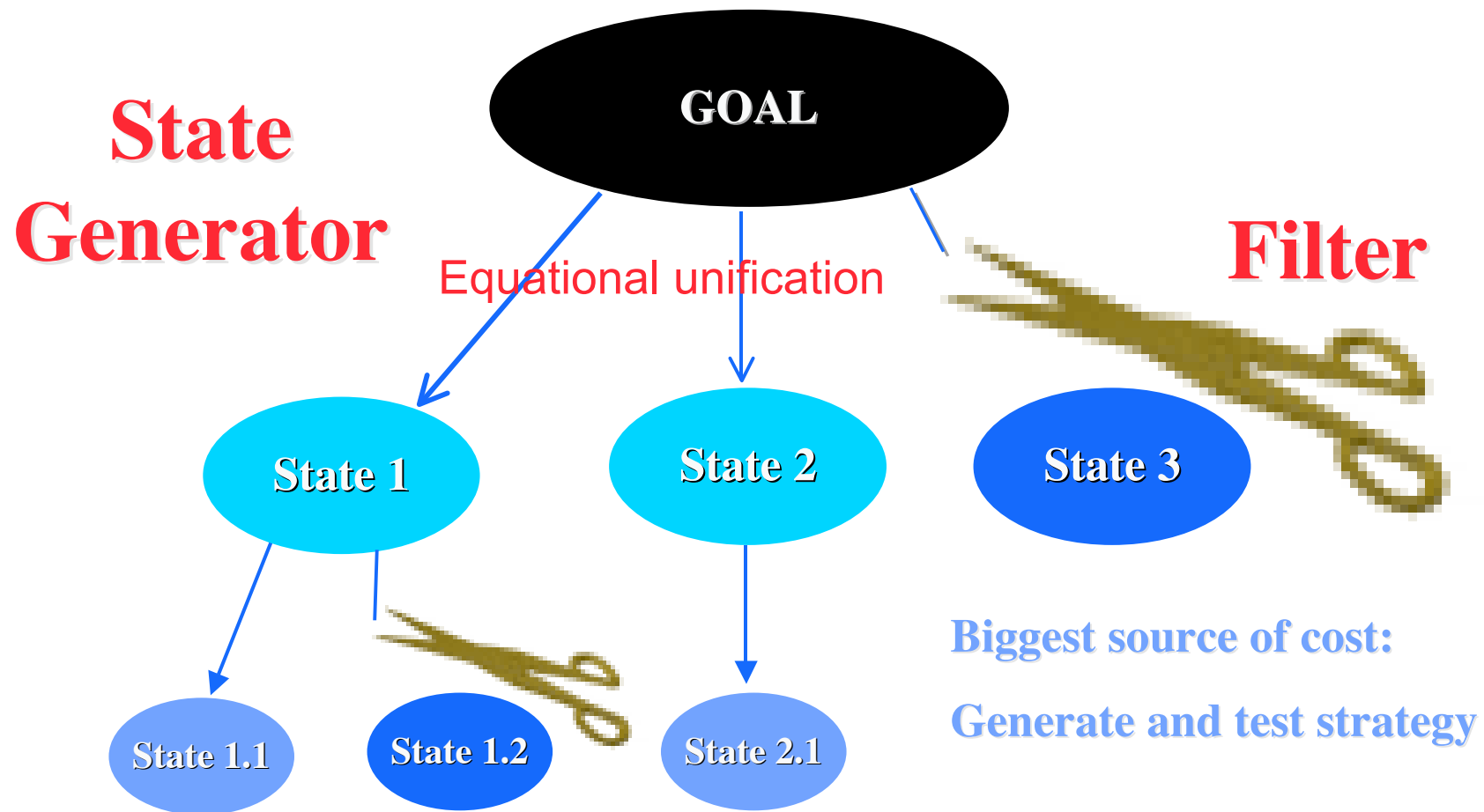
EARLY STATE EXPLORATION TOOLS

- **Interrogator (Millen, earliest version 1984)**
 - **First to automate intruder**
 - **Earliest version similar to a finite state model-checker**
 - **Searched finite number of rounds to find attack**
- **Ina Test (Kemmerer, 1987)**
 - **General-purpose symbolic execution tool, could be used to reproduce attacks**
- **NRL Protocol Analyzer (Meadows, earliest version 1989)**
 - **Automated intruder in a way similar to Interrogator**
 - **Implemented full Dolev-Yao model**
 - **Used symbolic representation of states, and supported use of lemmas to reduce infinite state space to finite one**
 - **In earliest version, lemmas proved by hand, later offered automated support**

NRL PROTOCOL ANALYZER

- A formal methods tool for the analysis of crypto protocols
- Uses standard Dolev-Yao model of intruder
- User specifies insecure state using a combination of constants and existentially quantified variables
 - NPA works backwards from state to determine if there is a path to it
 - May make substitutions to the variables as it goes
- Uses rewrite rules to specify properties of cryptosystems
 - Narrowing to match up rule outputs with state description
- Search space is initially infinite
 - User may prove a set of lemmas to cut down search space size
 - When a state is generated, lemmas are used to determine whether it should be kept or discarded

HOW NPA EXPLORES STATE SPACE



EXAMPLE

- **Rewrite rules:**

$$pke(pubkey(U), pke(privkey(U), X)) \rightarrow X$$

$$pke(privkey(U), pke(pubkey(U), X)) \rightarrow X$$

- **Rule for signing**

If receive message X, then produce message $pke(privkey(server), X)$

- **Try and find state in which intruder knows word Y**
- **First result: $Y = pke(privkey(server), X)$, intruder needs to know X**
- **Second result:**

$$X = pke(pubkey(server), W),$$

$$Y = pke(privkey(server), pke(pubkey(server), W)) = W$$

Intruder needs to know $X = pke(pubkey(server), W)$

- **Can use rule as oracle for decrypting encrypted messages**

EXAMPLE OF LEMMA GENERATION

- Consider protocol with only one rule:
 - IF a rcvs X THEN a sends $d(k,X)$
- Suppose you want to know if the intruder can learn m, not known initially
 - If you want to know word m, system tells you that you need to know $e(k,m)$
 - If you want to know $e(k,m)$ system tells you that you need to know $e(k,e(k,m))$, and so on
 - Try to show the unobtainability of the formal language A defined by

1. $A \rightarrow m$

2. $A \rightarrow e(k,A)$

PROVING THE LEMMA

- Try to show the unobtainability of the formal language A defined by
 - 1. $A \rightarrow m$
 - 2. $A \rightarrow e(k,A)$
 - Already tried Analyzer on first production
 - Try it on second, find that you need to know $e(k,e(k,A))$, which is also in A
- Have proved that, in order to learn a word from A, need to already know a word from A
- Analyzer can show automatically that languages are unreachable

THE AGE OF BELIEF LOGICS

- Burrows, Abadi, and Needham (BAN) logic for analyzing authentication protocols, 1989
 - Codified common-sense reasoning about cryptographic properties into a set of rules about beliefs that could be inferred during operation of a crypto protocol
 - Easy to use and intuitively appealing
- A host of descendants, GNY, AT, SvO, AAPA (automated and augmented GNY)
- Trade-off high level of abstraction with efficiency and ease of use
- However, have been shown to be effective at flagging a large number of protocol errors [Brackin - 1998,99]

BURROWS, ABADI, AND NEEDHAM (BAN) LOGIC 1989

- **Builds upon statements about beliefs in messages sent through the course of a protocol**
- **Example: “If I’ve received a message encrypted with key K, and I believe that only Alice and I know K, then I believe that the message was originated by either Alice or me.”**
- **In the analysis of a protocol, an initial set of beliefs is assumed.**
- **Each message received is mapped to another set of beliefs.**
- **Inference rules used to determine what beliefs can be derived from initial beliefs and beliefs gained from participating in the protocol**
- **If resulting set of beliefs adequate, the protocol is assumed to be correct**
- **If set of beliefs not adequate, this observation can lead to the discovery of a security flaw**

SOME BAN INFERENCE RULES

- 1. (message-meaning) If P believes that K is a key shared with Q AND P sees K[X] THEN P believes that Q said X**
- 2. (nonce-verification) If P believes that X is fresh AND P believes that Q said X THEN P believes that Q believes X**
- 3. (freshness) If P believes that X is fresh, then P believes that $\langle X, Y \rangle$ is fresh**
- 4. (jurisdiction) If P believes that Q believes X AND P believes that Q controls X THEN P believes X**

EXAMPLE BAN ANALYSIS (SKETCH)

1. $A \rightarrow S: B, N_A$

2. $S \rightarrow A: K_A[N_A, B, K_{AB}]$

- **Initial beliefs:** A believes N_A is fresh, A believes , for all K, S controls the fact that K is a good key for A and B, A believes that K_A is a good key for A and S
- From message 2 and message meaning rule, conclude that A believes S once said $\langle N_A, K_{AB} \text{ key for A and B} \rangle$
- From A believes that N_A is fresh, and freshness rule, conclude that A believes that $\langle N_A, K_{AB} \text{ key for A and B} \rangle$ is fresh
- From A believes that $\langle N_A, K_{AB} \text{ key A and B} \rangle$ is fresh, and A believes S once said $\langle N_A, K_{AB} \text{ key for A and B} \rangle$, and nonce-verification, conclude that A believes S believes $\langle N_A, K_{AB} \text{ key for A and B} \rangle$
- From A believes S believes $\langle N_A, K_{AB} \text{ key for A and B} \rangle$, conclude A believes S believes K_{AB} key for A and B
- From A believes S controls K for A and B, and A believes S believes K_{AB} key for A and B, and jurisdiction rule, conclude A believes K_{AB} key for A and B

THE AGE OF MODEL CHECKERS

- Goal of a model checker: to verify system properties by exhaustive search of a finite set of states
- How a model checker is used
 - Specify your system
 - Specify the behavior of your system, usually using a temporal logic
 - Use the model checker to exhaustively generate possible behaviors of the system that can violate the desired behavior
 - For example, generate all paths possible starting from an initial state
 - Can only search finite space, but with suitable abstractions, can also use on infinite systems

LOWE'S USE OF FDR

- Lowe first to use a model checker (FDR) to demonstrate a protocol failure (Needham-Schroeder public key protocol)
- FDR: model checker based on Hoare's communication sequential processes
- User can specify size of search space in terms of number of participants, protocol executions, etc.
- Some work since then
 - Developed high-level specification language, Casper
 - Conditions that will guarantee that, if a protocol is secure for a finite search space, then it is secure for an infinite space

NEEDHAM-SCHROEDER PUBLIC-KEY PROTOCOL

1. $A \rightarrow B: [R_A, A]K_B$
2. $B \rightarrow A: [R_A, R_B]K_A$
3. $A \rightarrow B: [R_B]K_B$

THE ATTACK

1. $A \rightarrow I: [R_A, A]K_I$
- 1': $IA \rightarrow B: [R_A, A]K_B$
- 2': $B \rightarrow A: [R_A, R_B]K_A$
3. $A \rightarrow I: [R_B]K_I$
- 3': $IA \rightarrow B: [R_B]K_B$

CHALLENGES IN MODEL-CHECKING CRYPTO PROTOCOLS

- Sources of infinity
 - unbounded number of actions by intruder
 - unbounded number of concurrent sessions that could interact
 - unbounded number of data items such as keys, nonces, etc.
 - Roscoe et al. have shown that, if you bound the number of concurrent sessions, can also bound nonces, etc.
- Undecidability of secrecy problem
 - easy to show that this also implies undecidability of authentication problem

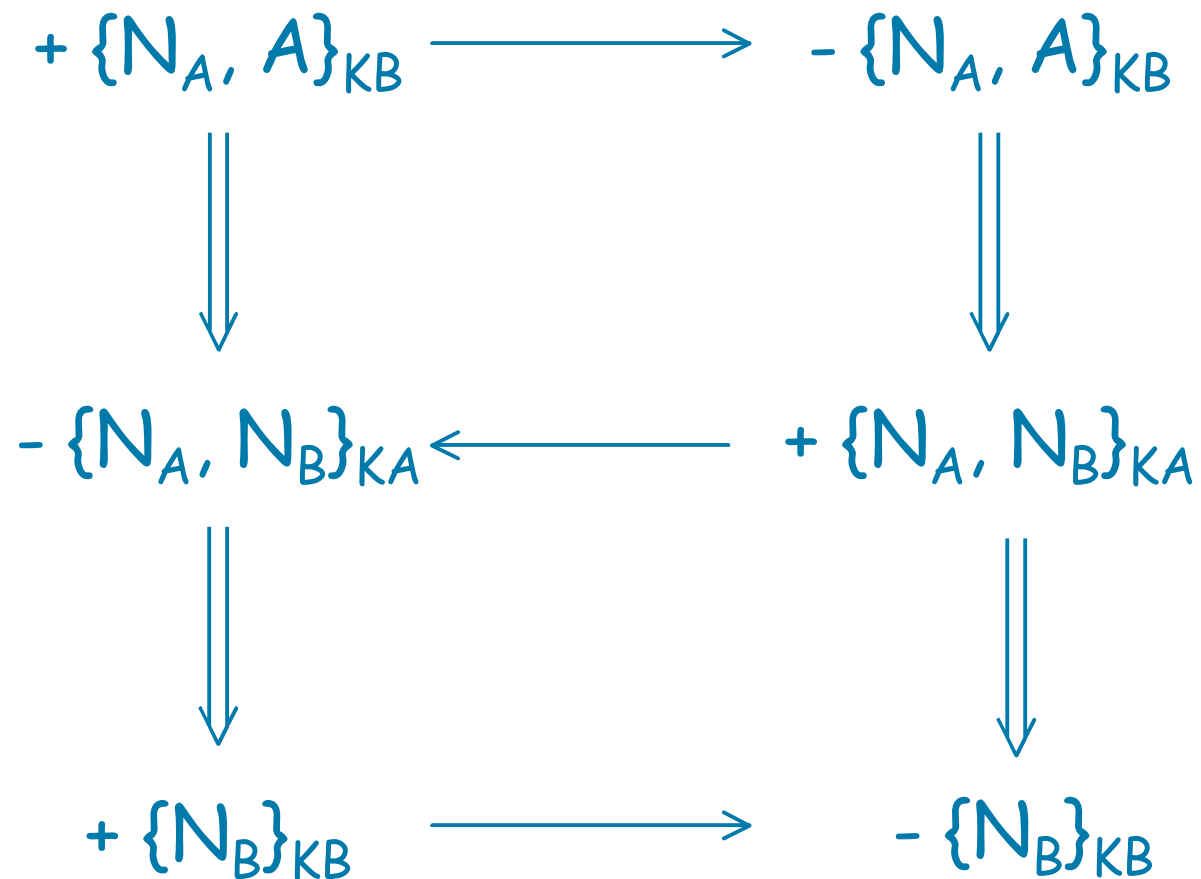
THEOREM PROVERS

- **Paulson's inductive approach**
 - Uses Isabelle theorem prover to inductively define invariants
 - Has developed library of techniques which have been taken up by others
- **Some special-purpose tools**
 - TAPS (Cohen)
 - Cryptyc Type Checker (Gordon and Jeffries)
- **Many, many more**

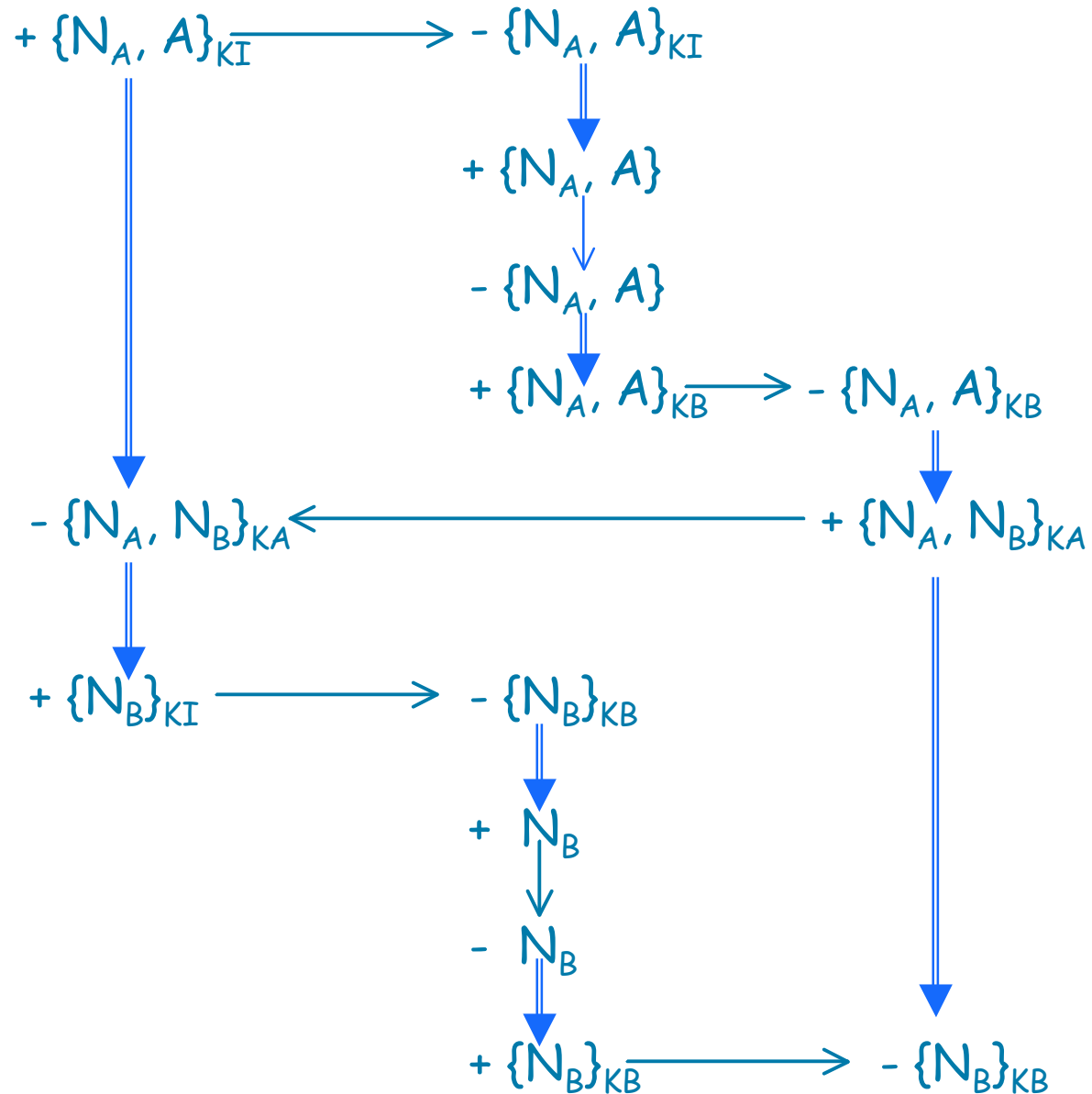
STRAND SPACES (THAYER, HERZOG,GUTMANN)

- **Graph-theoretic model for crypto-protocol analysis**
 - **Similar to Lamport's theory of causality**
- **Strands (linear graphs) represent local executions**
 - **Protocol strands: local execution of protocol participant in one protocol instance**
 - **Penetrator strands: describe actions such as concatenation, encryption, decryption, etc.**
- **Strands contain both positive (output) and negative (input) nodes**
- **Can glue up strands along positive and negative nodes to obtain a bundle**

EXAMPLE BUNDLE - NS PUBLIC KEY



ATTACK ON NS PUBLIC KEY



ADVANTAGES OF STRAND SPACES

- Simple, easy to work with model
- Graph-theoretic approach easy to visualize
- Can use as basis of comparison with other models
- We'll meet up with strand spaces again

OPEN PROBLEMS



Emerging Properties of Protocols

- **Greater interoperation**
 - Meta protocols to negotiate agreement upon protocols, eg. ISAKMP-IKE
- **Negotiation of policy**
 - Security associations
 - Certificate hierarchies
- **Greater complexity**
 - Especially for electronic commerce
- **Group-oriented protocols**
- **Emerging security threats**
 - Denial of service
 - Traffic analysis

ISSUE 1:

Composability

- **Avoid harmful interactions between protocols**
 - **Example, consider a protocol suite of slightly different protocols that satisfy different requirements**
 - **Need to be sure that one doesn't get confused with another**
- **Assure secure reliance between protocols**
 - **Suppose that one protocol relies on another for a particular service**
 - **Assure that service cannot be compromised or spoofed**

ISSUE 2: Incremental Analysis

- Many protocols part of a family of protocols that differ only slightly
 - Would be helpful to be able to reuse proofs
- When integrating formal analysis in design, necessary to be able to redo proofs rapidly as design changes
 - Again, would be helpful to be able to reuse parts of proofs that are still valid
- But -- analysis of crypto protocols requires checking for unsafe interaction of all parts
 - How can we tell if any of our proofs are still valid
- Closely related to modular evaluation problem
- See work by Datta, Derek, Mitchell, & Pavlovic Pavlovic & Meadows
 - Go back to epistemic logic, but reason on more concrete level
 - E.g. knowledge of sequences of events that occurred
 - Concentrate on properties that are monotonic

ISSUE 3: PROBABILISTIC AND GAME-THEORETIC ANALYSIS

- Many attacks on protocols are probabilistic
 - Traffic Analysis
 - Can an attacker increase the accuracy of its picture of the network?
 - Protocols Using Lightweight Authentication
 - How much do they reduce the likelihood for a successful attack?
- Many protocol properties can be modeled in terms of games
 - Electronic commerce protocols
 - Denial of service
 - Trading off effort of attacker against cost to defender

ISSUE 4:

High Fidelity

- There are a number of protocol problems that occur below the level at which most formal methods work operates, but above the cryptoalgorithm level
 - Modes of encryption
 - Integrity checks
 - Bellovin (Usenix 96) describes a cut-and-paste attack on an earlier version of ESP relying on ESP's use of CBC mode and TCP's not checking length of packets
 - Malleable cryptosystems
 - Some techniques (e.g. Chaum's blinded signatures, actually make use of malleability)
- Some work exists on formal models for properties at this level
 - Stubblebine-Gligor (Security and Privacy '92, '93)
 - Can automated tools be extended to deal with these properties?
- When can we prove that it is safe to ignore these properties
 - E.g, that protocol is sound against attacks on this level

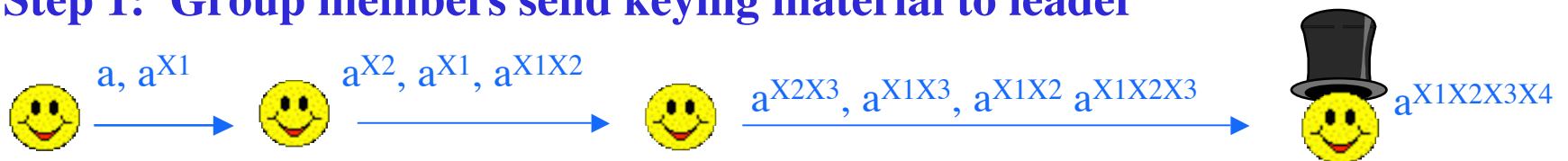
ISSUE 5:

Open-Ended and Group Protocols

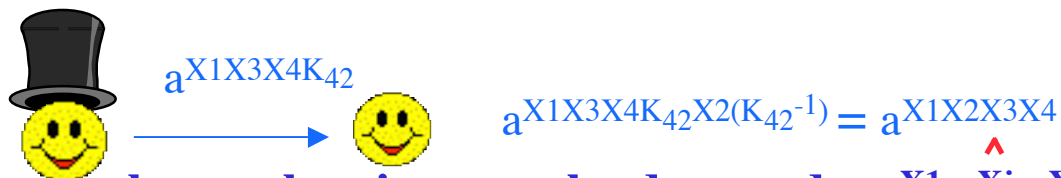
- Many of the newer protocols use open-ended data structures
- Examples
 - Protocols for negotiating choices from an open-ended set of algorithms
 - Group-oriented protocols
 - Group authentication
 - Secure auction, voting, etc.
 - Anonymity: Chaum mixes
- Beginning to see more work done in this area

CLIQUE GROUP KEY DISTRIBUTION PROTOCOL

- Group leader n shares common secret exponent K_{nj} with each principal j
- Step 1: Group members send keying material to leader



- Group leader raises each value received to the X_n .
- At the end, for each group member j , group leader has $a^{X1 \dots Xj \dots Xn}$.
- Group leader also has group key $a^{X1 \dots Xn}$.
- Step 2: Leader sends key info to members, encrypted w. shared secret exponents



- For each member j , group leader sends $a^{X1 \dots Xj \dots Xn K_{nj}}$ to j .
- Group member j computes $a^{X1 \dots Xj \dots Xn K_{nj} Xj(K_{nj}^{-1})} = \text{group key}$

ISSUE 6: TRANSACTION PROCESSING

- E-commerce protocols such as SET allow parties to agree on complex data structures
 - Need to assure integrity and consistency of the structure
 - Different parts of the structure introduced by different principals
 - Some parts of the structure kept secret from some of the principals
- Protocols such as ISAKMP-IKE agree on somewhat less complex, but open-ended data structures
- Otway-Rees protocol (1987) tried to agree on a two-part data structure (key and key id)
 - Thayer, Herzog, and Guttman (1998) showed protocol didn't achieve that
- This is a problem that could use some close attention

ISSUE 7:

Getting it into the Real World

- Protocol developers are aware that assurance is a problem
 - If you care enough about security to use cryptography, you probably care enough to worry about whether you're using it correctly
- But -- formal methods are seen as an arcane field only accessible to a few experts
 - Seen as a barrier to integration of formal methods into the design process
 - Promotes adversarial relationship between developers and verifiers
 - Leads to inflexibility in application of formal methods
- How can we make formal methods more accessible and more flexible?
- What is the best way of introducing them into the design process?

BIBLIOGRAPHY

These talks based on the following papers:

Meadows, C., "Formal Methods for Cryptographic Protocol Analysis: Emerging Issues and Trends", IEEE Journal on Selected Areas in Communication, Vol. 21, No. 1, pp. 44-54, January 2003.

Meadows, C., "Open Issues in Formal Methods for Cryptographic Protocol Analysis," Proceedings of DISCEX 2000, IEEE Computer Society Press, pp. 237-250, January, 2000.

Meadows, C. "Formal Verification of Cryptographic Protocols: A Survey," Advances in Cryptology - Asiacrypt '94, LNCS 917, Springer-Verlag, 1995, pp. 133-150.

All of the above available at

<http://chacs.nrl.navy.mil/projects/crypto.html>