Cryptographic Protocols

NISNET Finse Winter School, May 3-8, 2009

Stig F. Mjølsnes

Dep.Telematics, NTNU

May 4, 2009

Information Security of Communication

- Confidentiality, Integrity and Availability
- "The Science of Information Integrity (Simmons)
- Researchers distinguish between

  Cryptographic Primitives Mathematical Functions
  e.g. One-way (hash) functions, . . .

  Crypto Protocols Use the primitives in communication goals
  e.g. Key Distribution, . . .
- Secure primitives are *necessary but not sufficient* for secure communication

On Padlocks, Bikes and Protocols

Friendly thanks to Cas Cremers

What is a protocol?

Communication Protocol is a set of rules that controls the interaction of n parties or principals. First used in a 1967 NPL memo

Roles Sender—Recipient, Client—Server, Initiator—Responder, Alice—Malice—Bob

Multiparty Normally $n \geq 3$, f.ex. The Dining Cryptographers

Communication Channel The medium enabling the message exchange between the parties. One-to-one, multi- or broadcast, secrecy channel, noisy/perfect. Storage channel allows defining the 1-party case.

Cryptographic Protocol is a communication protocol that includes one or more cryptographic primitives.

Runs and Rounds A run is one instance of a protocol execution with variable assignments. The protocol may be repeated in several rounds.

Protocol As a Language

- Language
  Alphabet of symbols, syntax of words in the language, and the grammer rules of constructing sentences.
- Protocol
  Coding of the messages, the protocol messages, and sequence of message exchanges according to the protocol generates acceptable sentences in the language.
- The effect/service to the communicating parties is analogous to the semantics of the language.
- As for a common language, there must be an a priori agreement between the communicating parties about the rules and procedures of the protocol.

Elements of Protocol Description

1. **The assumptions** about the protocol environment, in particular the properties of the channel(s)
2. **The encoding** of each message in the vocabulary
3. **The vocabulary** of messages that can be used in the exchange
4. **The behaviour** and **processing rules** required of the communicating parties by the protocol
5. **The service** provided to the communicating parties by the protocol

Elements of Protocol Description

1. **The assumptions** about the protocol environment, in particular the properties of the channel(s)
2. **The encoding** of each message in the vocabulary
3. **The vocabulary** of messages that can be used in the exchange
4. **The behaviour** and **processing rules** required of the communicating parties by the protocol
5. **The service** provided to the communicating parties by the protocol

Elements of Protocol Description

1. **The assumptions** about the protocol environment, in particular the properties of the channel(s)
2. **The encoding** of each message in the vocabulary
3. **The vocabulary** of messages that can be used in the exchange
4. **The behaviour** and **processing rules** required of the communicating parties by the protocol
5. **The service** provided to the communicating parties by the protocol

Elements of Protocol Description

1. **The assumptions** about the protocol environment, in particular the properties of the channel(s)
2. **The encoding** of each message in the vocabulary
3. **The vocabulary** of messages that can be used in the exchange
4. **The behaviour** and **processing rules** required of the communicating parties by the protocol
5. **The service** provided to the communicating parties by the protocol

Elements of Protocol Description

1. **The assumptions** about the protocol environment, in particular the properties of the channel(s)
2. **The encoding** of each message in the vocabulary
3. **The vocabulary** of messages that can be used in the exchange
4. **The behaviour** and **processing rules** required of the communicating parties by the protocol
5. **The service** provided to the communicating parties by the protocol

Elements of Protocol Description

1. **The assumptions** about the protocol environment, in particular the properties of the channel(s)
2. **The encoding** of each message in the vocabulary
3. **The vocabulary** of messages that can be used in the exchange
4. **The behaviour** and **processing rules** required of the communicating parties by the protocol
5. **The service** provided to the communicating parties by the protocol

Modeling: Function or Reaction?

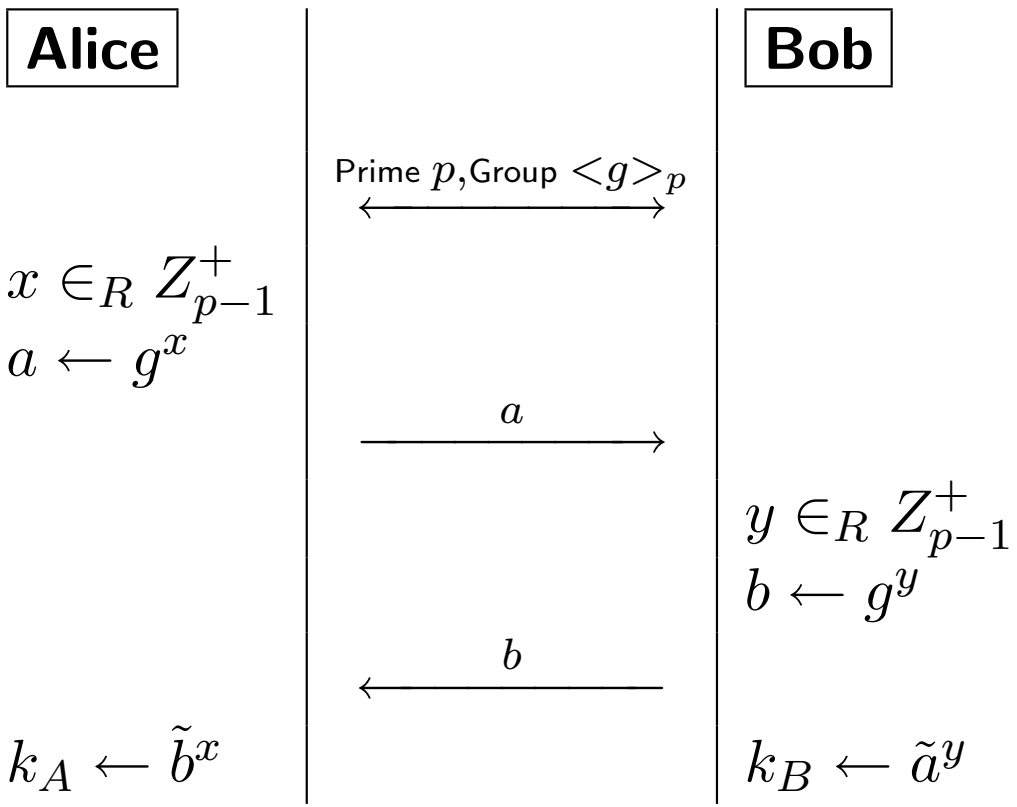
Function An input—output function description $y = f(x)$ The *algorithm* transforms some input to some output, then stops.

$$y \leftarrow A_f(x), \text{ where } x, y \in \{0,1\}^*$$

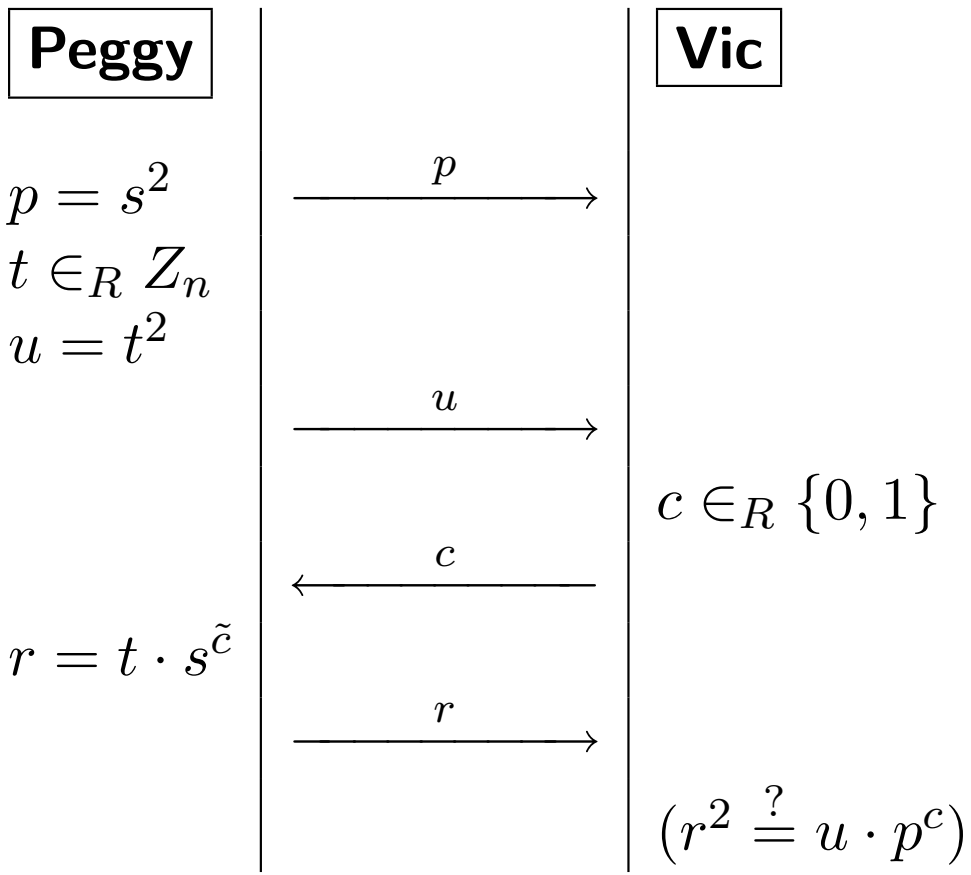
Behaviour A reactive finite state machine description. Accepts some input message, generates some output message, and changes its inner state variable, indefinitely or until some stop state.

$$(Q, q_0, M, T(q, a))$$

Message Sequence Description



Another Example



Input—Output Description

Note that in cryptoprotocols (as of today) there is no intermediate branching except error handling. If error occurs then stop. Hence a cryptoprotocol is a determined sequence of computations, a *normal* protocol run is a composition of input and output values. Private keys and randomization are just another input to the function.

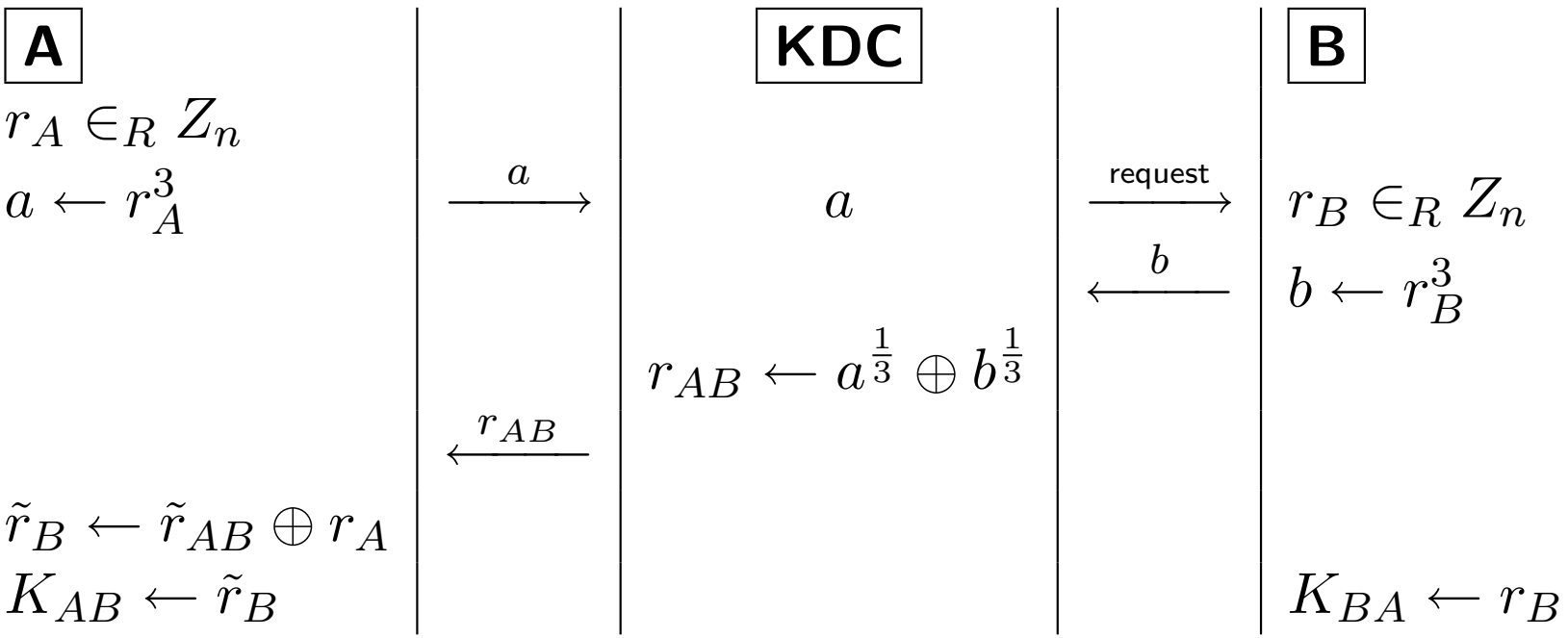
Each principal view: $k_A \leftarrow \Pi_A(p, g, x, \tilde{b}), k_B \leftarrow \Pi_B(p, g, y, \tilde{a})$

Total view: $(k_A, k_B) \leftarrow \Pi(p, g, x, y)$

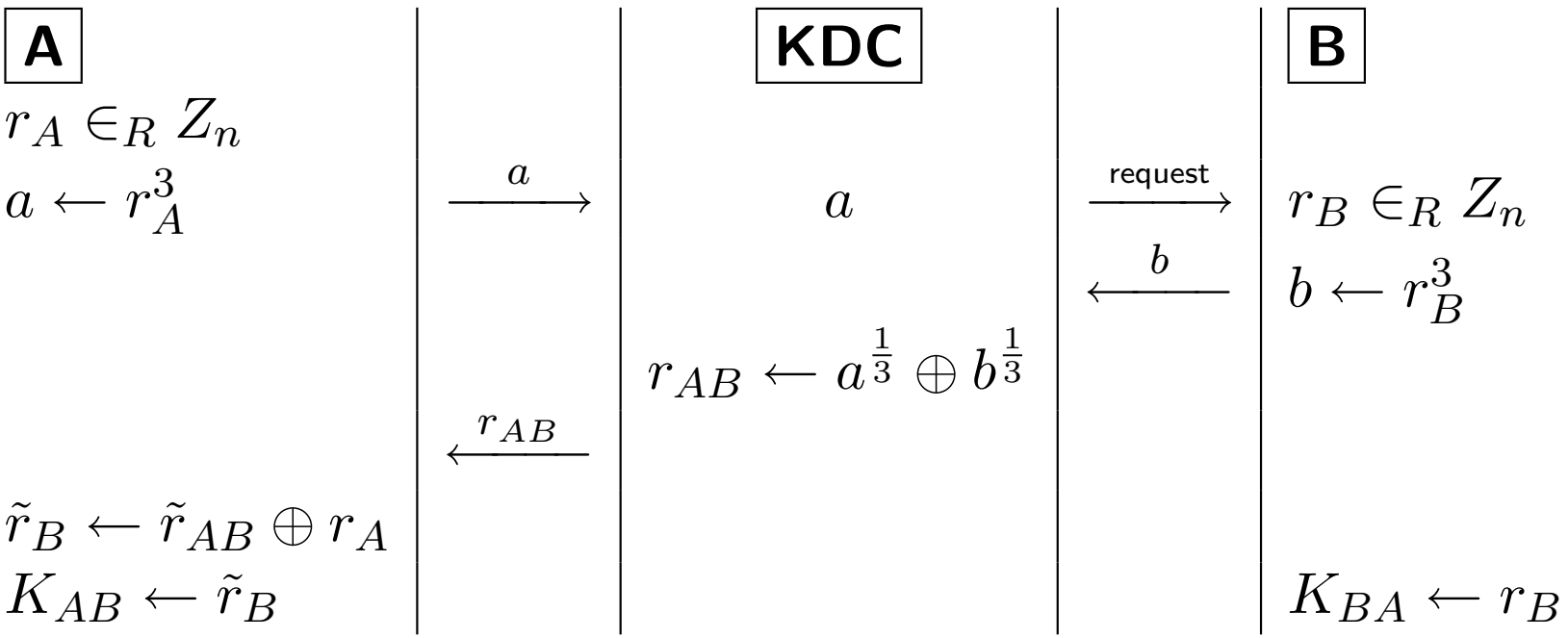
Attack Models

- Passiv wiretapping model (Shannon)
+ Active outsider attacker (Needham-Schroeder)
+ Active insider attacker (Dolev-Yao)
  1. Malice is a legitimate principal.
  2. She can initiate interaction with all principals ("black boxes").
  3. She is the network (NS-model)
+ Allow for collusion of 'maliced' principals.

TMN Key Distribution Protocol



TMN Key Distribution Protocol



Two provably secure primitives, one information-theoretic, one under computational complexity security (RSA with e=3, cuberoot extraction.) Attacker input (a, b, r_{AB}) solve $r_B = a^{1/3} \oplus r_{AB}$ or $r_B = b^{1/3}$. Secure?

Is the TMN protocol secure?

- Yes, it appears all right under active outside attack.
- No, not secure if collusion/cooperation of principals can happen.
- Well, actually not even under the Dolev-Yao model because Malice can take on the role of several principals, hence she will act as two principals.
- Hint for the attack: Observe that KDC is a decryption service, then input $r^3 \cdot b$ where b is from a previous observed run.

TMN Protocol Attack

Wiretapped b. D provides C with r_D.

C		**KDC**		**D**
$r_C \in_R Z_n$				$r_D \in_R Z_n$
$c \leftarrow r_C^3 \cdot b$	$\xrightarrow{c}$	a	$\xrightarrow{\text{request}}$	
			$\xleftarrow{d}$	$b \leftarrow r_D^3$
		$r_{CD} \leftarrow c^{1/3} \oplus d^{1/3}$		
	$\xleftarrow{r_{CD}}$			
$r_{CD} = r_C \cdot r_B \oplus r_D$				
$r_B = (r_{CD} \oplus r_D)/r_C$				

TMN Protocol Attack

Wiretapped b. D provides C with r_D.

C		**KDC**		**D**
$r_C \in_R Z_n$				
$c \leftarrow r_C^3 \cdot b$	$\xrightarrow{c}$	a	$\xrightarrow{\text{request}}$	$r_D \in_R Z_n$
			$\xleftarrow{d}$	$b \leftarrow r_D^3$
		$r_{CD} \leftarrow c^{1/3} \oplus d^{1/3}$		
	$\xleftarrow{r_{CD}}$			
$r_{CD} = r_C \cdot r_B \oplus r_D$				
$r_B = (r_{CD} \oplus r_D)/r_C$				

Alternatively, C sends just b. Flaw: input is assumed random without structure. Kind advice: Always check to see what happens if random values are chosen non-random and with structure.

Check the resilience against sharing private input?

Reasons for Failure

- Incorrect cryptoprimitives
- Incorrect protocols
- Incorrect implementations
- Incorrect environment assumptions
- Incorrect operations

A Succinct Statement

Security protocols are three-line programs that people still manage to get wrong. — Roger Needham

Simmon's Principle 1

G.J.Simmons. Cryptanalysis and Protocol Failures. Comm.ACM 37(11) Nov 1994

Carefully **enumerate all of the properties** of all of the quantities involved, both those explicitly stated in the protocol specification and those implicitly assumed in the setting.

Simmon's Principle 2

- Take nothing for granted.
- **Go through the list** of properties assuming that none of them are as they are claimed or tacitly assumed to be unless a proof technique exists to either enforce or verify their nature.
- For each possible violation of a property, critically examine the protocol **to see if this makes any difference in the outcome** of the execution of the protocol.
- **Consider combinations** of parameters as well as single parameters.

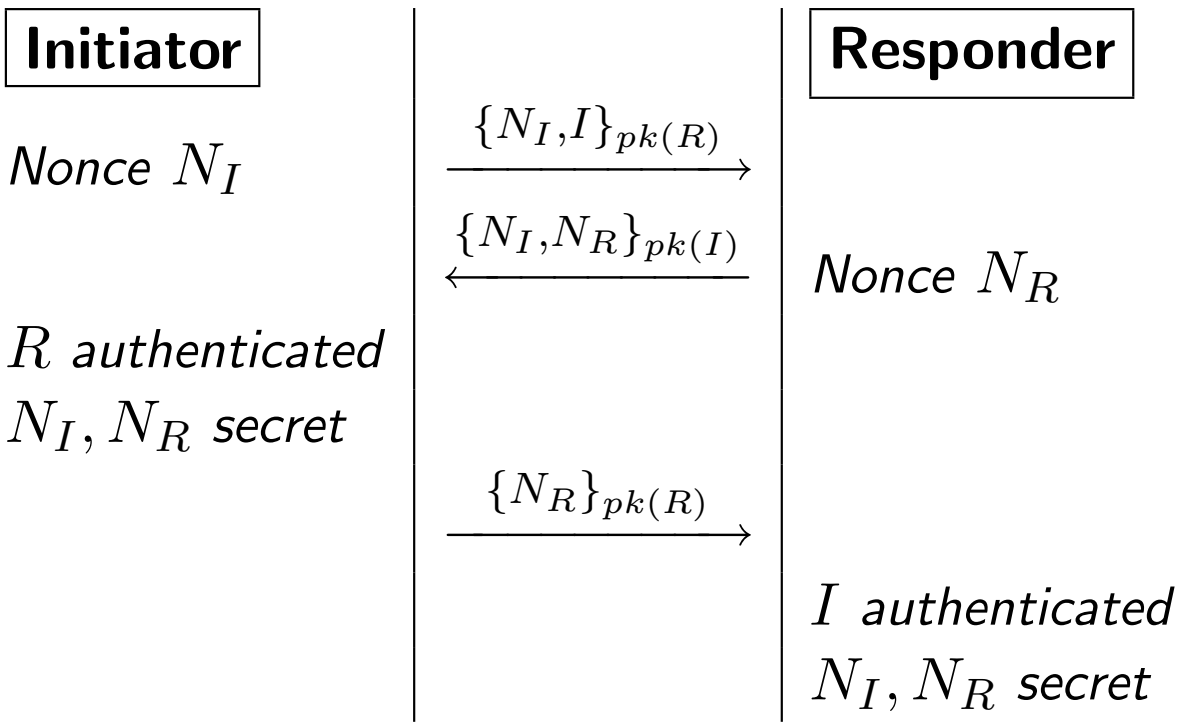
Simmon's Principle 3

- **If the outcome of the protocol can be influenced** as a result of a violation of one or more of the assumed properties, it is essential to then **determine whether this can be exploited** to advance some meaningful deception.
- Example: DH protocol can, but no meaningful deception.
- Protocol failures occur whenever the function of the protocol can be subverted as a consequence of the violations.

Divide and Conquer Approach

- Separate the concerns of primitive and protocol, assume perfect cryptoprimitives $\{M\}_K$
- Commonly used protocol specification
  1. $I \rightarrow R : \{na, I\}_{PK(R)}$
  2. $R \rightarrow I : \{na, nb, I\}_{PK(I)}$
  3. $I \rightarrow R : \{nb\}_{PK(R)}$
- Very incomplete and deceptive notation
- Additionally needs: declarations of constants, variables, functions, inverse keys, types, pre/postconditions, etc.

A classical protocol



Is the protocol correct wrt. to the claimed properties?

- It has been proved secure (in BAN logic).

Is the protocol correct wrt. to the claimed properties?

- It has been proved secure (in BAN logic).
- Sorry, there exists a man-in-the-middle attack! (Lowe)
  Exercise: Find it.

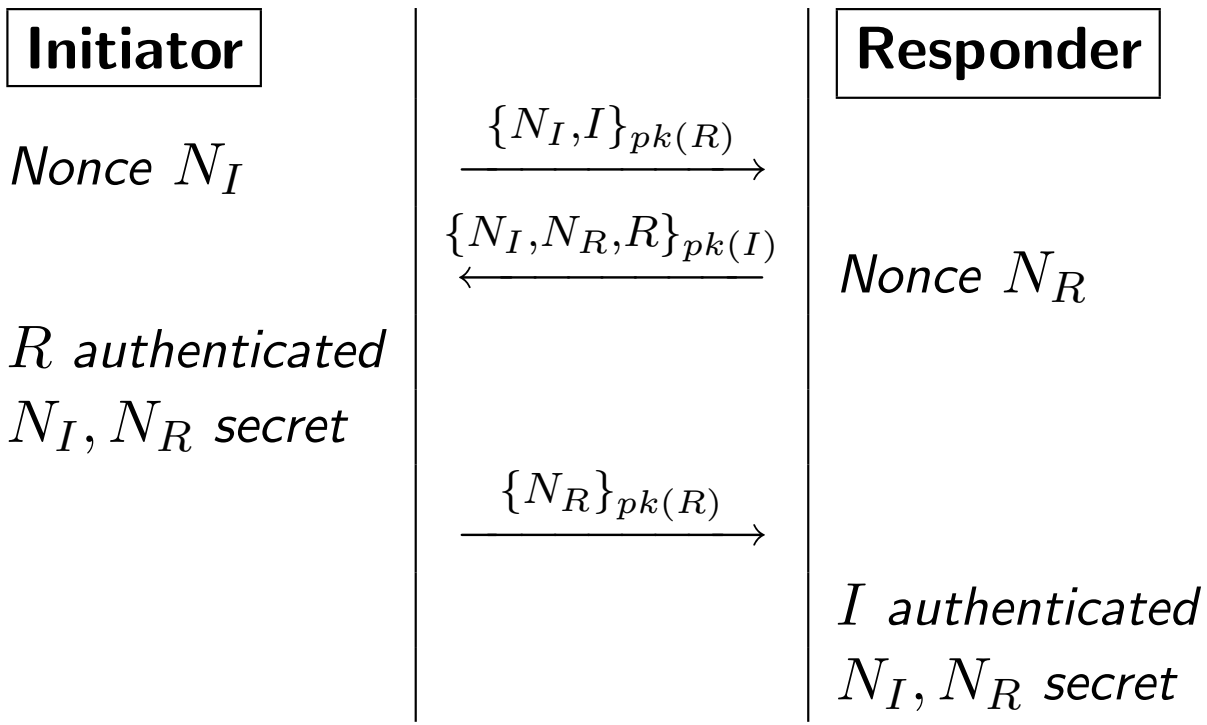
Is the protocol correct wrt. to the claimed properties?

- It has been proved secure (in BAN logic).
- Sorry, there exists a man-in-the-middle attack! (Lowe)
  Exercise: Find it.
- Why is this so difficult to spot for a "three-line program"?

Is the protocol correct wrt. to the claimed properties?

- It has been proved secure (in BAN logic).
- Sorry, there exists a man-in-the-middle attack! (Lowe)
  Exercise: Find it.
- Why is this so difficult to spot for a "three-line program"?
  - quite complex underlying model despite the suggestion of simplicity in the description.
  - Unbounded number of role instances of the protocol
  - Assumptions about the environment and attackers are not clear or implicit.

Here is a fix


Initiator
Responder
Nonce N_I
$\{N_I, I\}_{pk(R)}$
$\{N_I, N_R, R\}_{pk(I)}$
Nonce N_R
R authenticated
N_I, N_R secret
$\{N_R\}_{pk(R)}$
I authenticated
N_I, N_R secret

Prudent Engineering Practice for Cryptographic Protocols

Abadi & Needham. IEEE Trans.Software Engineering 22(1) 1996)

- Heuristics for the engineering of good cryptoprotocols
- Experience and intuition based on observations of common patterns in the records of flawed proposals of the 1980-90s
- Analytical formalisms do not themselves give design rules.
- Examples from authentication protocols, but claimed generally applicable.

Basic Principle 1: **Explicit Communication**

Every message should say what it means: its interpretation should depend only on its content. It should be possible to write down a straight-forward sentence describing the content—or in a suitable formalism available.

- "Authentication of any message in the protocol depends only on information contained in the message itself or already in the possession of the recipient."
- "Always include in the outgoing encrypted message all the information gathered.'

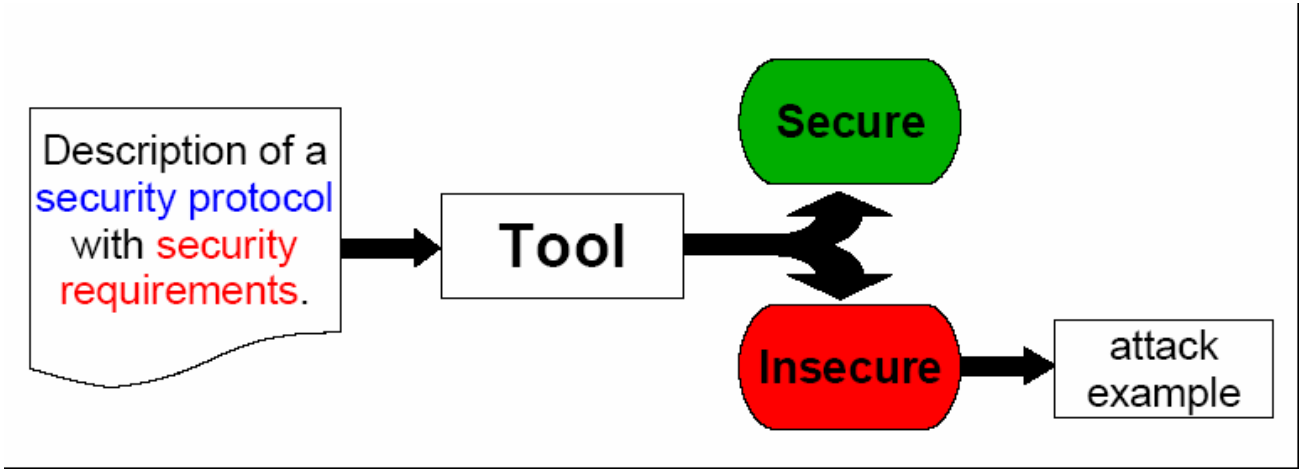
Basic Principle 2: **Appropriate conditions for action**

The conditions for a [received] message to be acted upon should be clearly set out so that someone reviewing a design may see whether they are acceptable or not.

Some Formal **Analysis** Tools

- Making statements about protocols manually is hard, we need machine tools!
- BAN (Belief) Logic of authentication (prove correctness, flaws are found indirectly). (Casper/FDR, Murphi, Brutus, NRL, Athena), CoProve, Hermes, AVISPA, ProVerif
- Model checking: systematic (even exhaustive) search for undesirable properties never occur (safety) or always occur (liveness).

Cremer's Scyther Protocol Analysis Tool

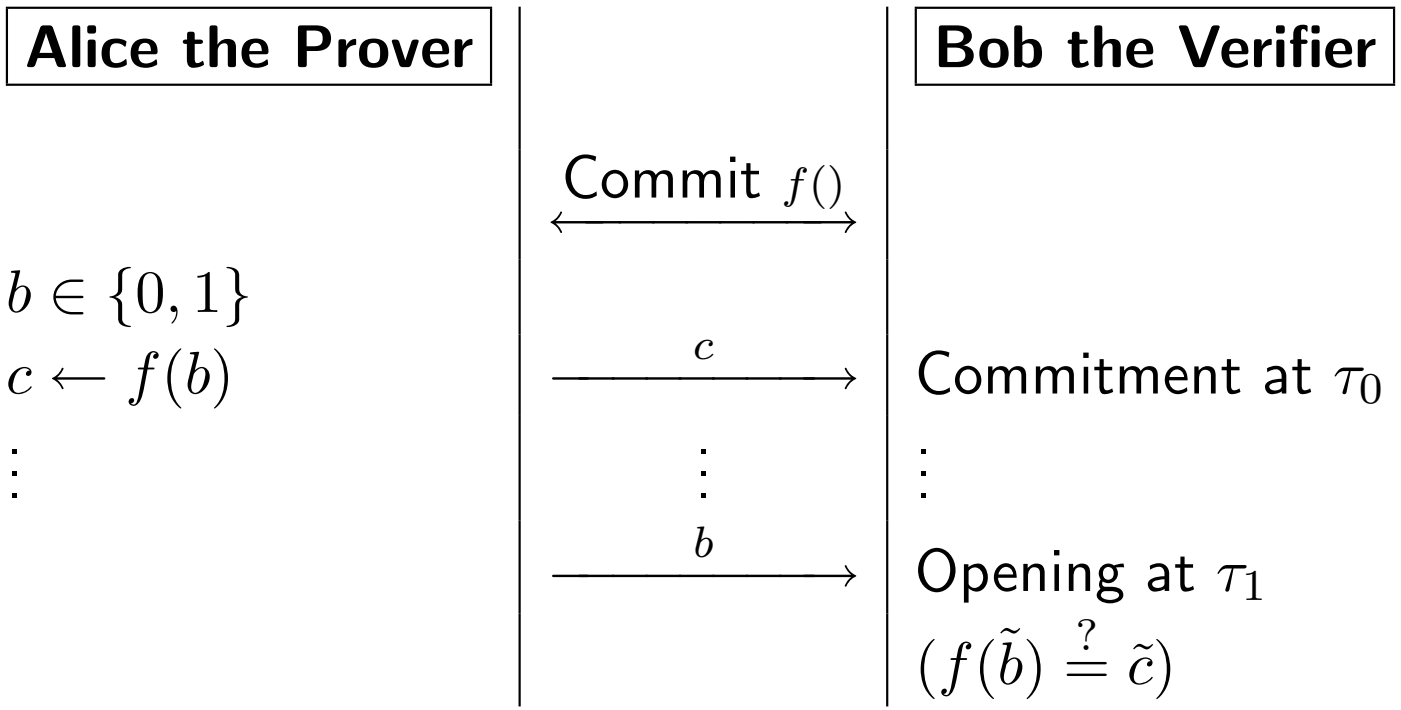


- Specialized model checker for authentication protocols that tries to emulate simple theorem proving methods
  - Uses backward trace search, so guaranteed to stop
- Easy language for protocol specifications and claims
- Shows attack scenarios graphically
- The fastest method and tool available. Check it out!

Assignment/Exercise

Make *your definition* of the properties of *authenticity* and *data integrity* in a protocol context. Then check implications of your definitions whether one is a necessary condition of the other.

An Elementary Cryptographic Protocol



Properties required:

Binding The Prover can only open c with b after τ_0.

Hiding The Verifier cannot know b before τ_1

Bit Commitment Algorithms

Generation algorithm run either by P or V.
Output: **commit:** $\{0,1\}^\ell \times \{0,1\} \mapsto \{0,1\}^\ell$

Commitment algorithm run by P on the secret value b.
$c \leftarrow$ **commit**(r, b) where $r \in_R \{0,1\}^\ell$

Opening of (r, b) by P, and V computes the predicate
if $(c \stackrel{?}{=}$ **commit**$(\tilde{r}, \tilde{b}))$ Accept **else** Reject

Prime Groups and Discrete Logarithm

- Let $< g >_p$ be a multiplicative group of prime order p with a generator g There is a unique x for each group element y such that $y = g^x$. Computing the inverse function $x = \log_g y$ is called the *discrete logarithm* w.r.t. $< g >_p$.
- Computing the discrete log is known to be hard for large p when $p - 1 = kq$, where q is a large prime number and k is small. (DLP assumption).
- Such groups and generators can be constructed efficiently for large p.
- The group multiplication and exponentiation can be computed efficiently.
- Group elements can be efficiently sampled with uniform distribution.
- Equality of group elements can be efficiently tested.

Commitment Flavours

$$\text{commit}(x) = g^x \mod p \text{ where } b = \text{lsb}(x) \quad (1)$$

Binding x is uniquely determined by c.

Hiding Discrete log problem (check details here).

$$\text{commit}(r, b) = g^r \cdot h^b \mod p \quad (2)$$

Binding Comp. hard to find two valid openings $(r, 0)$ and $(r', 1)$ for c because this yields $h = g^{r-r'}$.

Hiding Equal prob.distributions of $\text{commit}(r, 0)$ and $\text{commit}(r, 1)$.

"Security Flavours"	Binding	Hiding
Unconditional	(1)	(2)
Computational	(2)	(1)

Commitment Flavours

$$\text{commit}(x) = g^x \mod p \text{ where } b = \text{lsb}(x) \tag{1}$$

Binding x is uniquely determined by c.

Hiding Discrete log problem (check details here).

$$\text{commit}(r, b) = g^r \cdot h^b \mod p \tag{2}$$

Binding Comp. hard to find two valid openings $(r, 0)$ and $(r', 1)$ for c because this yields $h = g^{r-r'}$.

Hiding Equal prob.distributions of commit$(r, 0)$ and commit$(r, 1)$.

"Security Flavours"	Binding	Hiding
Unconditional	(1)	(2)
Computational	(2)	(1)

Commitment Flavours

$$\text{commit}(x) = g^x \mod p \text{ where } b = \text{lsb}(x) \tag{1}$$

Binding x is uniquely determined by c.
Hiding Discrete log problem (check details here).

$$\text{commit}(r, b) = g^r \cdot h^b \mod p \tag{2}$$

Binding Comp. hard to find two valid openings $(r, 0)$ and $(r', 1)$ for c because this yields $h = g^{r-r'}$.
Hiding Equal prob.distributions of commit$(r, 0)$ and commit$(r, 1)$.

"Security Flavours"	Binding	Hiding
Unconditional	(1)	(2)
Computational	(2)	(1)

The Impossibility of Unconditionally Binding and Hiding

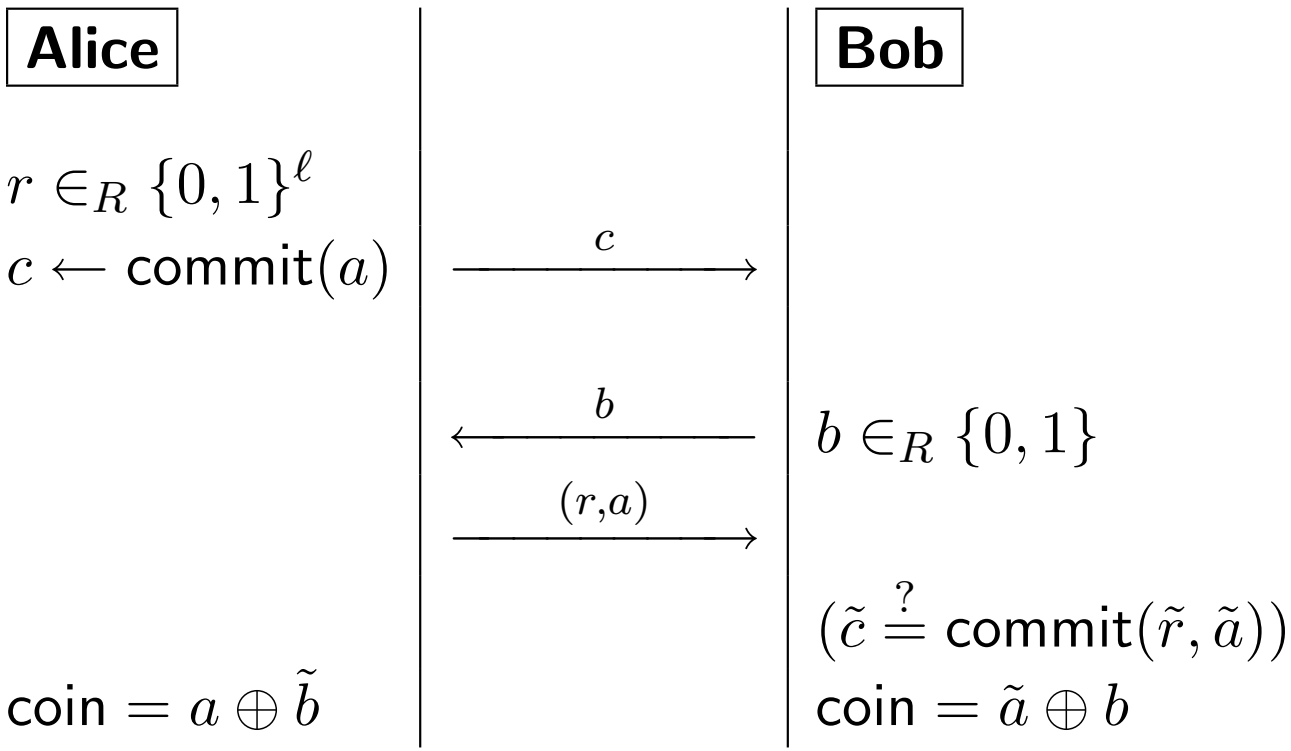
Sketch of proof

Imagine a commitment scheme that is both unconditional binding and hiding. Let $c = \text{commit}(r, 0)$ be the commitment of P.

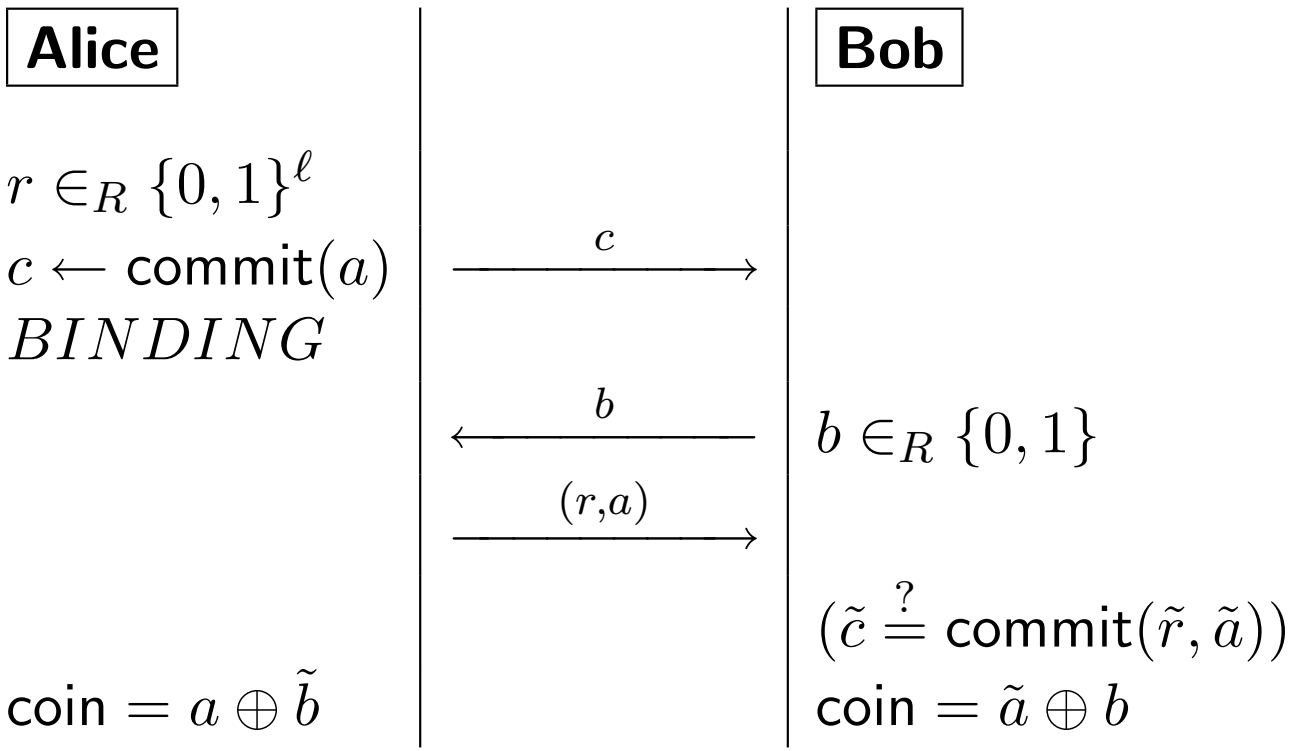
There must exist at least another value r' for $c = \text{commit}(r', 1)$. If not, the comp.unbounded V can uniquely determine the committed bit from P, and unconditional hiding will fail.

But now comp. unbounded P is able to open c both ways, and unconditional binding fails. Hence the theorem.

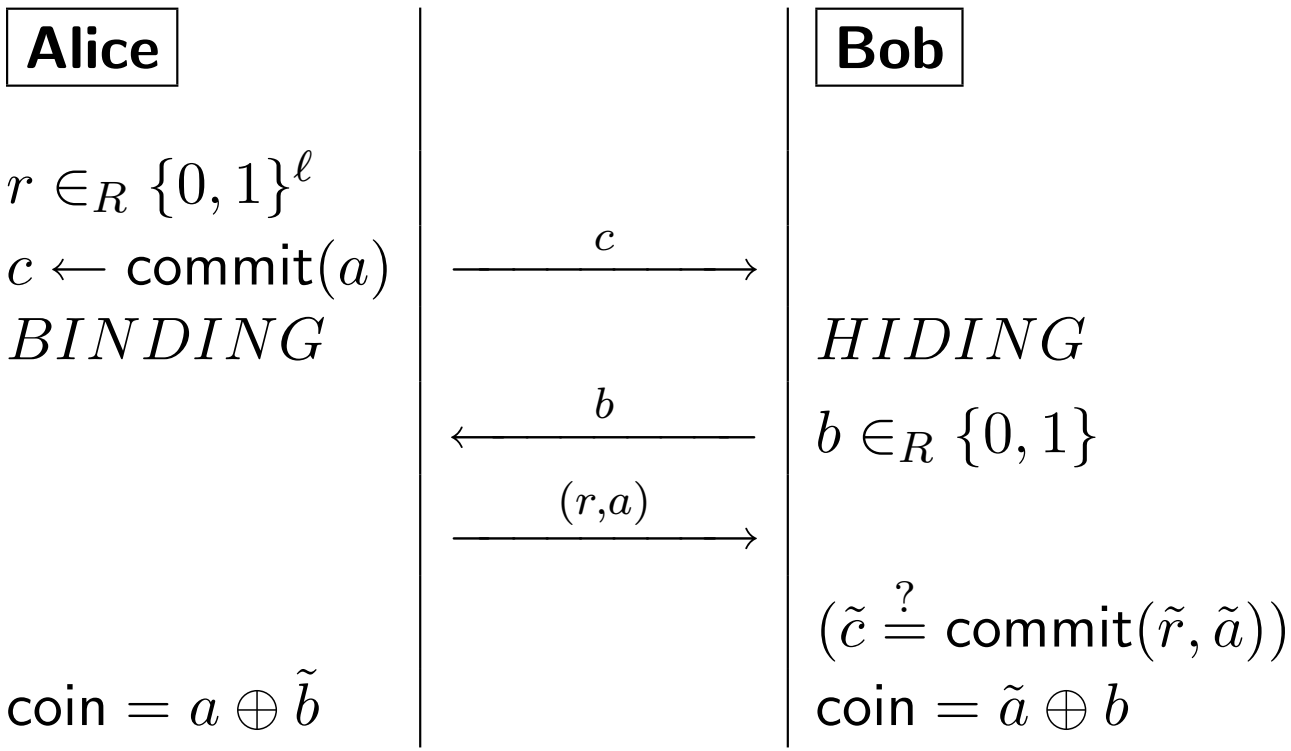
Coin Flipping by Commitment



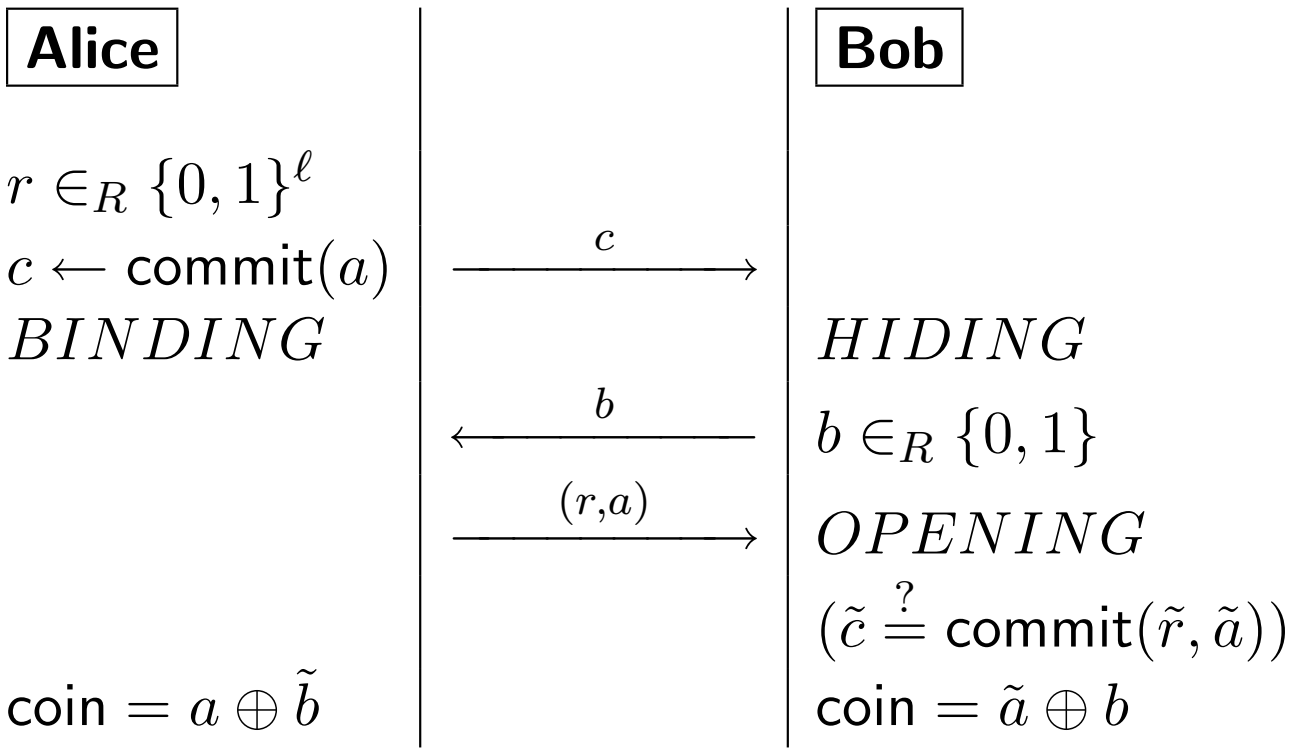
Coin Flipping by Commitment



Coin Flipping by Commitment



Coin Flipping by Commitment



Entity authentication

Identification Protocol

An interactive process between a claimant and a verifier for establishing the correctness of a claimed identity.

Alice the Claimant		**Bob the Guardian**
	Knock,knock! $\longrightarrow$	
	$\longleftarrow$ Who's there?	
	It's me Alice! $\longrightarrow$	
	$\longleftarrow$ Evidence please?	
	Sesame! $\longrightarrow$	Lookup$\{(Id, f(pwd))\}$

- Unilateral or mutual identification.
- Based on *knowledge*, bearer instruments, or biometrics, or combined.
- Normally the first stage of security association set up.
- Giving away the secret key, total release of secret information

Lamport's one-time passwords

"Winding up" a oneway function

$w_0 \overset{f}{\longmapsto} f^t(w_t)$

$w_t \overset{f}{\longmapsto} w_{t-1} \overset{f}{\longmapsto} w_{t-2} \overset{f}{\longmapsto} \ldots \overset{f}{\longmapsto} w_1 \overset{f}{\longmapsto} w_0$

Start with w_0 and show w_1.

In general, the Verifier checks $(w_t \overset{?}{=} f(w_{t+1}))$ and stores w_{t+1} for next session.

Some Notions of Entity Identification (Fiat-Shamir)

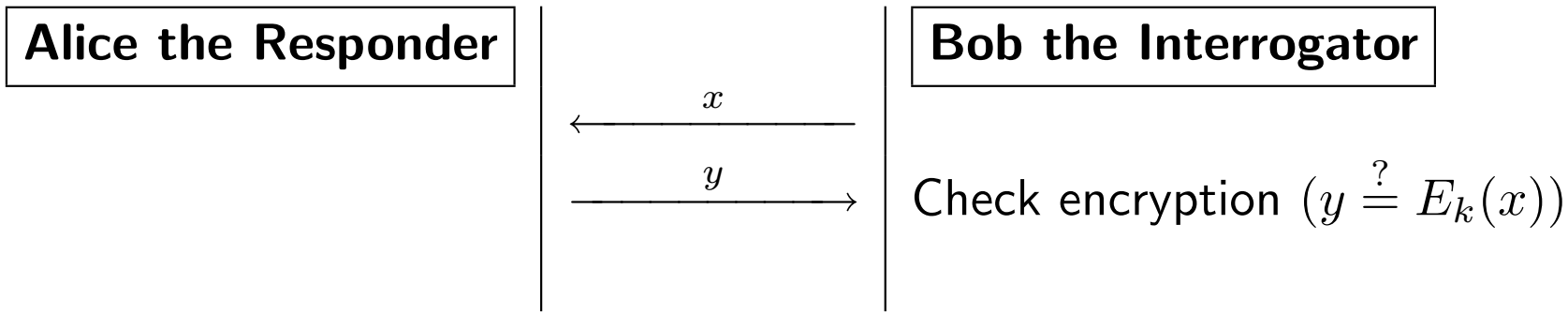
Authentication "Alice can show to Bob that she is Alice"

Identification 'Alice can show to Bob that she is Alice, but Bob cannot show to Charles that he is Alice."

Non-repudiation "Alice can show to Bob that she is Alice, but Bob cannot even show to himself that he is Alice."

Identification Friend or Foe Challenge—Response Protocol

Radar signal active response to identify the approaching aircraft.



- Also in vehicle identification in toll road payments; RFID.
- 'Implicit challenge' by shared synchronized non-repeating counter or clock/time enables a one-message protocol.
- Symmetric key distribution.

Identification Friend or Foe Challenge—Response Protocol

Radar signal active response to identify the approaching aircraft.

Alice the Responder		**Bob the Interrogator**
	$\xleftarrow{\quad x \quad}$	
	$\xrightarrow{\quad y \quad}$	Check encryption $(y \stackrel{?}{=} E_k(x))$
		Check *signature*$(y \stackrel{?}{=} s(x))$

- Also in vehicle identification in toll road payments; RFID.
- 'Implicit challenge' by shared synchronized non-repeating counter or clock/time enables a one-message protocol.
- Symmetric key distribution.
- Diffie&Hellmann's observation: The decryption function is not needed here, use a keyed oneway (hash) function.

Identification Friend or Foe Challenge—Response Protocol

Radar signal active response to identify the approaching aircraft.

Alice the Responder		**Bob the Interrogator**
	$\xleftarrow{\quad x \quad}$	
	$\xrightarrow{\quad y \quad}$	Check encryption $(y \stackrel{?}{=} E_k(x))$
		Check *signature*$(y \stackrel{?}{=} s(x))$

- Also in vehicle identification in toll road payments; RFID.
- 'Implicit challenge' by shared synchronized non-repeating counter or clock/time enables a one-message protocol.
- Symmetric key distribution.
- Diffie&Hellmann's observation: The decryption function is not needed here, use a keyed oneway (hash) function.
- Public-key based interrogator, and where to store the private key.

MIG-in-the-middle

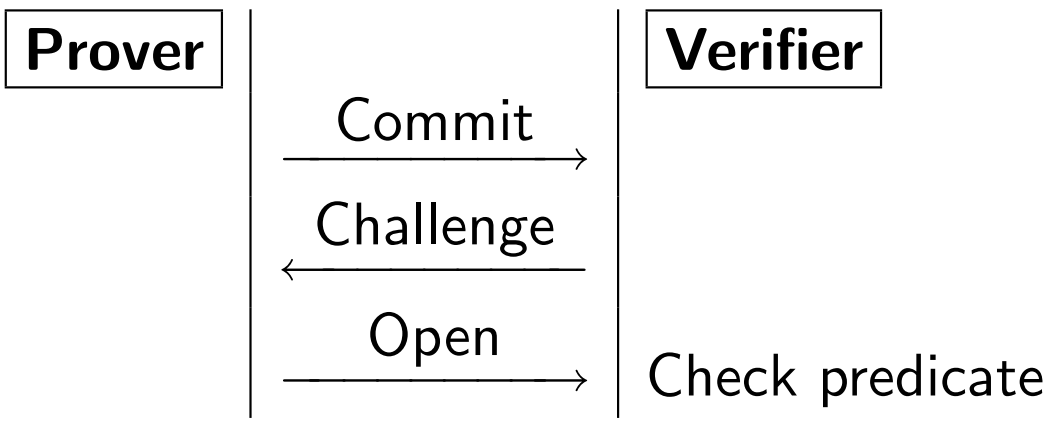
Attack Types

Alice imposter, i.e., Malice claiming to be Alice but is not.
Bob imposter, i.e., Malice acting as Bob but is not.

1. Off-line analysis of information acquired from protocol runs to find the password or key.
2. On-line/active and adaptive trials.
3. Mediating the challenge to an entity that can compute the correct response.

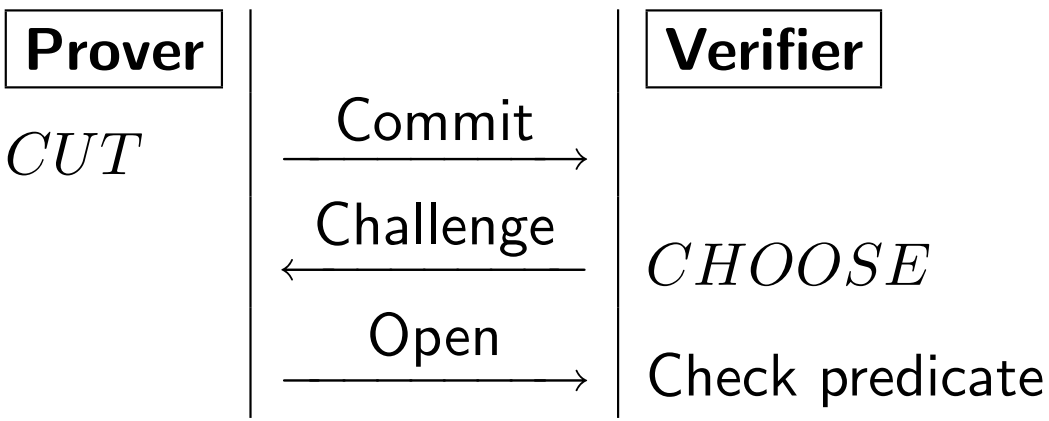
A Template for Interactive Proofs

The idea: Combine Commitment with Challenge—Response Interaction



A Template for Interactive Proofs

The idea: Combine Commitment with Challenge—Response Interaction



Example: Schnorr's Proof of Knowledge Protocol

Peggy the Prover		**Vic the Verifier**
$Y = g^s \pmod p, q \mid p-1$		(p, q, g)
$k \in_R Z_q$		
$c \leftarrow g^k \pmod p$	$\xrightarrow{c}$	
	$\xleftarrow{i}$	$i \in_R \{0, 1\}$
$r \leftarrow k + is \pmod q$	$\xrightarrow{r}$	
		$(g^r \stackrel{?}{=} c \cdot Y^i) \Rightarrow Accept$

Why should Vic be convinced?

- Peggy must be prepared to answer both $r_0 = k$ and $r_1 = k + s$, hence she can compute s.
- If not, too many unknown, so Peggy guesses the challenge i, chooses some r and computes the commitment accordingly $c = g^r / Y^i$.

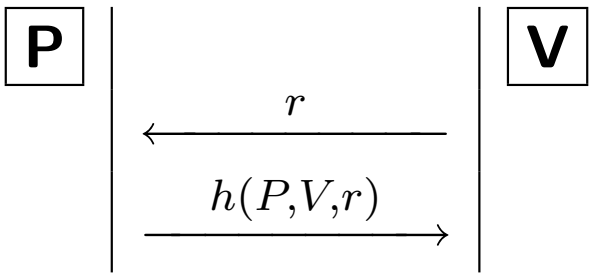
Interactive Proof Properties

Completeness Show $\forall s, k, i : g^r = c \cdot Y^i$

Soundness Prover doesn't know s, see previous slide.

Zeroknowledge Vic may without interaction efficiently *simulate* the protocol transcript (c, i, r) to be indistinguishable from true runs with Peggy.

Zeroknowledge interaction example



Zeroknowledge and Simulation

The paradigm states a zeroknowledge interaction *yields nothing beyond the validity of the assertion*, in the sense that anything that is feasible computable from a zeroknowledge interaction can be feasibly computed from the valid assertion alone.

Simulation Vic may without interaction *simulate protocol runs* by outputting a transcript (c, i, r) that is computationally indistinguishable from interactive runs with Peggy.

Undistinguishability The transcript output is a stochastic variable. The *stochastic variables* (P, V^*) and M_{V^*} cannot be distinguished (within probabilistic polynomial time.)

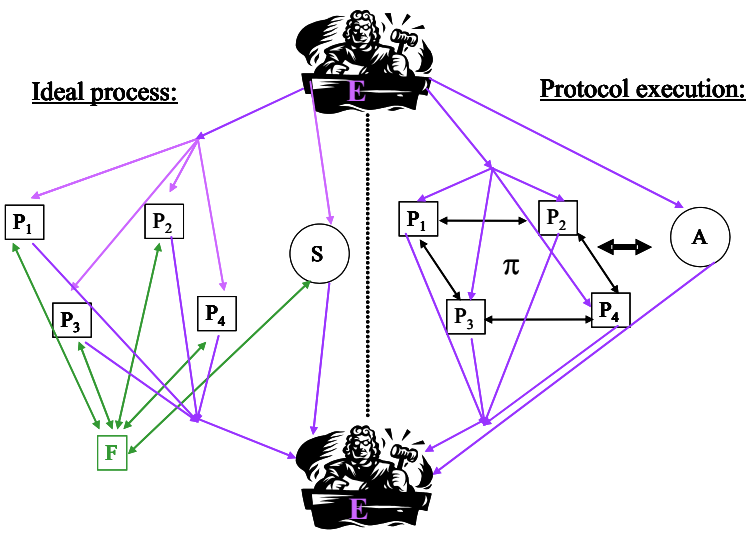
Simulation procedure similar idea to the soundness argument.

Composition of Protocols

- *Sequential* composition of runs of two zero-knowledge protocols maintains the property of zero-knowledge.
- *Parallel* composition of runs does not maintain the zero-knowledge property in general.
- *Concurrent* protocol composition, that is, allowing arbitrary message interleaving of simultaneous runs, does not maintain the zero-knowledge property in general.
- The security of protocol composition in general is a grand challenge to research in cryptoprotocols.
  - It is the reality of communication networks.
  - What are the concerns?
    - Interdependencies of security requirements and the execution environment (definitions!)
    - Bad message interactions.
  - Enables a well-known engineering method: Modular Design.
    - Analyze and understand the component properties.
    - Build systems out of components.

Universal Composability Framework

A protocol is secure for some task if it *emulates an ideal process* where the parties hand their inputs to a *third party*, who locally computes the desired outputs and hands them back to the parties.GMW87



Π securely realizes ideal F.
$\forall A \exists S : E$ cannot distinguish.

Application Areas of Cryptoprotocols

- E-commerce: Fairness, accountability
- On-line Auctions and Gambling: trading and financial markets, shopping
- Health Informatics and Registries: Availability and Privacy
- Secure Multiparty Computations and Distributed Storage (VIFF)
- Electronic Voting: Correctness, acountability, privacy, coercion-freeness...
- Further suggestions? Finance, Content distribution/sharing, Climate Measurements and Monitoring, ...