Inductive Proofs of Computational Security

Anupam Datta (CMU)

Joint work with Arnab Roy, Ante Derek, John Mitchell (Stanford)

Outline

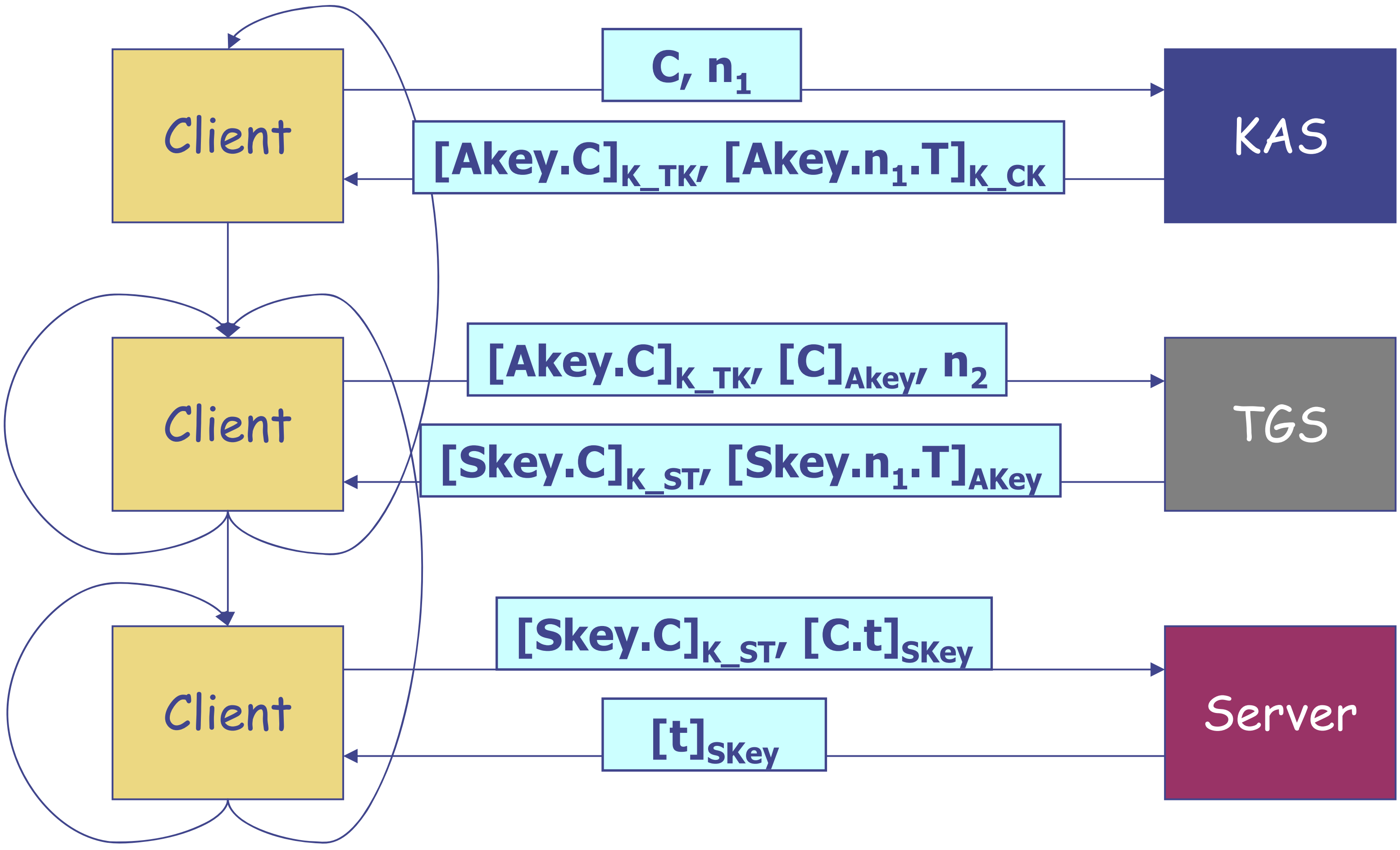
- Network Protocols
  - Partipator Model
  - Adversary Model
- Cryptographic Security
  - Cryptographic Primitives
  - Security Definitions
- Formal Proofs
  - Computational PCL: Syntax, Semantics, Proof System

Protocols

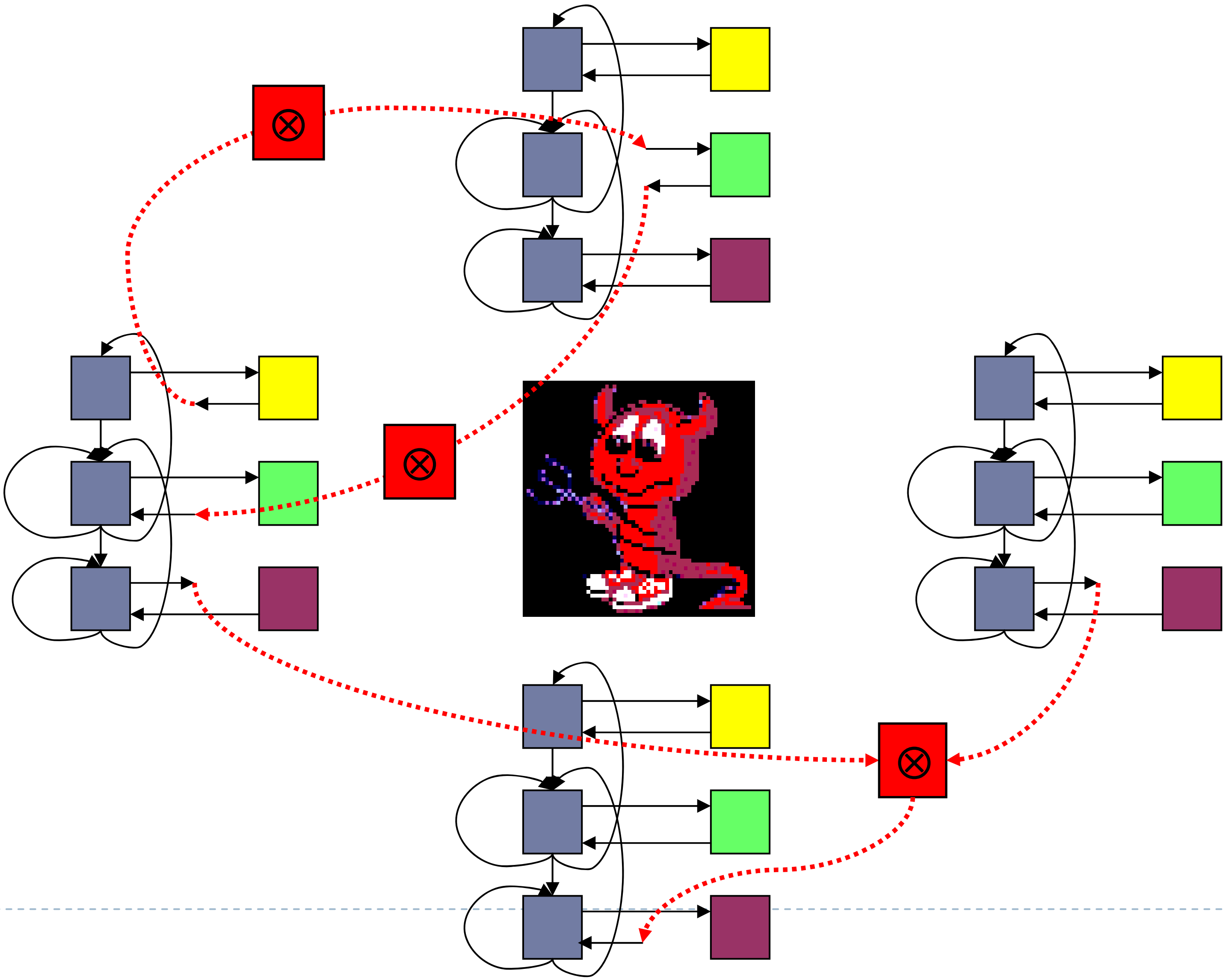
- Distributed Programs
  - Protocol is a fixed set of 'roles' written as programs
  - A 'thread' is an instance of a role being executed by a principal
  - A single principal can execute multiple threads

- Actions in a role
  - Communication: send m; recv m;
  - Pairing, Unpairing: m := pair m_0, m_1; match m as m_0, m_1;
  - Encryption, Decryption: m' := enc m, k; m' := dec m, k;
  - Nonce generation: new m;
  - Pattern matching: match m as m'; …

Kerberos V5

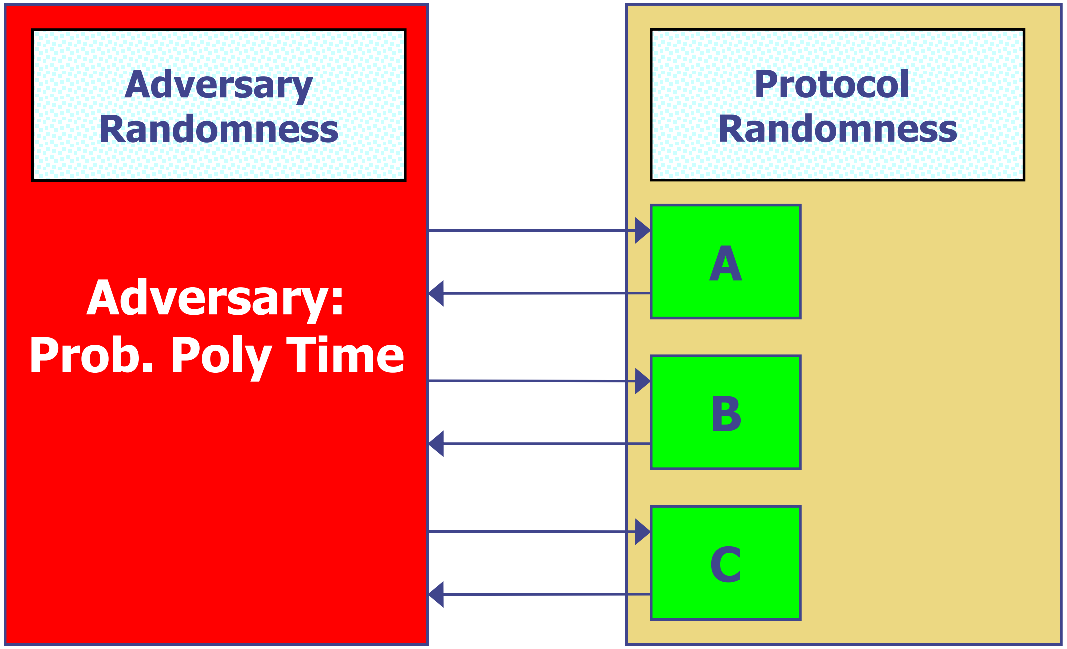

Client
C, n_1
KAS
$[Akey.C]_{K\_TK}$, $[Akey.n_1.T]_{K\_CK}$
Client
$[Akey.C]_{K\_TK}$, $[C]_{Akey}$, n_2
TGS
$[Skey.C]_{K\_ST}$, $[Skey.n_1.T]_{AKey}$
Client
$[Skey.C]_{K\_ST}$, $[C.t]_{SKey}$
Server
$[t]_{SKey}$

Active Computational Adversary



Abstraction:
Protocol Execution Model

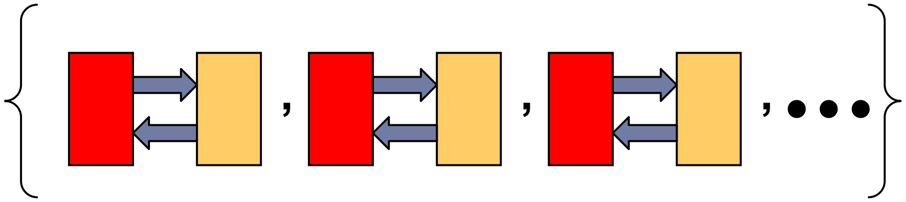
- Network Protocols
  - Partipator Model
  - Adversary Model



Adversary Randomness:
- **Random coin flips for the PPT algorithm**

Protocol Randomness:
- **Key generation**
- **Randomness for encryption, signatures, …**

- Result:
  - Set of computational traces:



One for each adversary and protocol randomness

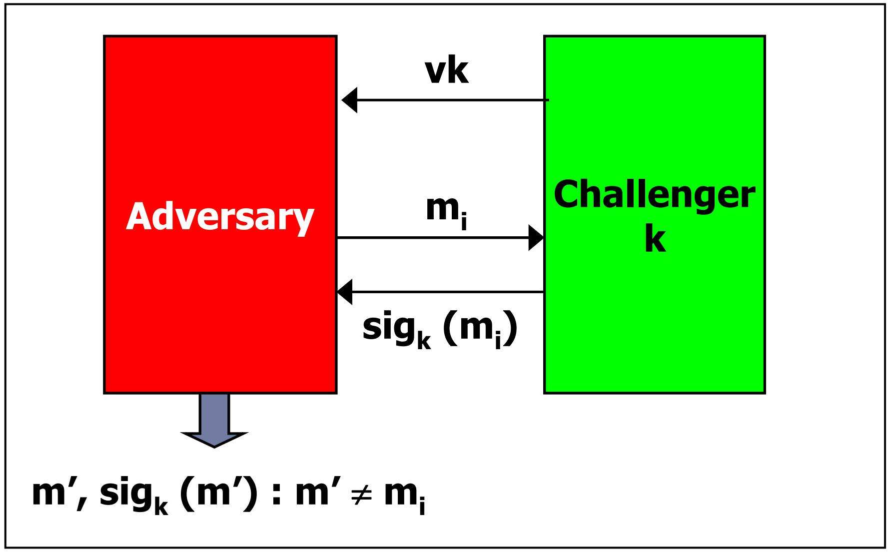
Basic concepts

- Computational complexity
  - Adversary runs in probabilistic polynomial time
    - Polynomial in security parameter
    - Key lengths also polynomial in security parameter
- Acceptable advantage of adversary
  - A negligible function $\nu(x)$: N $\rightarrow$ R is a function that asymptotically decreases faster than the reciprocal of any polynomial in x, i.e.,

$$\forall \text{ polynomial } p.\exists N.\forall n > N.\nu(n) < \frac{1}{p(n)}$$

Example: Security of signatures

Existential Unforgeability under Chosen Message Attack



vk : public verification key
k : private signing key

Advantage(Adversary, η) = Prob[Adversary succeeds for sec. param. η]

A signature scheme is CMA secure if
$\forall$Prob-Polytime A.
Advantage (A, η) is a negligible function of η

- Formal Proofs
  - Syntax, Semantics, Proof System

Computational PCL

- Proof system for direct reasoning
  - Verify (X, sig_Y(m), Y) $\wedge$ Honest (Y) $\Rightarrow$ Sign (Y, m)
  - No explicit use of probabilities and computational complexity
  - No explicit arguments about actions of attackers
- Semantics capture idea that properties hold with high probability against PPT attackers
  - Explicit use of probabilities and computational complexity
  - Probabilistic polynomial time attackers
  - Soundness proofs one time
- Soundness implies result equivalent to security proof by cryptographic reductions

Proof System

DH0 $\mathsf{DHGood}(X, a, x)$, for a of any atomic type, except nonce, *viz.* name or key

DH1 $\mathsf{New}(Y, n) \wedge n \neq x \supset \mathsf{DHGood}(X, n, x)$

DH2 $[\texttt{receive}\ m;]_X\ \mathsf{DHGood}(X, m, x)$

DH3 $[m := \texttt{expg}\ x;]_X\ \mathsf{DHGood}(X, m, x)$

DH4 $\mathsf{DHGood}(X, m_0, x) \wedge \mathsf{DHGood}(X, m_1, x)\ [m := m_0.m_1;]_X\ \mathsf{DHGood}(X, m, x)$

DH5 $\mathsf{DHGood}(X, m, x)\ [m' := \texttt{symenc}\ m, k;]_X\ \mathsf{DHGood}(X, m', x)$

DH6 $\mathsf{DHGood}(X, m, x)\ [m' := \texttt{hash}\ m;]_X\ \mathsf{DHGood}(X, m', x)$

DHGood(X, m_0, x) $\wedge$ DHGood(X, m_1, x)	**[m := pair m_0, m_1;]$_X$**	**DHGood(X, m, x)**
Pre-condition	**Action**	**Post-condition**

Applications

- We proved the following protocols secure in the complexity theoretic model:
  - Kerberos V5 with Symmetric Key initialization
    - Secrecy proofs first time in literature
  - Kerberos V5 with Public Key initialization
    - Secrecy proofs first time in literature
  - IKEv2
    - Proofs first time in literature
- We found an attack on the first phase of Kerberos V5 with Diffie Hellman initialization, proposed an easy fix and proved the resulting protocol secure.

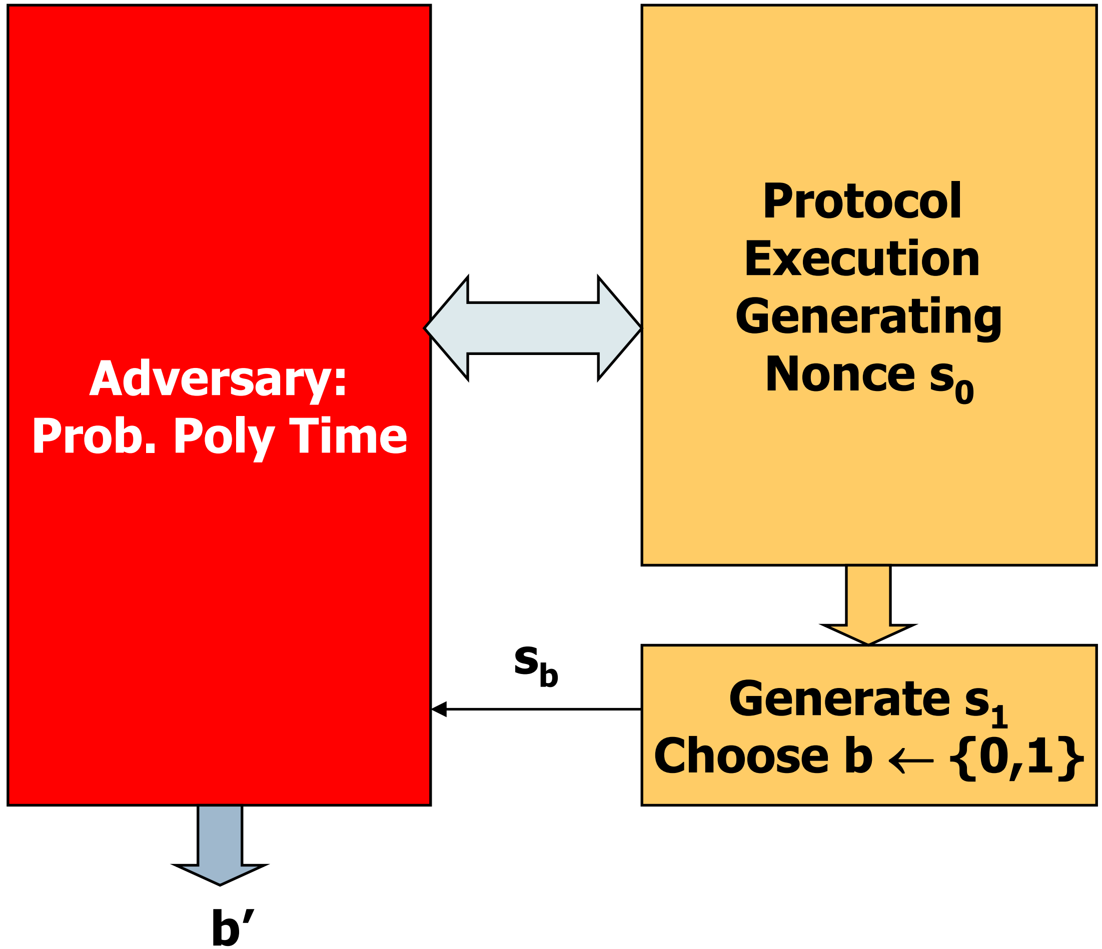
Why our way?

- Why logical methods?
  - Proofs are rigorous but shorter than semantic proofs
  - Carry the same meaning as the semantic proofs
  - Potentially automatable

- Why complexity theoretic model?
  - Protocols are built using cryptographic primitives
  - Cryptographers prove their constructions correct with respect to the complexity theoretic model

Inductive Trace Properties for Computational Security

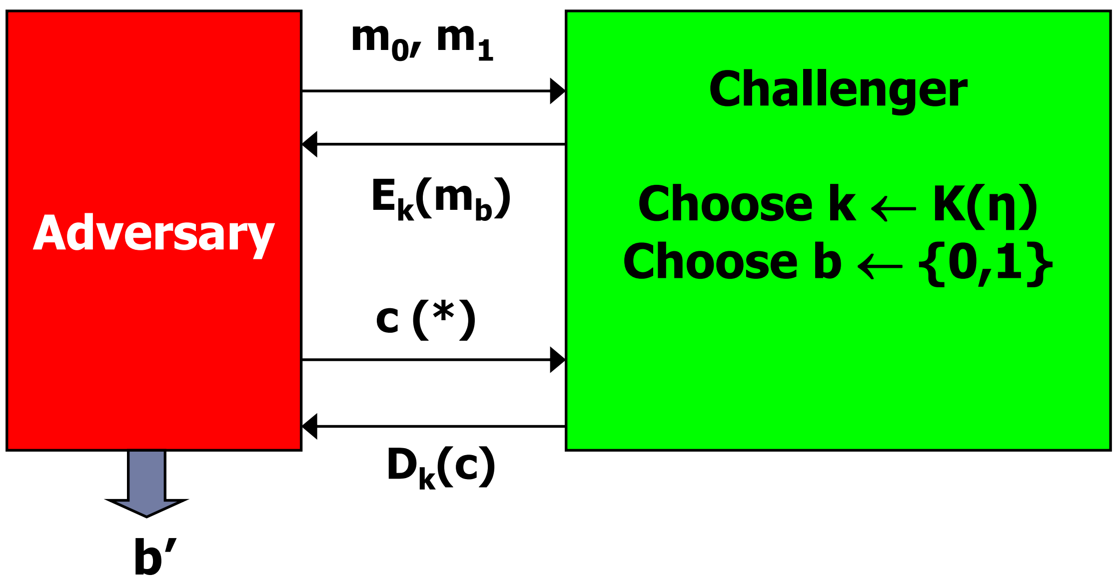
Secrecy Notion: Real or Random Game



$\boxed{Adv(A, \eta) = Pr[b' = b] - 1/2}$

IND-CCA Game

(Key Gen Algo K, Encryption Algo E, Decryption Algo D)
Fix security parameter η



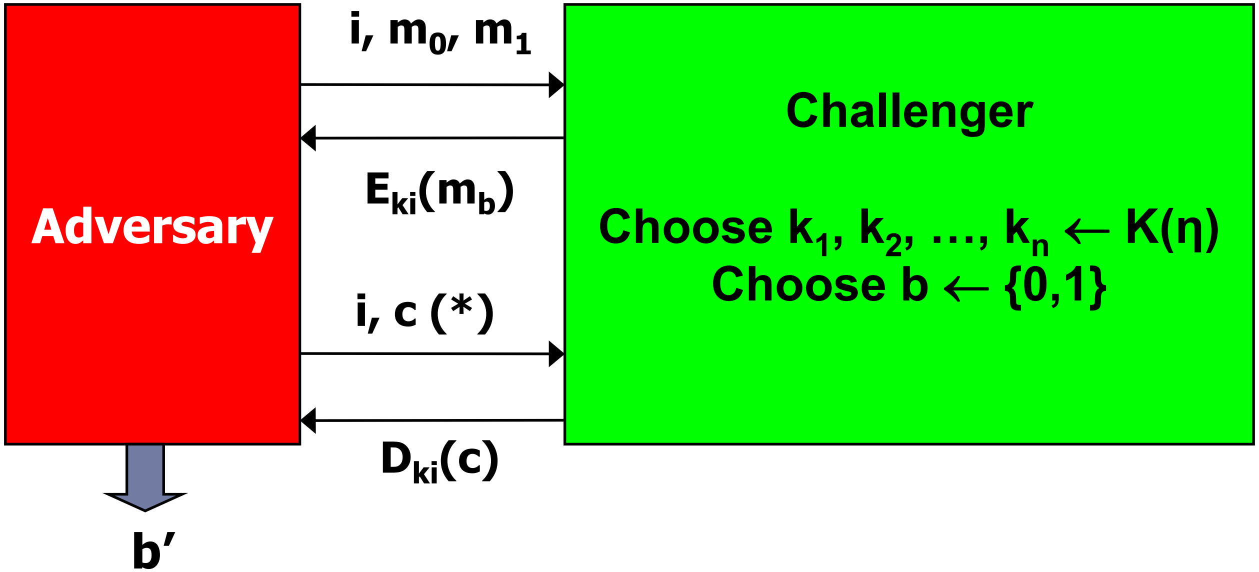
(*): c’s should be different from any encryption response

$Adv(A, \eta) = Pr[b' = b] - ½$

An encryption scheme is IND-CCA secure if
$\forall$ Prob-Polytime A.
Adv (A, η) is a negligible function of η

n-IND-CCA Game

(Key Gen Algo K, Encryption Algo E, Decryption Algo D)
Fix security parameter η



(*): c's should be different from any encryption response

Adv (A, η) = Pr[b′ = b] - ½

An encryption scheme is n-IND-CCA secure if
∀ Prob-Polytime A. Adv (A, η) is a negligible function of η

[BBM00] shows that an encryption scheme is
n-IND-CCA secure ⇔ IND-CCA secure.

Secrecy Notion: Indistinguishability

- Secrecy Property:
  - Indistinguishability for the nonce holds if

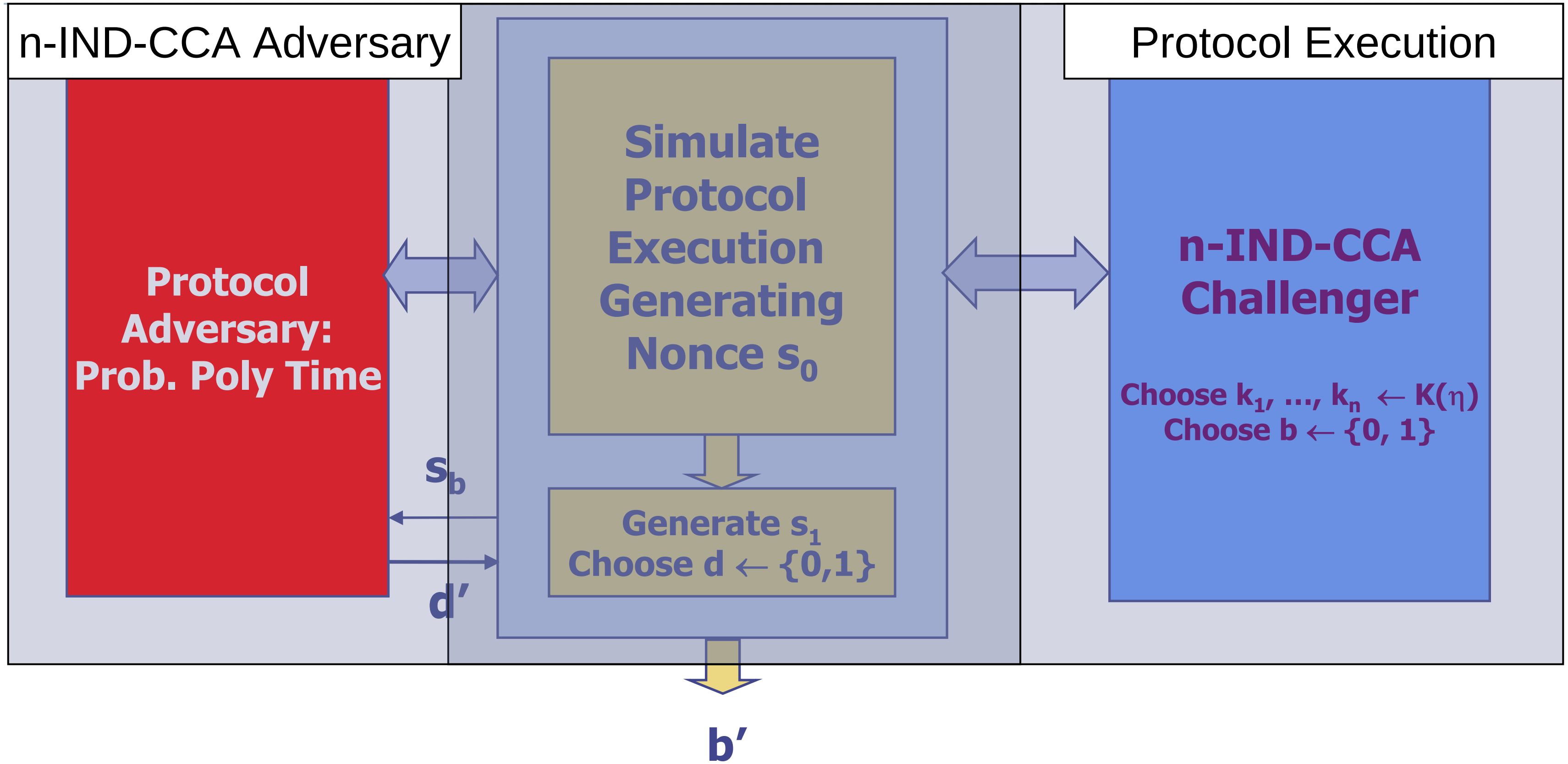
$\forall$Prob-Polytime A.

Adv (A, η) is a negligible function of η

- We want to prove:
  - If the encryption scheme is IND-CCA secure then indistinguishability for the nonce holds if it is protected by a set of keys.

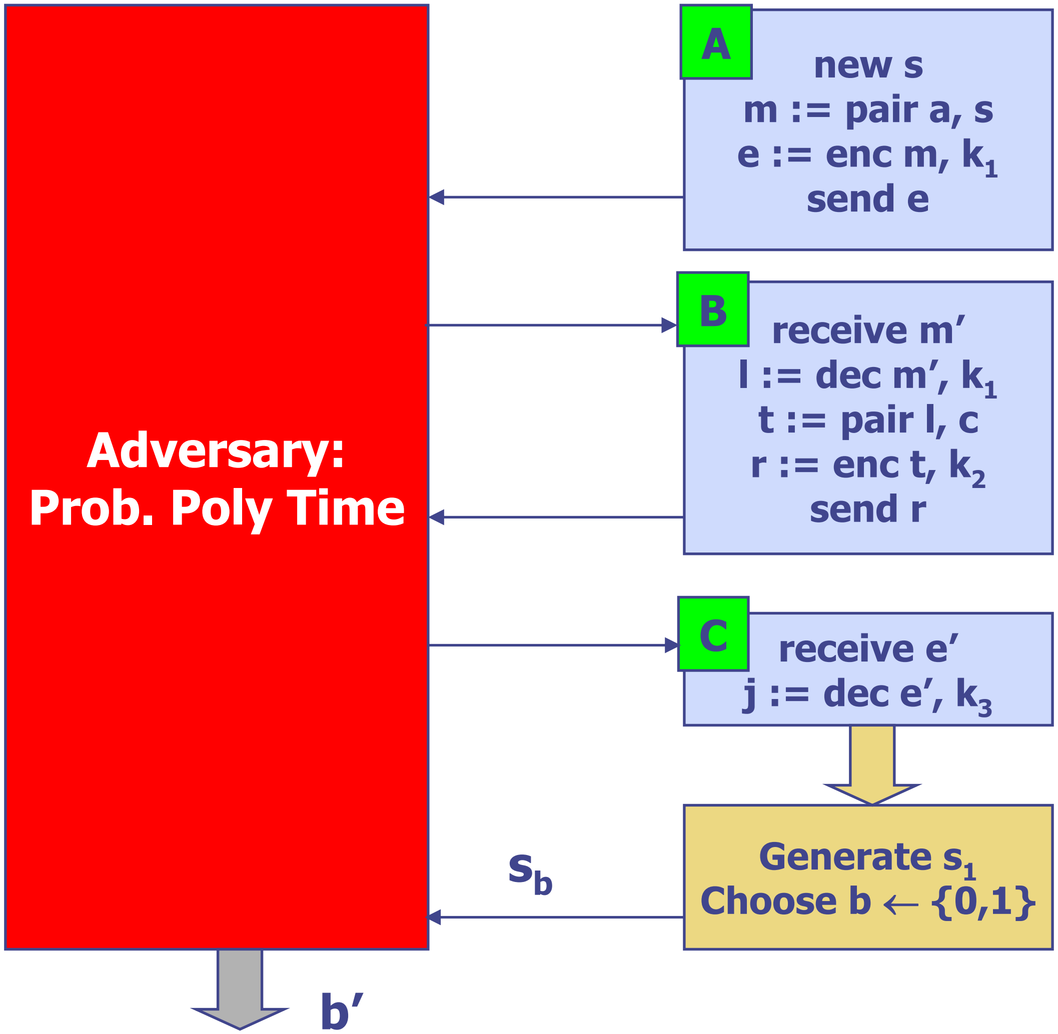
- Proof Strategy:
  - Reduction! – if an adversary can break protocol then there is an adversary which can break CCA (contrapositive)

Reduction

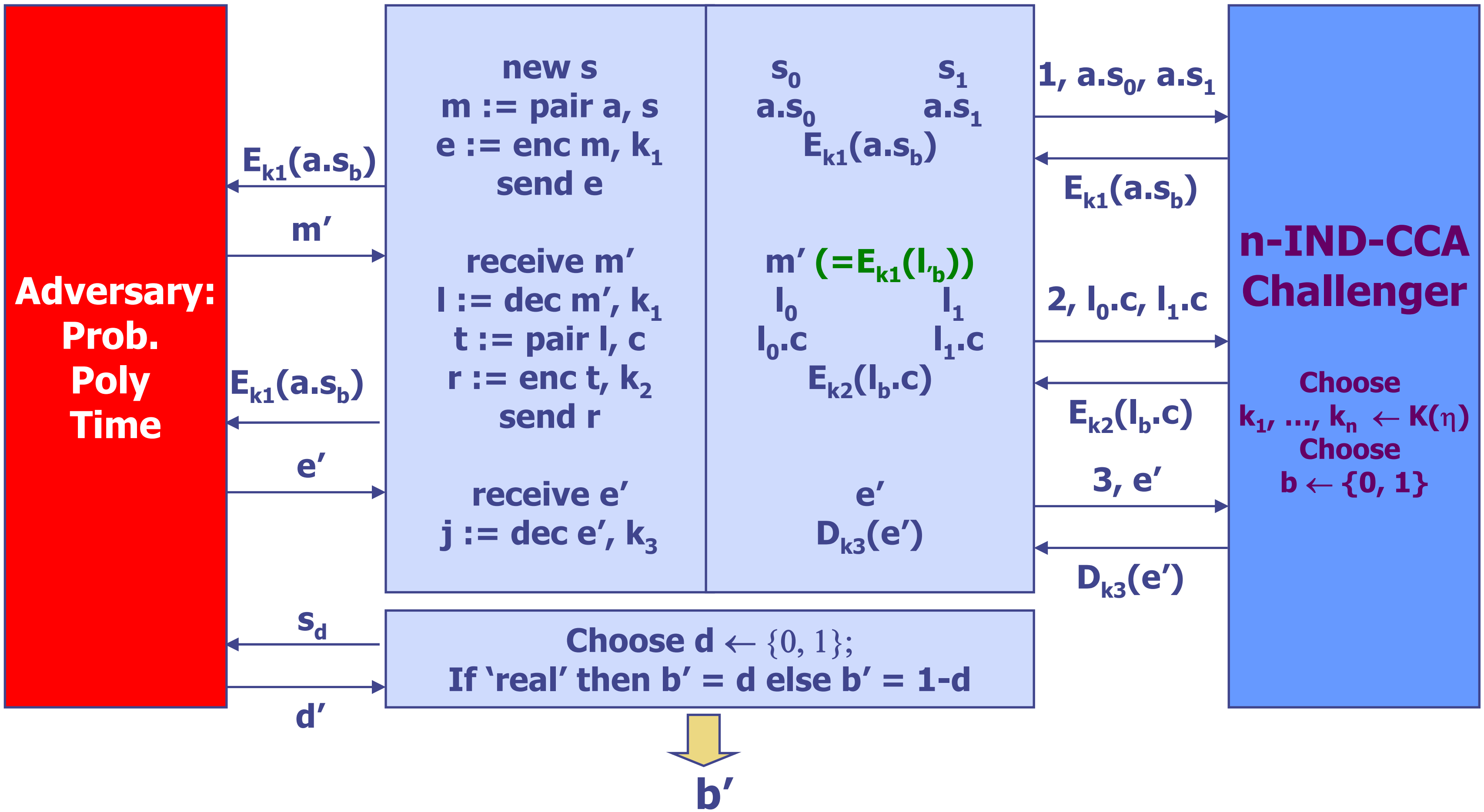


Show that:
If
for nonce indist game Adv (A, η) is non-negligible
Then
for Simulator S, Adv(S, η) against n-IND-CCA game is non-negligible

Protocol example



Reduction



Adv (A, η) for nonce indist game = Adv(S, η) against n-IND-CCA game

Secretive Protocols

- A trace is a *secretive* trace with respect to nonce s and set of keys K if the following properties hold for every thread belonging to honest principals:
  - The thread which generates s, ensures that s is encrypted with a key k in K in any message sent out.
  - Whenever a thread decrypts a message with a key k in K and parses the decryption, it ensures that the results are re-encrypted with some key k' in K in any message sent out.
- A protocol is *secretive* if it overwhelmingly produces secretive traces.
- An inductive property over actions of honest parties
  - Formalization in Computational Protocol Composition Logic.

Relating "Secretive" Protocols to Computational Secrecy

- Theorem:

  If

  - the protocol is "secretive"
  - the nonce-generator is honest
  - the key-holders are honest

  Then

  - the key generated from the nonce satisfies indistinguishability

Do an inductive proof
- *for each protocol*

Proof is by reduction to a multi-party IND-CCA game
– *one time soundness proof*

Proof System to Establish "Secretive" Protocol – "Good" terms

- Proof of construction of good terms is carried out inductively over actions of honest principals

G0 $\mathsf{Good}(X, a, s, \mathcal{K})$, if a is of an atomic type different from nonce or key

G1 $\mathsf{New}(Y, n) \wedge n \neq s \supset \mathsf{Good}(X, n, s, \mathcal{K})$

G2 $[\mathtt{receive}\ \ m;]_X\ \mathsf{Good}(X, m, s, \mathcal{K})$

G3 $\mathsf{Good}(X, m, s, \mathcal{K})\ [\mathsf{a}]_X\ \mathsf{Good}(X, m, s, \mathcal{K})$, for all actions a

G4 $\mathsf{Good}(X, m, s, \mathcal{K})\ [\mathtt{match}\ \ m\ \mathtt{as}\ \ m';]_X\ \mathsf{Good}(X, m', s, \mathcal{K})$

G5 $\mathsf{Good}(X, m_0, s, \mathcal{K}) \wedge \mathsf{Good}(X, m_1, s, \mathcal{K})\ [m := m_0.m_1;]_X\ \mathsf{Good}(X, m, s, \mathcal{K})$

G6 $\mathsf{Good}(X, m, s, \mathcal{K})\ [\mathtt{match}\ \ m\ \mathtt{as}\ \ m_0.m_1;]_X\ \mathsf{Good}(X, m_0, s, \mathcal{K}) \wedge \mathsf{Good}(X, m_1, s, \mathcal{K})$

G7 $\mathsf{Good}(X, m, s, \mathcal{K}) \vee k \in \mathcal{K}\ [m' := \mathtt{symenc}\ \ m, k;]_X\ \mathsf{Good}(X, m', s, \mathcal{K})$

G8 $\mathsf{Good}(X, m, s, \mathcal{K}) \wedge k \notin \mathcal{K}\ [m' := \mathtt{symdec}\ \ m, k;]_X\ \mathsf{Good}(X, m', s, \mathcal{K})$

Proof System to Establish "Secretive" Protocol – Induction

- A protocol is "secretive" if all honest participants send out only "good" terms.

$\forall$roles ρ in protocol Q.
$\forall$segments P in role ρ.

$$\frac{\text{SendGood}(X, s, K)\ [P]_X\ \Phi \supset \text{SendGood}(X, s, K)}{Q \mid\!\!- \Phi \supset \text{Secretive}(s, K)}$$

Example

- Let n be the putative secret and K = $\{k_1, k_2, \ldots\}$
- We want to prove that protocol satisfies Secretive(n, K)
- Consider the following fragment of the protocol:

```
recv e;
t := dec e, k;
match t as A.n';
p := enc n', k;
send p;
```

Case: $k \notin K$

	recv e;	Good(e, n, K)	
			Axiom G2
Good(e, n, K) $\wedge$ k $\notin$ K	t := dec e, k;	Good(t, n, K)	
			Axiom G8
Good(t, n, K)	match t as A.n';	Good(A, n, K) $\wedge$ Good(n', n, K)	
			Axiom G4
Good(n', n, K)	p := enc n', k;	Good(p, n, K)	
			Axiom G6
Good(p, n, K)	send p;	✓	

Case: $k \in K$

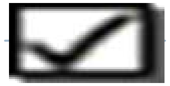
recv e;

t := dec e, k;

match t as A.n';

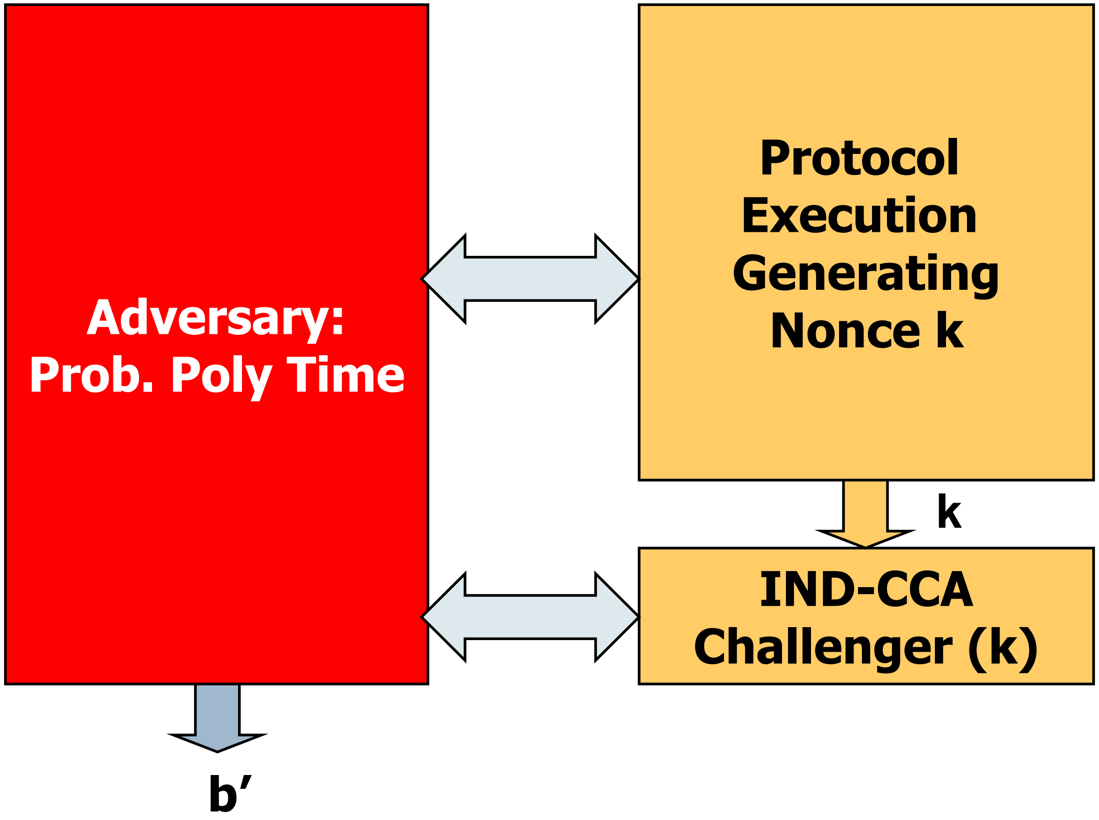
$k \in K$ **p := enc n', k;** **Good(p, n, K)**

Axiom G7

Good(p, n, K) **send p;** 

Good Keys: A weaker notion [DDMW06]

- Key is "good" for a certain purpose
- Intuition: Exchanged key is good for encrypting messages if no attacker can win an appropriate game played with that key.



Relating "Secretive" Protocols to "Good" Keys

- Theorem:

  If

  - the protocol is "secretive"
  - the nonce-generator is honest
  - the nonce may be used as a key
  - the key-holders are honest

  Then

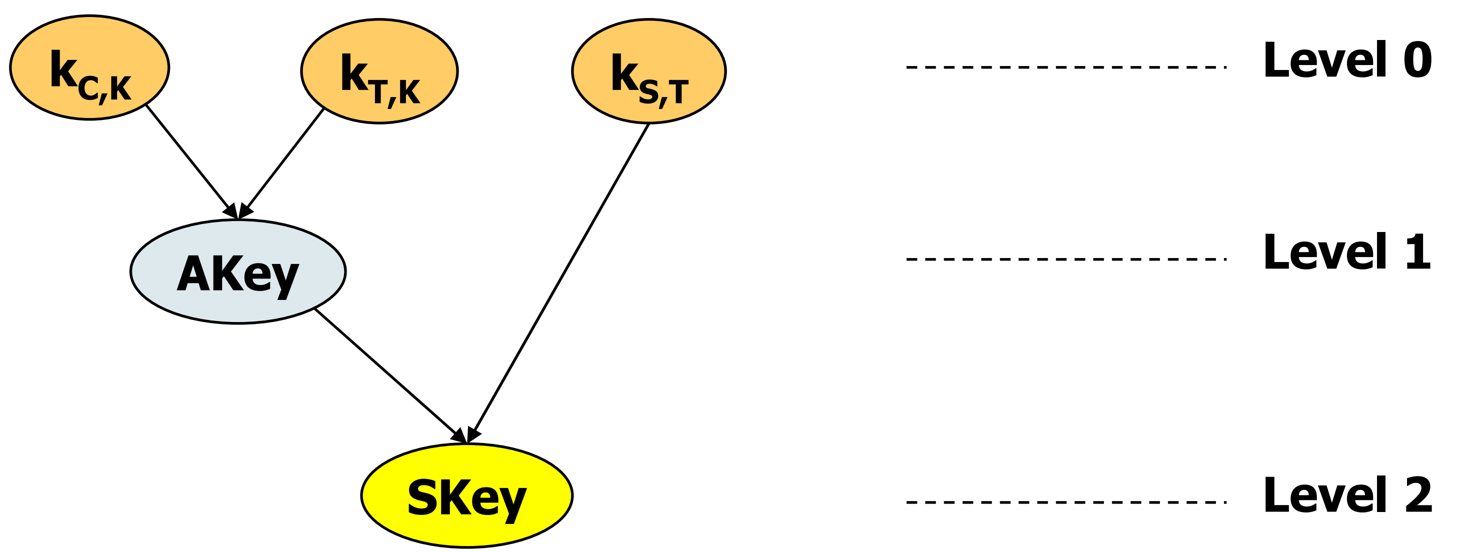
  - the key generated from the nonce is a "good" key

Do an inductive proof
- *for each protocol*

Proof is by reduction to a multi-party IND-CCA game
– one time soundness proof

Key Graphs

- Many interesting protocols establish a hierarchy of keys. For example – Kerberos, IEEE 802.11i



Keys at level i may be used to encrypt keys of level j < i

Some Results

Language	Crypto Assumption	Property
Secret not used as a key	IND-CCA	Secrecy: Indist for level-1
Secret used as a symmetric key	IND-CCA	Secrecy: GoodKey for level-1
Secret not used as a key	IND-CCA	Secrecy: Indist for key DAGs
Secret used as a symmetric key.	IND-CCA	Secrecy: GoodKey for key DAGs
Auth of msg encrypted with the secret.	IND-CPA+INT-CTXT	Authentication for key DAGs

Kerberos V5 results

If Client C completes the protocol with Kerberos Authentication Server K, Ticket Granting Server T and Application Server S then information available to C can be sufficient to guarantee:

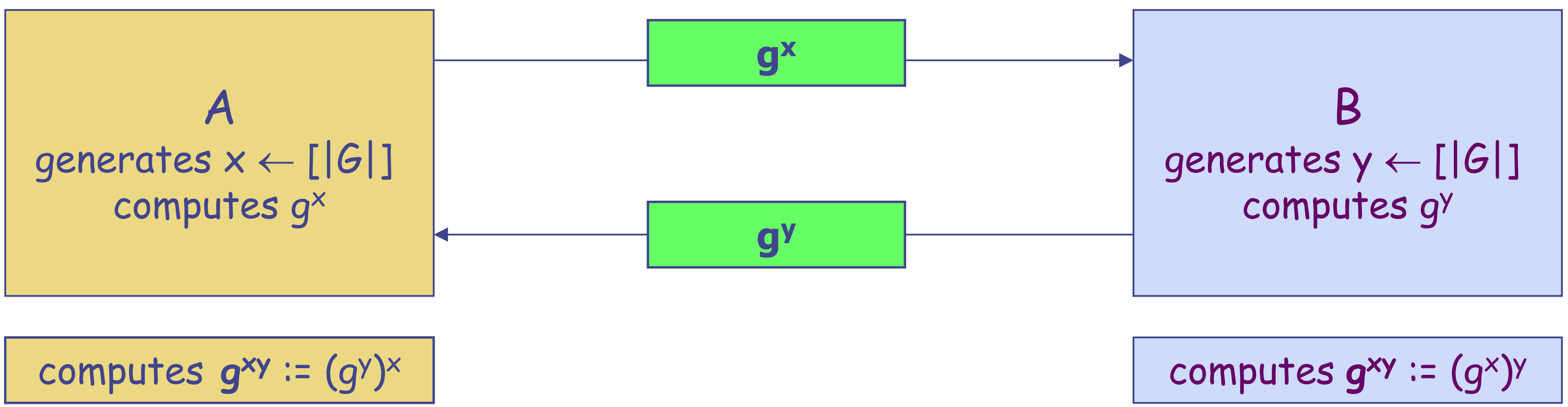
Type	Honesty Assumption	Guarantee
Authenticity	C, K	A message containing a valid ticket granting ticket was indeed sent by K intended for (C, T), with overwhelming probability.
Authenticity	C, K, T	A message containing a valid server ticket was indeed sent by T intended for (C, S), with overwhelming probability.
Secrecy	C, K, T	AKey is a good key for C, K and T.
Secrecy	C, K, T, S	SKey is a good key for C, K, T and S.

- Similar results are proved from the perspective of K, T and S as well
- Theorems proved in [ESORICS2007]

Diffie Hellman

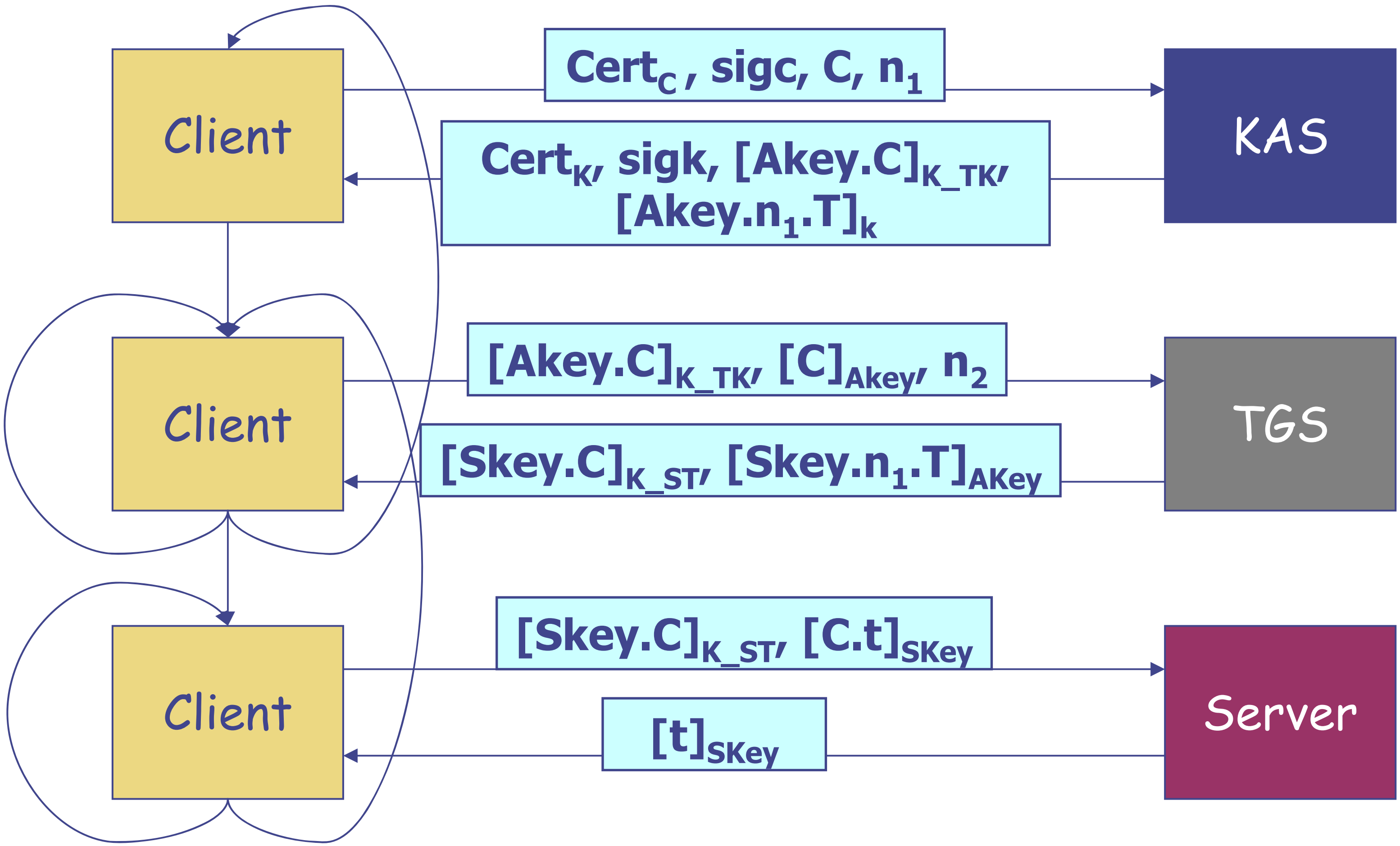
Diffie-Hellman Primer

Fix group G satisfying certain cryptographic properties



g^{xy} is secret to a passive adversary

Kerberos with DHINIT



Is the KAS authenticated after the first phase?

Client C → KAS K: $Cert_C$, Sig_C(“Auth”, Hash(C, T, n_1), m_1, g^x), C, T, n_1

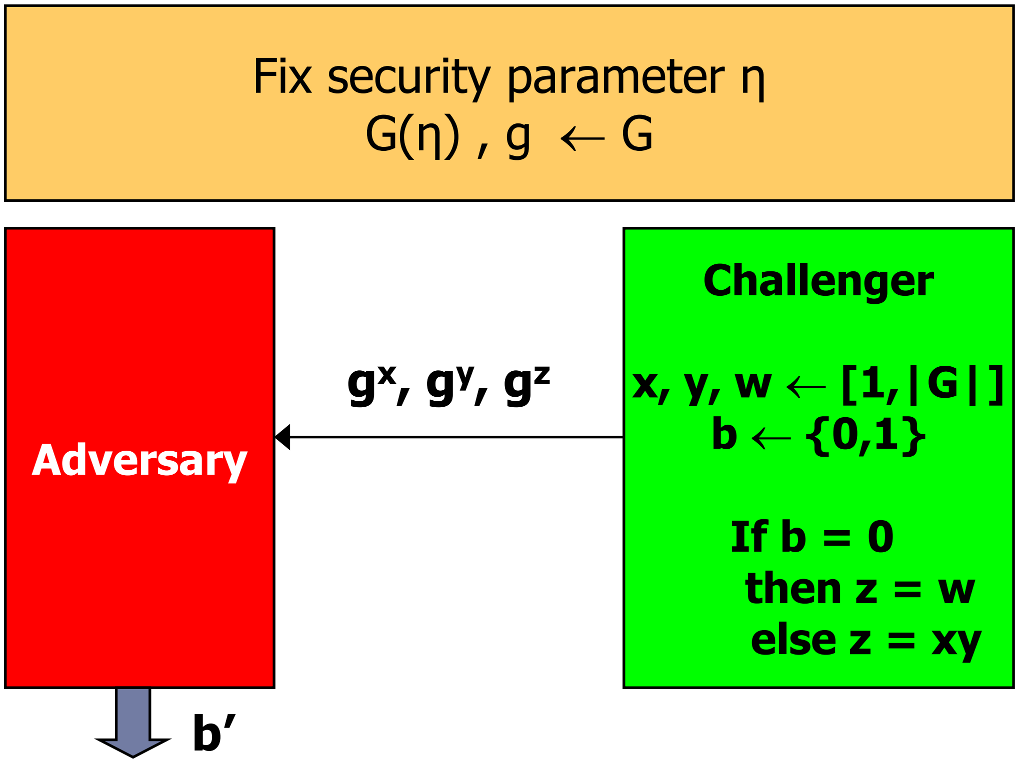
KAS K → Client C: $Cert_K$, Sig_K(“DHKey”, g^y, m_1), TGT(C), Enc-Akey

Client C —1→  —2→ KAS K

KAS K —3→ —4→ Client C

1. $Cert_C$, Sig_C(“Auth”, Hash(C, T, n_1), m_1, g^x), C, T, n_1
2. $Cert_I$, Sig_I(“Auth”, Hash(**I**, T, n_1), m_1, g^x), **I**, T, n_1
3. $Cert_K$, Sig_K(“DHKey”, g^y, m_1), TGT(**I**), Enc-Akey
4. $Cert_K$, Sig_K(“DHKey”, g^y, m_1), TGT(**I**), Enc-Akey

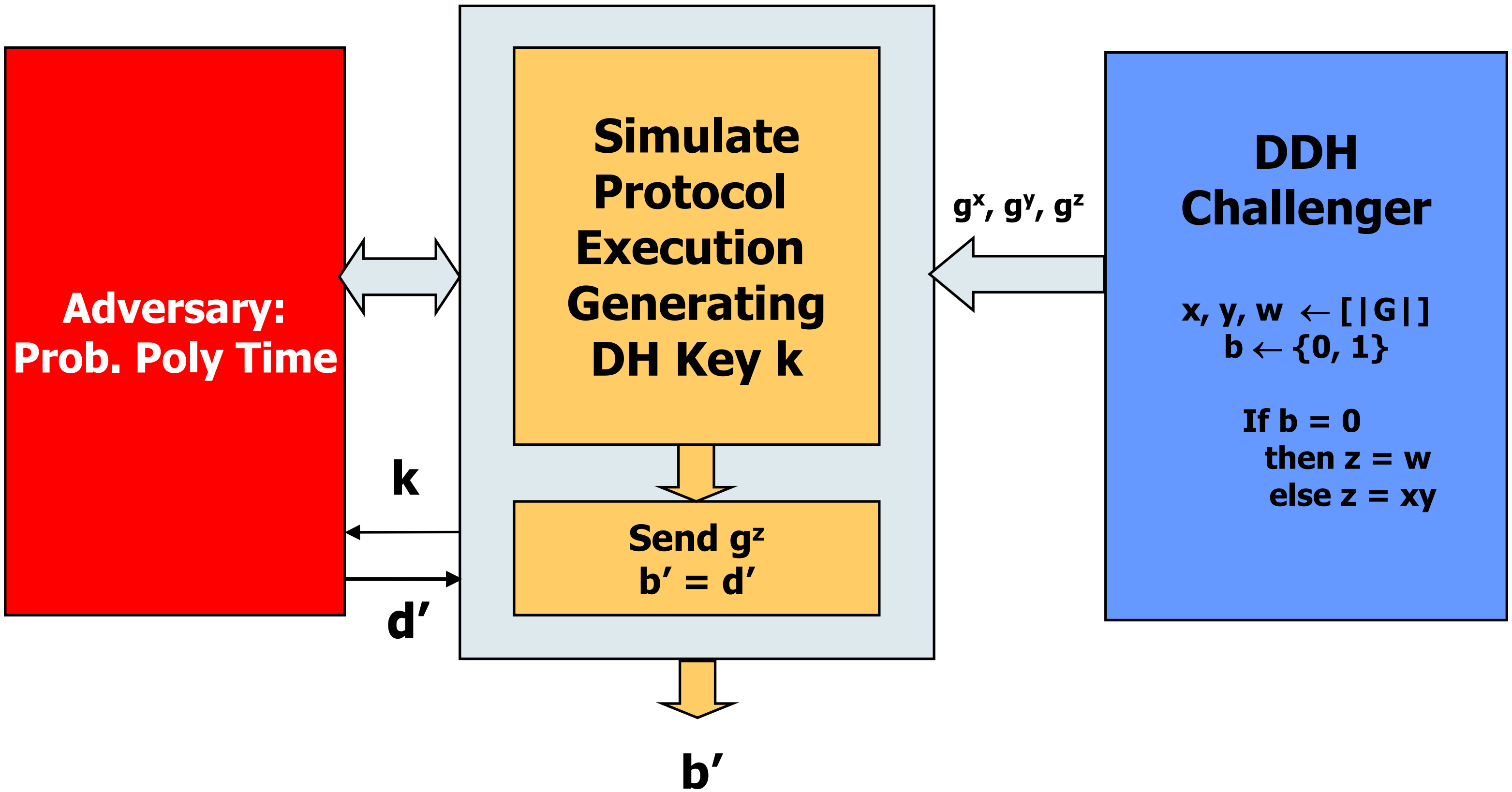
Decisional Diffie Hellman Assumption



Adv (A, η) = Pr[b′ = b] - ½

The DDH assumption holds if
∀Prob-Polytime A.
Adv (A, η) is a negligible function of η

Reduction

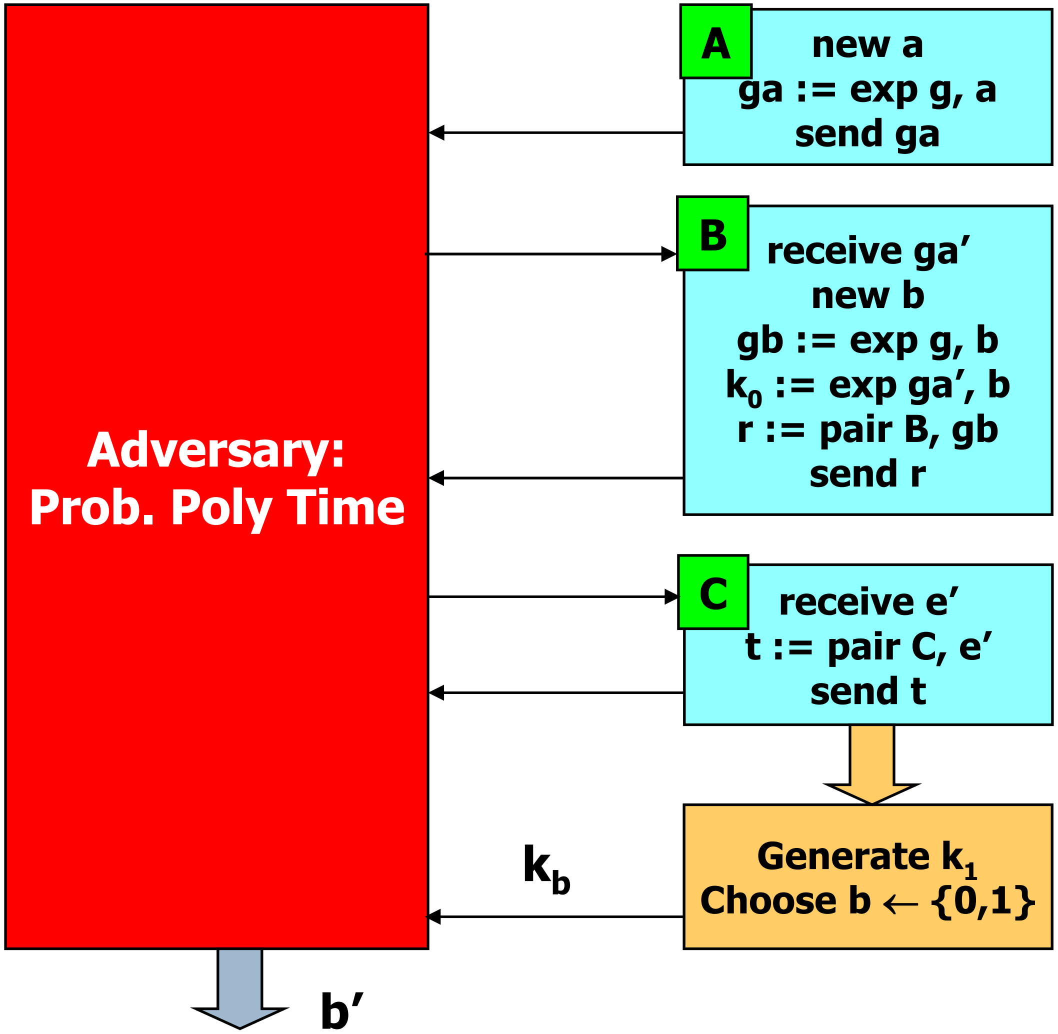


Show that:

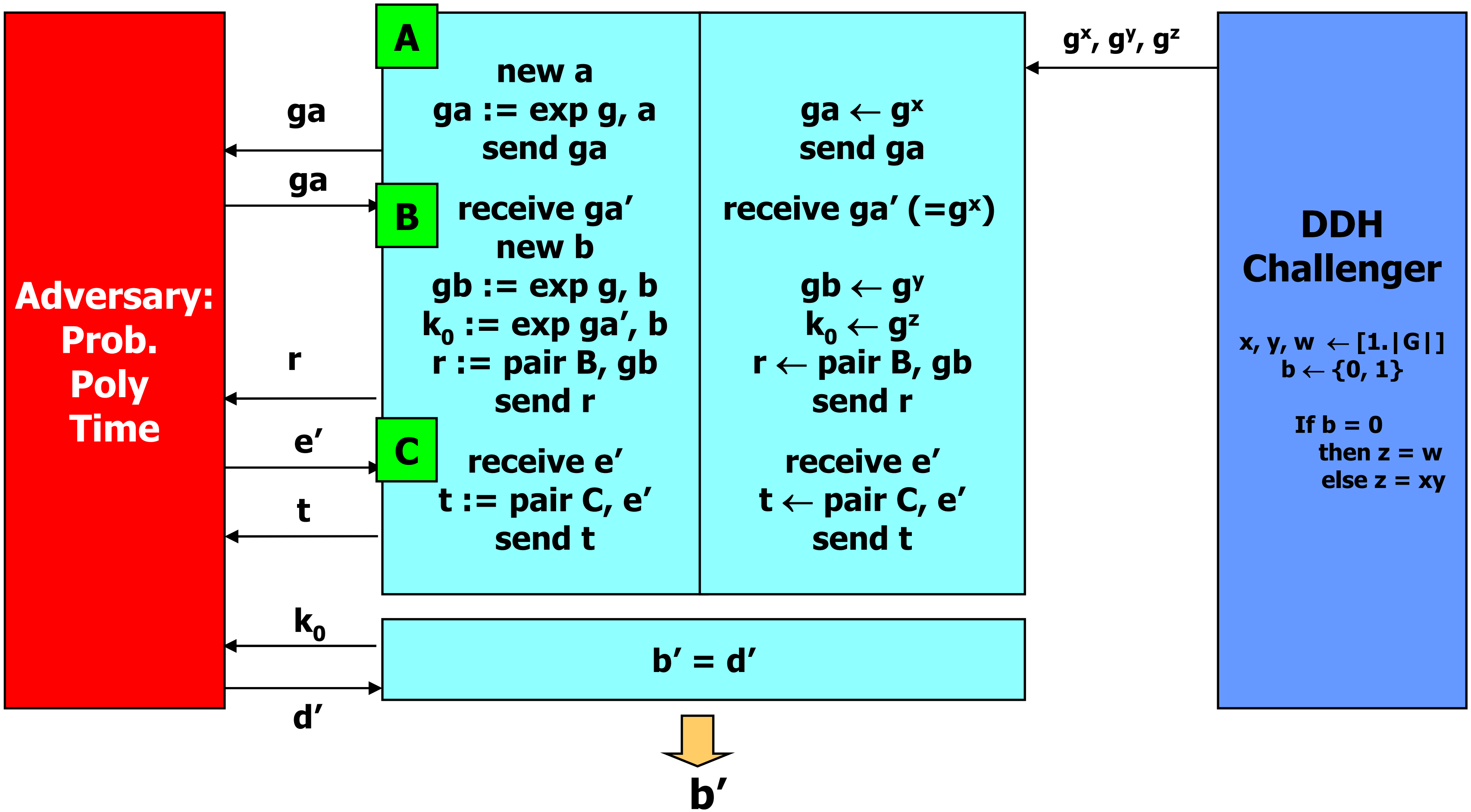
If for key indist game Adv (A, η) is non-negligible

Then for Simulator S, Adv(S, η) against DDH game is non-negligible

Protocol example



Reduction



Adv (A, η) for DH-key indist game = Adv(S, η) against DDH game

DHStrongSecretive Property

- A trace is a *DHStrongSecretive* trace with respect to (x, y) if the following properties hold for every thread belonging to honest principals if,
  - the thread which generates x ensures that it appears only exponentiated as g^x in any message sent out. Similarly for y.
  - the generators of x, y only use each other's DH exponentials to generate the key.
- A protocol is *DHStrongSecretive* if it overwhelmingly produces DHStrongSecretive traces.
- An inductive property over actions of honest parties
  - Formalization in Computational Protocol Composition Logic.

Relating "DHStrongSecretive" Protocols to Computational Secrecy

- Theorem:

  If

  - the protocol is (x,y)-DHStrongSecretive
  - the x, y generators are honest

  Then

  - the key generated from g^{xy} satisfies key indistinguishability

Inductive property of protocol

Proof is by reduction to a DDH game *– one time soundness proof*

Some Results

Language	Crypto Assumption	Property
Secret not used as a key	DDH	Secrecy: Indist
Secret used as a symmetric key	DDH+IND-CPA/CCA	Secrecy: GoodKey for DHStrongSecretive
Secret used as a symmetric key	DDH+INT-CTXT	Authentication for DHStrongSecretive
Secret used as a symmetric key	CDH+RO+INT-CTXT	Authentication for DHSecretive
Secret used to protect other secrets	DDH+IND-CCA	Secrecy of keys protected by DHKey
…	…	… so on

Axioms to prove DH-“safety”

DH0	$\mathsf{DHGood}(X, a, x)$, for a of any atomic type, except nonce, *viz.* name or key
DH1	$\mathsf{New}(Y, n) \wedge n \neq x \supset \mathsf{DHGood}(X, n, x)$
DH2	$[\texttt{receive}\ m;]_X\ \mathsf{DHGood}(X, m, x)$
DH3	$[m := \texttt{expg}\ x;]_X\ \mathsf{DHGood}(X, m, x)$
DH4	$\mathsf{DHGood}(X, m_0, x) \wedge \mathsf{DHGood}(X, m_1, x)\ [m := m_0.m_1;]_X\ \mathsf{DHGood}(X, m, x)$
DH5	$\mathsf{DHGood}(X, m, x)\ [m' := \texttt{symenc}\ m, k;]_X\ \mathsf{DHGood}(X, m', x)$
DH6	$\mathsf{DHGood}(X, m, x)\ [m' := \texttt{hash}\ m;]_X\ \mathsf{DHGood}(X, m', x)$

$$\underbrace{\boxed{\mathbf{DHGood}(X, m_0, x) \wedge \mathbf{DHGood}(X, m_1, x)}}_{\textbf{Pre-condition}}\quad \underbrace{\boxed{[m := \mathbf{pair}\ m_0, m_1;]_X}}_{\textbf{Action}}\quad \underbrace{\boxed{\mathbf{DHGood}(X, m, x)}}_{\textbf{Post-condition}}$$

Kerberos DHINIT Results

- If Client C completes the protocol with Kerberos Authentication Server K, Ticket Granting Server T and Application Server S then information available to C can be sufficient to guarantee:

Type	Honesty Assumption	Guarantee
Authenticity	C, K	A message containing a valid ticket granting ticket was indeed sent by K intended for (C, T), with overwhelming probability.
Authenticity	C, K, T	A message containing a valid server ticket was indeed sent by T intended for (C, S), with overwhelming probability.
Secrecy	C, K, T	AKey is a good key for C, K and T.
Secrecy	C, K, T, S	SKey is a good key for C, K, T and S.

- Similar results are proved from the perspective of K, T and S as well
- Theorems proved in [TGC2007]

IKEv2 Results

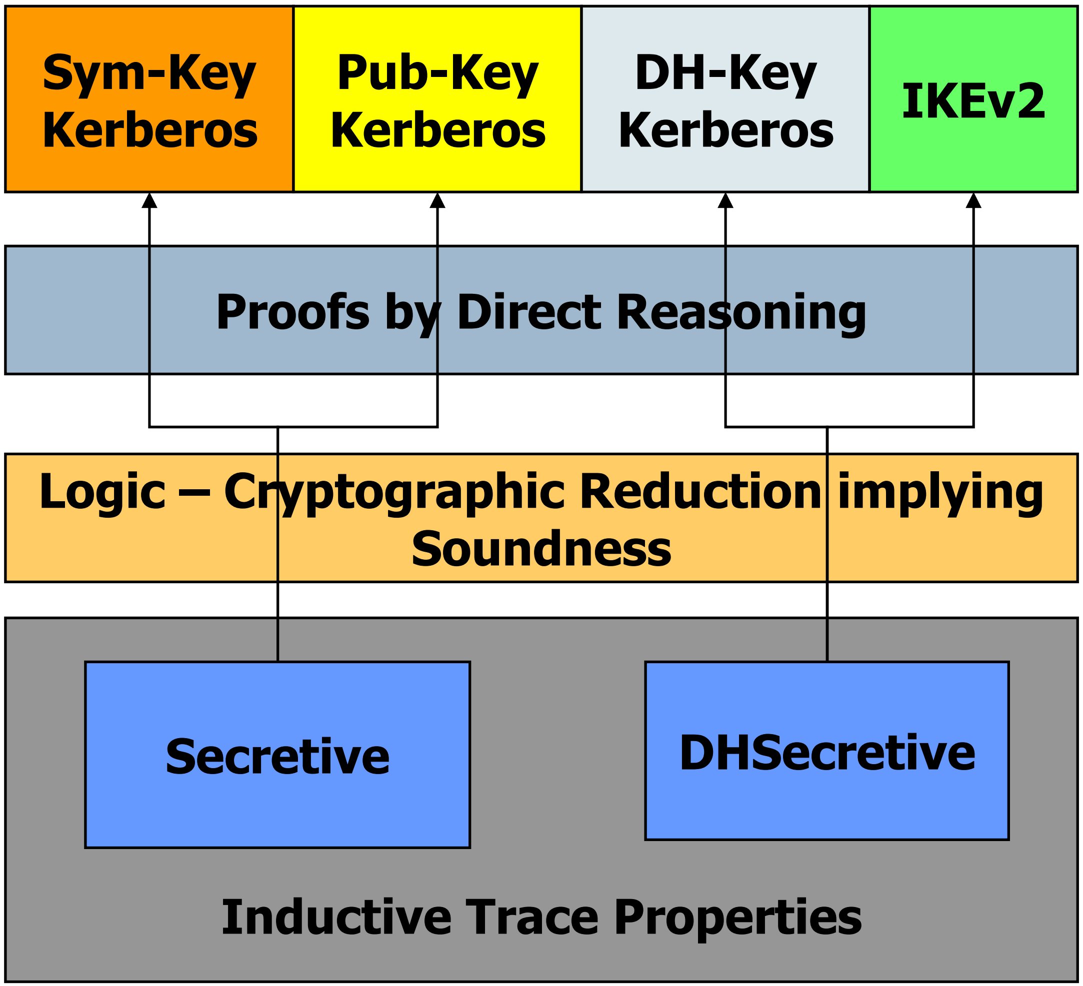
- IKEv2 is a protocol used to negotiate keys at the beginning of an IPsec session.
- If Initiator I completes the protocol with Responder R then I can infer the following guarantees:

Type	Honesty Assumption	Guarantee
Authenticity	I, R	Intended messages were indeed received and sent by R with overwhelming probability.
Secrecy	I, R	The exchanged keys are good keys for I and R.

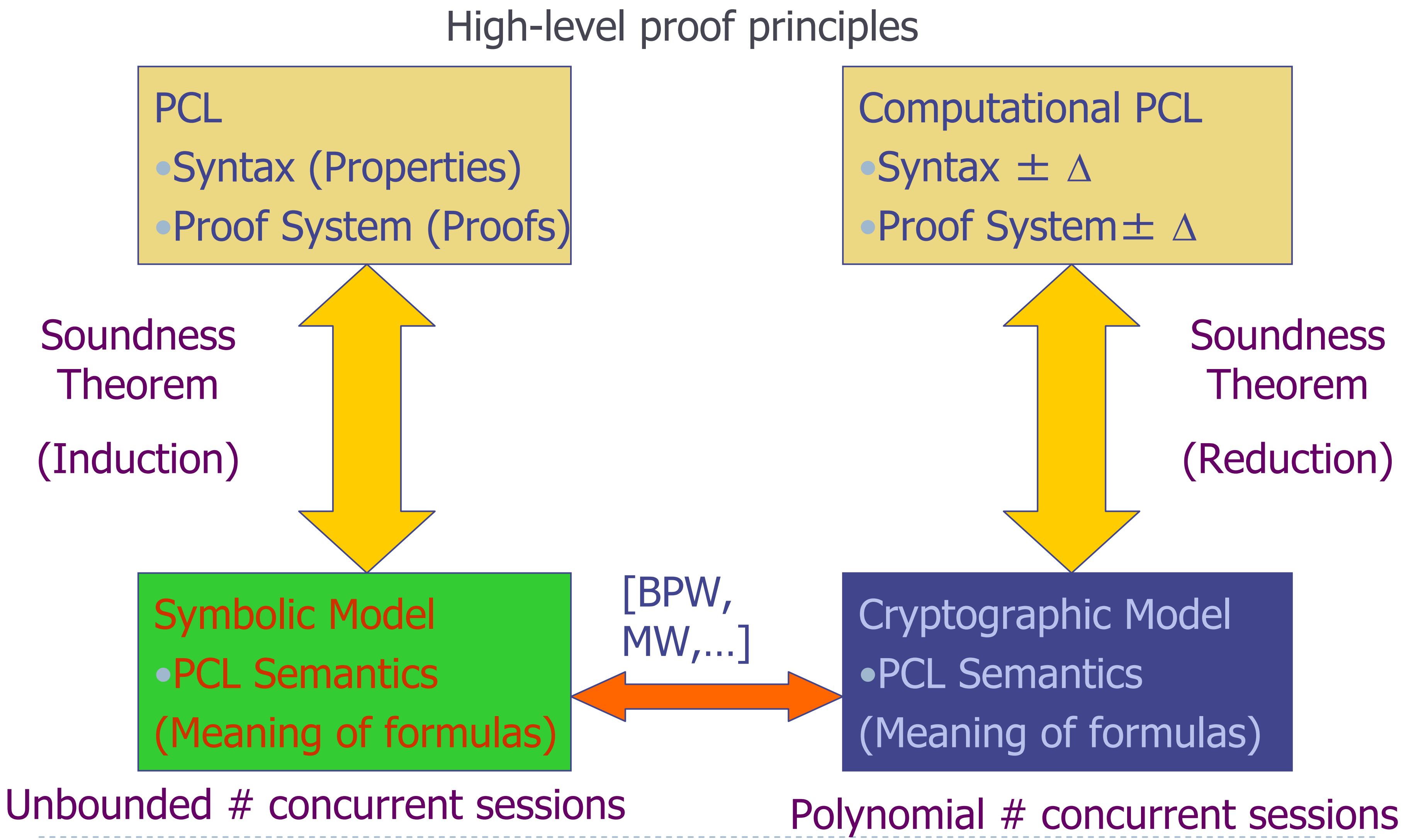
- Similar results are proved from the perspective of R as well

Conclusion

Summary of Results



PCL: Big Picture



Thanks!

Questions?