Analyzing SET with Inductive Method

Theorem Proving for Protocol Analysis

[Paulson]

- Prove correctness instead of looking for bugs
  - Use higher-order logic to reason about all possible protocol executions
- No finite bounds
  - Any number of interleaved runs
  - Algebraic theory of messages
  - No finite bounds on the attacker
- Mechanized proofs
  - Automated tools can fill in parts of proofs

Inductive Method

- Define the set of protocol traces
  - Given a protocol, a trace is one possible sequence of events, including attacker actions
- Prove correctness by induction
  - For every state in every trace, prove that no security condition fails
    - Works for safety properties only
  - Induction is on the length of the trace

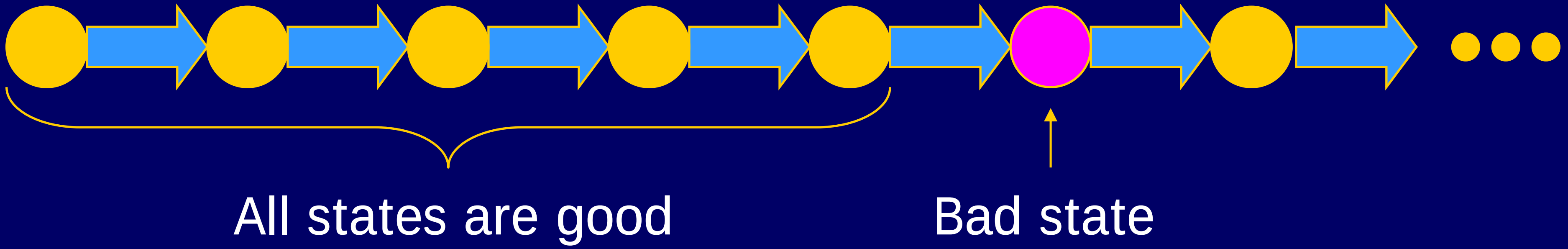
Two Forms of Induction

- Usual form for $\forall n \in Nat.\ P(n)$
  - Base case: $P(0)$
  - Induction step: $P(x) \Rightarrow P(x+1)$
  - Conclusion: $\forall n \in Nat.\ P(n)$
- Minimal counterexample form
  - Assume: $\exists x\ [\ \neg P(x) \wedge \forall y<x.\ P(y)\ ]$
  - Prove contradiction
  - Conclusion: $\forall n \in Nat.\ P(n)$

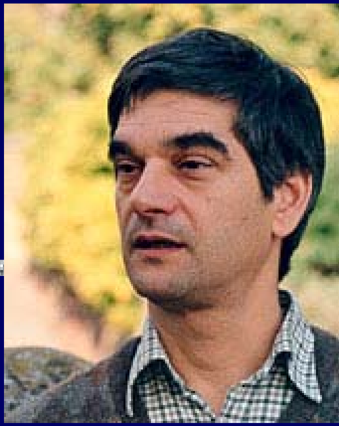
Both equivalent to "the natural numbers are well-ordered"

Induction for Protocol Analysis

- Given a set of traces, choose shortest sequence to a bad state
  - Bad state = state in which an invariant is violated
  - Assume all steps before that are OK
  - Derive contradiction
    - – Consider all possible actions taken at this step



Work by Larry Paulson



- Isabelle theorem prover
  - General tool; security protocols work since 1997
- Many case studies of security protocols
  - Verification of SET protocol (6 papers)
  - Kerberos (3 papers)
  - TLS protocol
  - Yahalom protocol, smart cards, etc

http://www.cl.cam.ac.uk/users/lcp/papers/protocols.html

Isabelle

- Automated support for proof development
  - Higher-order logic
  - Serves as a logical framework
  - Supports ZF set theory & HOL
  - Generic treatment of inference rules
- Powerful simplifier & classical reasoner
- Strong support for inductive definitions

Agents and Messages

$$
\begin{array}{lll}
\mathit{agent}\ \ A, B, \ldots & = & \text{Server} \mid \text{Friend}\ i \mid \text{Spy} \\
\mathit{msg}\ \ X, Y, \ldots & = & \text{Agent}\ A \quad \mid \\
 & & \text{Nonce}\ N \quad \mid \\
 & & \text{Key}\ K \quad \mid \\
 & & \{\, X,\ Y \,\} \quad \mid \\
 & & \text{Crypt}\ (K)\ X
\end{array}
$$

Typed, free term algebra, ...

Protocol Semantics

- "Set of event traces" semantics for protocols
- Operational model for honest agents
  - Similar to pi calculus or protocol composition logic
- Algebraic theory of messages defines attacker
  - Primitive operations: encrypt, decrypt, ...
  - Inductive closure of the intercepted messages under primitive operations defines the set of all messages available to the attacker
- Proofs mechanized using Isabelle/HOL

A Few Definitions

- Traces
  - A protocol is a set of traces
  - A trace is a sequence of events
  - Inductive definition involves implications

    if $ev_1, \ldots, ev_n \in evs$, then add ev' to evs
- Information from a set of messages
  - parts H : parts of messages in H
  - analz H : parts of messages in H that can be learned by attacker
    - Not every message part can be learned by attacker!
  - synth H : messages that can be constructed from H

Protocol Events

- Several types of events
  - A sends B message X
  - A receives X
  - A stores X

$A \rightarrow B \quad \{A, N_A\}_{pk(B)}$

If `ev` is a trace and `Na` is unused, add
`Says A B Crypt(pk B){A,Na}`

$B \rightarrow A \quad \{N_B, N_A\}_{pk(A)}$

If `Says A' B Crypt(pk B){A,X}` $\in$ `ev` and `Nb` is unused, add
`Says B A Crypt(pk A){Nb,X}`

$A \rightarrow B \quad \{N_B\}_{pk(B)}$

If `Says ...{X,Na}...` $\in$ `ev`, add
`Says A B Crypt(pk B){X}`

Attacker Capabilities: Analysis

analz H is what attacker can learn from H

$$X \in H \quad \Rightarrow \quad X \in \text{analz } H$$

$$\{X, Y\} \in \text{analz } H \quad \Rightarrow \quad X \in \text{analz } H$$

$$\{X, Y\} \in \text{analz } H \quad \Rightarrow \quad Y \in \text{analz } H$$

$$\text{Crypt } X\, K \in \text{analz } H$$

$$\& \qquad K^{-1} \in \text{analz } H \quad \Rightarrow \quad X \in \text{analz } H$$

Attacker Capabilities: Synthesis

synth H is what attacker can create from H

infinite set!

$$X \in H \quad \Rightarrow \quad X \in \text{synth } H$$

$$X \in \text{synth } H \;\&\; Y \in \text{synth } H \quad \Rightarrow \quad \{X, Y\} \in \text{synth } H$$

$$X \in \text{synth } H \;\&\; K \in \text{synth } H \quad \Rightarrow \quad \text{Crypt } (K)\, X \in \text{synth } H$$

Equations and Implications

analz(analz H) = analz H

synth(synth H) = synth H

analz(synth H) = analz H $\cup$ synth H

synth(analz H) = ???

But only if keys are atomic

Nonce $N \in$ synth H $\Rightarrow$ Nonce $N \in H$

Crypt (K) $X \in$ synth H $\Rightarrow$ Crypt (K) $X \in H$

or

$X \in$ synth H & $K \in H$

Attacker Events

If $X \in \text{synth}(\text{analz}(\text{spies } evs))$,

add *Says Spy B X*

X is not secret because attacker can construct it from the parts he learned from events *evs* (attacker announces all secrets he learns)

Correctness Conditions

If $\mathit{Says}\ B\ A\ \{N_b, X\}_{pk(A)} \in \mathit{evs}$ &

$\mathit{Says}\ A'\ B\ \{N_b\}_{pk(B)} \in \mathit{evs}$,

Then $\mathit{Says}\ A\ B\ \{N_b\}_{pk(B)} \in \mathit{evs}$

If B thinks he's talking to A,
then A must think she's talking to B

Secure Electronic Transactions (SET)

- Goal: privacy of online credit card transactions
  - Merchant doesn’t learn credit card details
  - Bank (credit card issuer) doesn’t learn what you buy
- Cardholders and merchants must register and receive electronic credentials
  - Proof of identity
  - Evidence of trustworthiness
- Expensive development effort, little deployment

Isabelle verification by

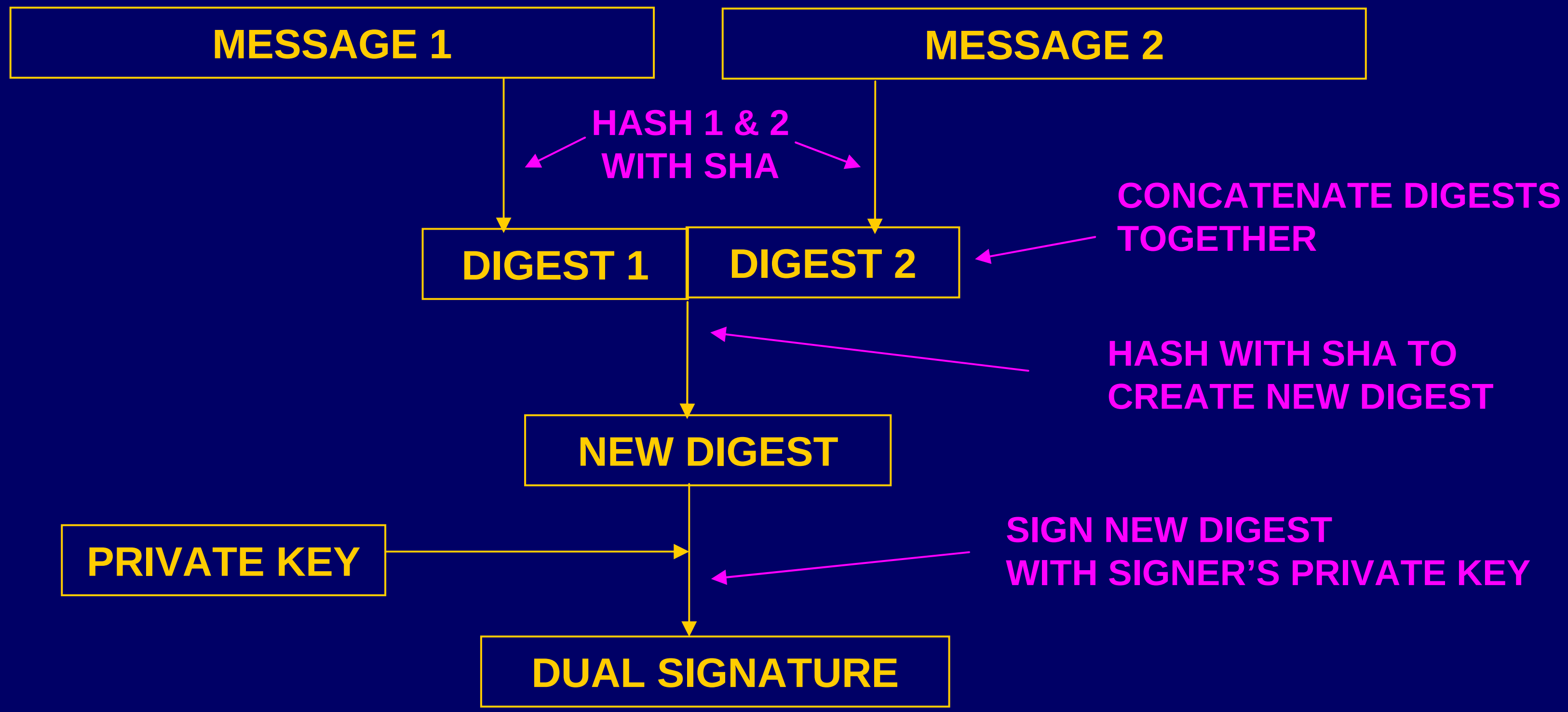
Larry Paulson, Giampaolo Bella, and Fabio Massacci

SET Documentation

- Business Description
  - General overview
  - 72 pages
- Programmer's Guide
  - Message formats & English description of actions
  - 619 pages
- Formal Protocol Definition
  - Message formats & the equivalent ASN.1 definitions
  - 254 pages

Total: 945 pages

Dual Signatures



- Link two messages sent to different receivers
- Each receiver can only read one message
  - Alice checks (message1, digest2, dual sig)
  - Bob checks (message2, digest1, dual sig)

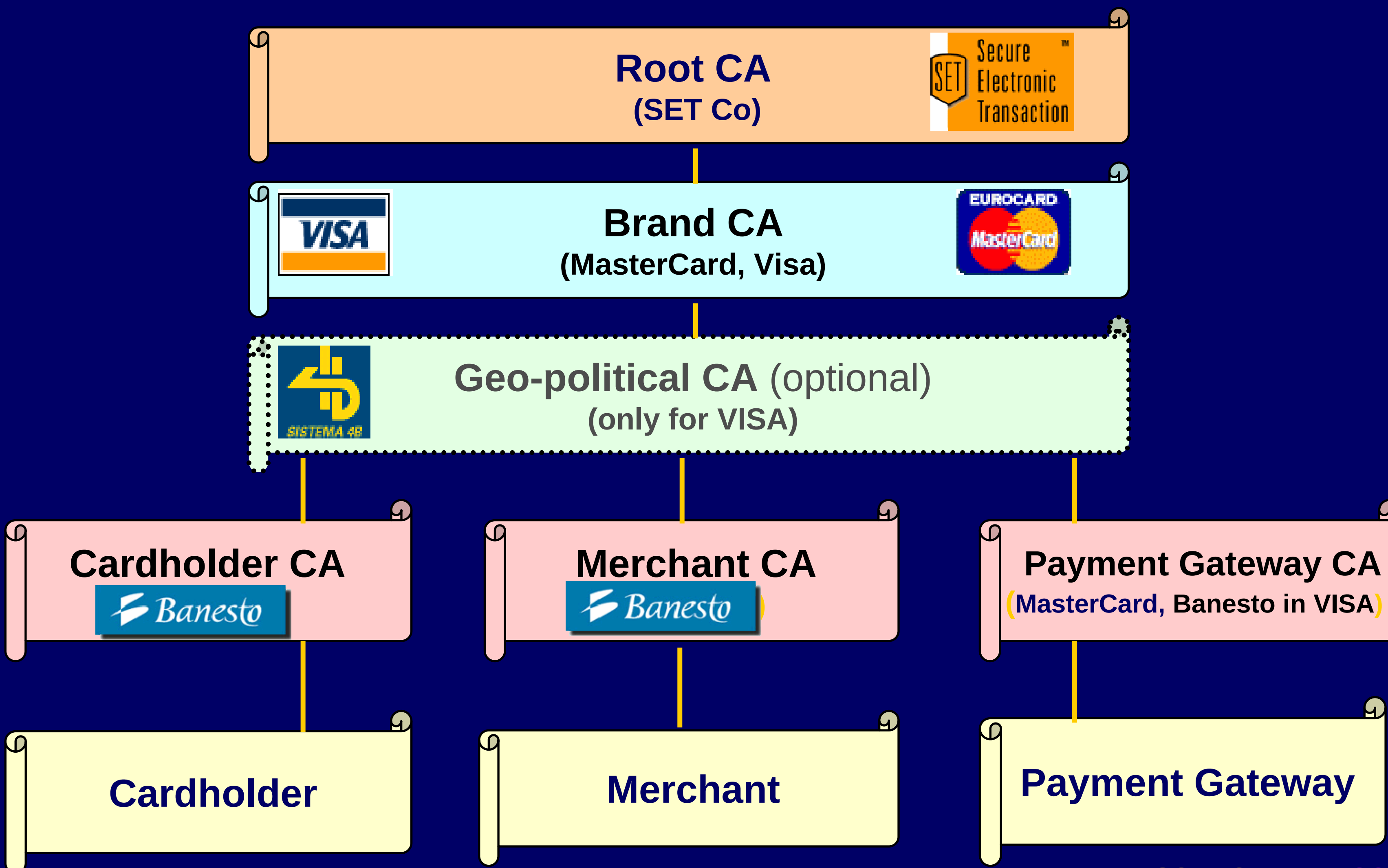
Verifying the SET Protocols

- Several sub-protocols
- Complex cryptographic primitives
  - Dual signatures for partial sharing of secrets
- Many types of principals
  - Cardholder, Merchant, Payment Gateway, CAs
- 1000 pages of specification and description
- SET is probably the upper limit of realistic verification

SET Terminology

- Issuer
  - Cardholder's bank
- Acquirer
  - Merchant's bank
- Payment gateway
  - Pays the merchant
- Certificate authority (CA)
  - Issues electronic credentials
- Trust hierarchy
  - Top CAs certify other CAs in the chain

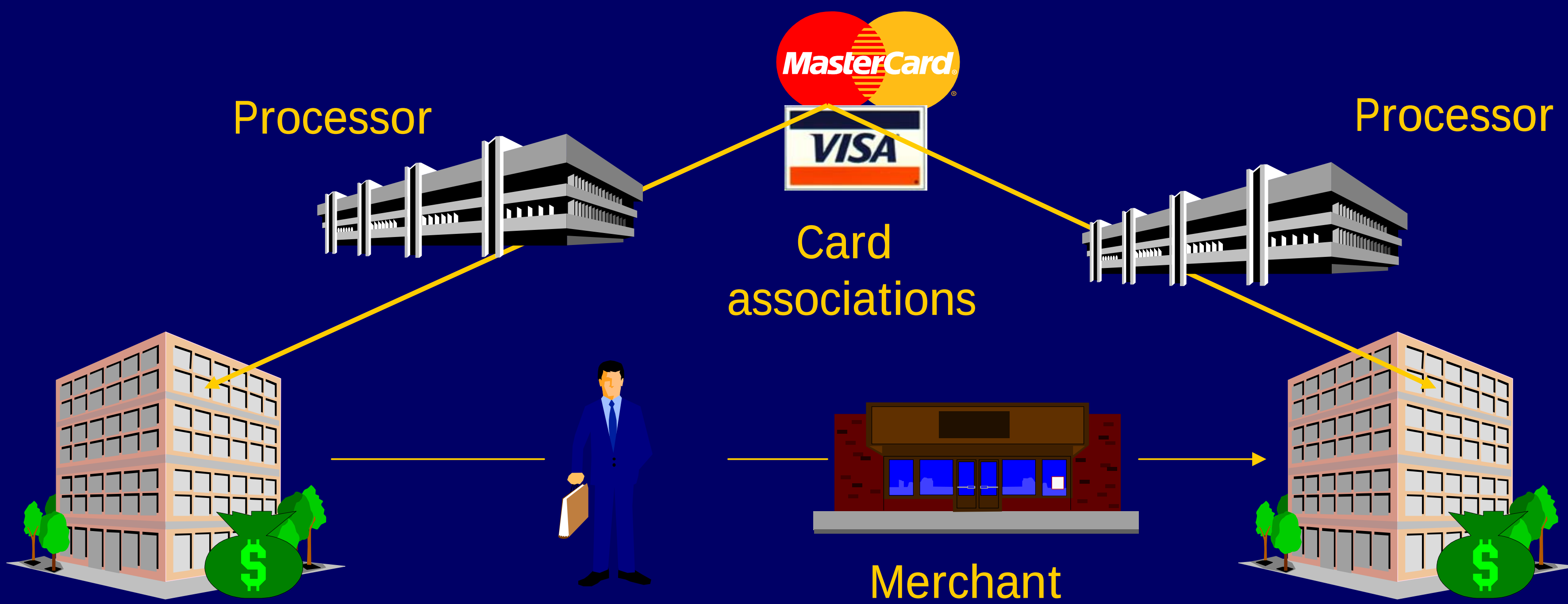
SET Certificate Hierarchy



SOURCE: INZA.COM

SOURCE: Michael I Shamos

Players



Issuing Bank
- Issues card
- Extends credit
- Assumes risk of card
- Cardholder reporting

Merchant Bank (Acquirer)
- Sets up merchant
- Extends credit
- Assumes risk of merchant

SOURCE: Michael I Shamos

1. Customer
- pays with card
- card swiped for mag data read
- (get signature)

2.Card Authorization
via dial, lease line, satellite

3 . Acquiring Bank's Processor
- directs connections to MC /VI
- obtains authorization from Issuer
- returns response to merchant
- five digit number that must be stored

MasterCard VISA

4 . Issuing Bank / Processor
- receives authoriz'n request
- verifies available funds
- places hold on funds

Authorizations

Batch settlement

5. Merchant
- stores authorizations and sales conducted
- captures sales (at end of day)
- submits batch for funding

6. Acquiring Bank / Processor
- scans settlement file
- verifies authorizations match captured data
- prepares file for MC/VI
- prepares funding file
- records txs for reporting

7. Issuing Bank / Processor
- receives settlement file from MC / VI
- funds MC / VI
- matches txs to auths
- post txs to cardholder
- records transactions for reporting

8. MC / VI
debit issuers / credit acquirers

9. Acquiring Bank
funds merchant

SET Consists in 5 Subprotocols

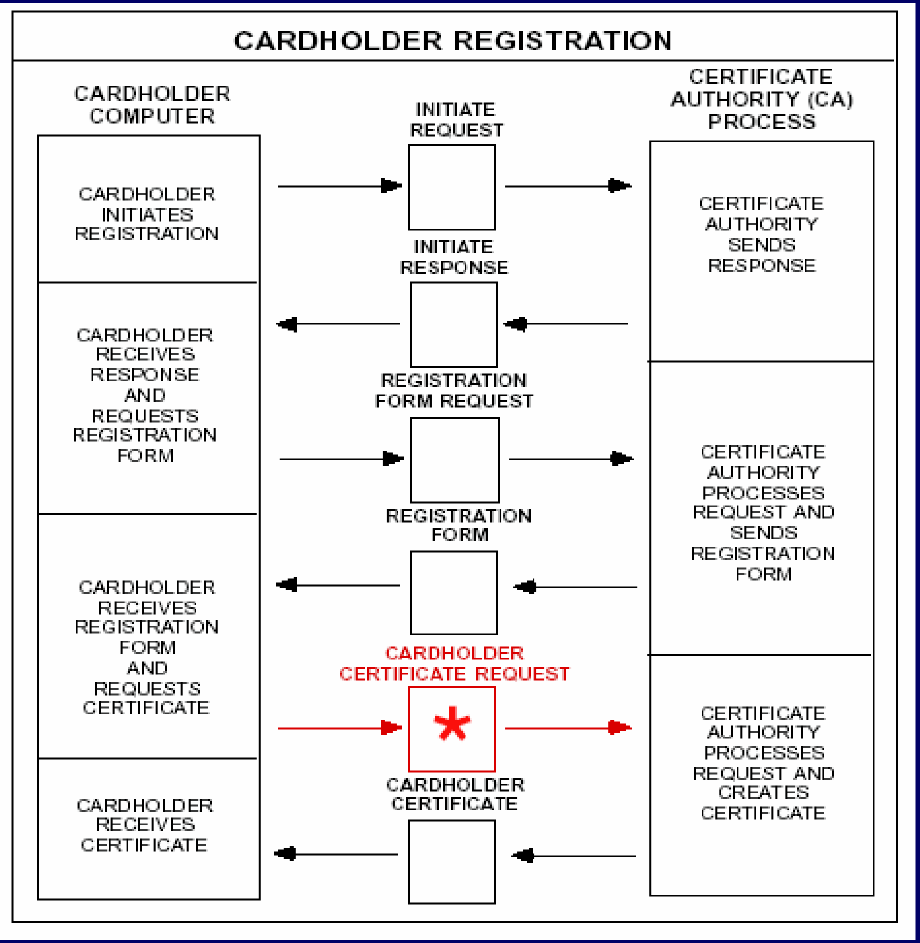
- Cardholder registration
- Merchant registration
- Purchase request
- Payment authorization
- Payment capture

Will look at these two briefly

Cardholder Registration

- Two parties
  - Cardholder
  - Certificate authority CA
- Cardholder sends credit card number to CA
- Cardholder completes *registration form*
  - Inserts security details
  - Discloses his public signature key
- Outcomes
  - Cardholder’s bank can vet the registration
  - CA associates cardholder’s signing key with card details

SET Registration Subprotocol



Certificate Request in Isabelle

$$
\begin{array}{l}
[\![evs5 \in set\_cr;\quad C = Cardholder\ k; \\
\quad Nonce\ NC3 \notin used\ evs5; \\
\quad Nonce\ CardSecret \notin used\ evs5;\ NC3 \neq CardSecret; \\
\quad Key\ KC2 \notin used\ evs5;\ KC2 \in symKeys; \\
\quad Key\ KC3 \notin used\ evs5;\ KC3 \in symKeys;\ KC2 \neq KC3; \\
\quad Gets\ C\ \ldots\ \in set\ evs5;\quad Says\ C\ (CA\ i)\ \ldots\ \in set\ evs5]\!] \\
\Longrightarrow Says\ C\ (CA\ i) \\
\qquad \{\!|Crypt\ KC3\ \{\!|Agent\ C,\ Nonce\ NC3,\ Key\ KC2,\ Key\ cardSK, \\
\qquad\qquad Crypt\ (invKey\ cardSK) \\
\qquad\qquad\quad (Hash\{\!|Agent\ C,\ Nonce\ NC3,\ Key\ KC2, \\
\qquad\qquad\qquad Key\ cardSK,\ Pan(pan\ C), \\
\qquad\qquad\qquad Nonce\ CardSecret|\!\})|\!\}, \\
\qquad\quad Crypt\ EKi\ \{\!|Key\ KC3,\ Pan\ (pan\ C),\ Nonce\ CardSecret|\!\}|\!\} \\
\quad \#\ evs5 \in set\_cr
\end{array}
$$

Key dependency:
KC3 protects KC2,
EKi protects KC3

Public signing key

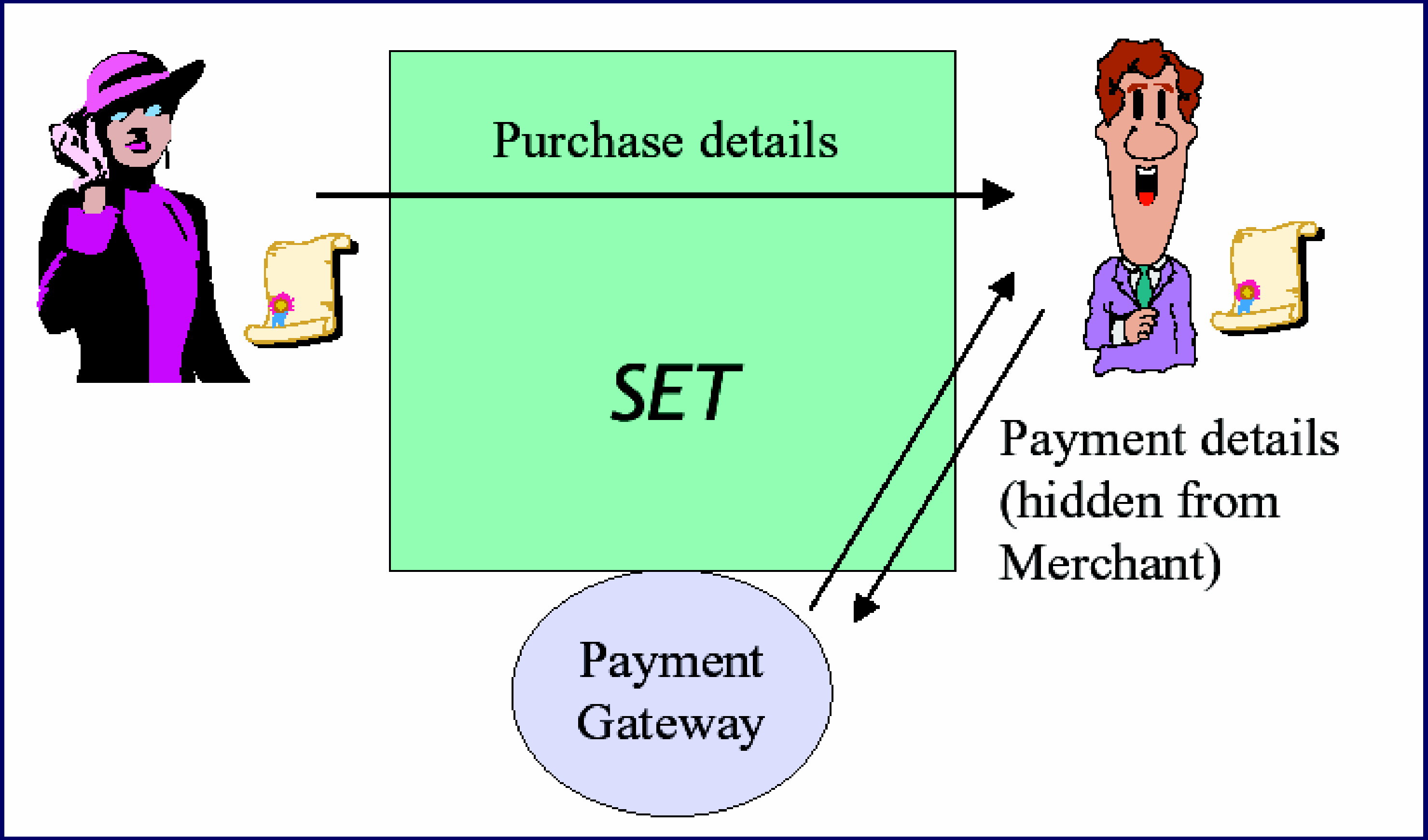
Encrypted, signed request to register account number (PAN) and to encrypt reply with key KC2

The symmetric key KC3 for decrypting the request is encrypted under CA's key EKi

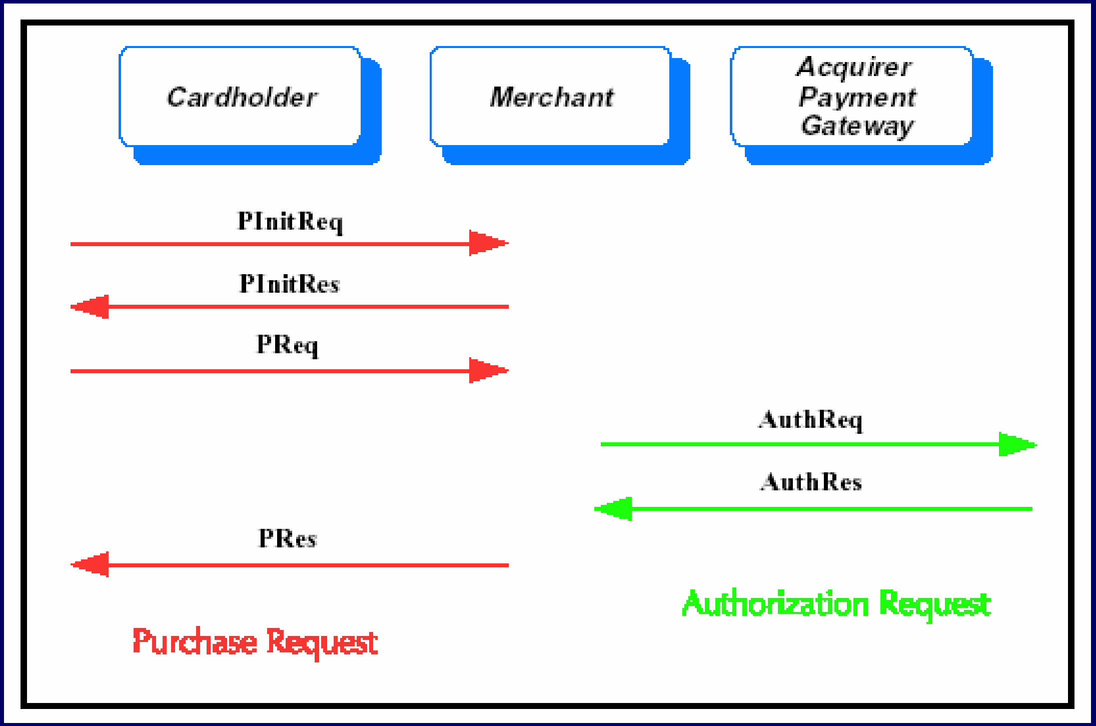
Secrecy of Session Keys and Nonces

- Secrecy is modeled as dependency
  - Session keys: EKi protects KC3, KC3 protects cardholder's request (which includes symmetric key KC2 and public key cardSK), KC2 protects CA's reply
  - Nonces: KC3 protects NC3, EKi protects CardSecret
- Dependency theorem
  - To learn KC2, need to know KC3; to learn KC3, need to know private key corresponding to EKi, etc.
- "Base case" lemmas
  - Session keys never encrypt PANs
  - Session keys never encrypt private keys

SET Purchase Subprotocol



SET Messages (Purchase Phase)



Dual Signatures for Privacy

- 3-way agreement with partial knowledge
  - Cardholder shares Order Information (OI) only with Merchant
  - Cardholder shares Payment Information (PI) only with Payment Gateway
- Cardholder signs hashes of OI, PI
  - Merchant can verify signature on hashed OI because he knows order description
  - Bank learns purchase amount from merchant and verifies its consistency with signed hash of PI
- Signatures guarantee non-repudiation

Purchase Request in Isabelle

```
⟦evsPReqS ∈ set_pur;  C = Cardholder k;  M = Merchant i;
  HOD = Hash⦃Number OrderDesc, Number PurchAmt⦄;
  PIHead = ⦃Number LID_C, Number XID, HOD, Number PurchAmt, Age
            Hash⦃Number XID, Nonce (CardSecret k)⦄⦄;
  OIData = ⦃Number XID, Nonce Chall_C, HOD, Nonce Cha      ⦄;
  PANData = ⦃Pan (pan C), Nonce (PANSecret k)⦄;
  PIData = ⦃PIHead, PANData⦄;
  PIDualSigned = ⦃sign (priSK C) ⦃Hash PIData, Hash OIData⦄,
                  EXcrypt KC2 EKj ⦃PIHead, Hash OIData⦄ PANData⦄;
  Gets C (sign (priSK M) ⦃...⦄) ∈ set evsPReqS;
  trans_details XID = ⦃Agent C, Agent M, Number OrderDesc,
                       Number PurchAmt⦄;
  Says C M ⦃Number LID_C, Nonce Chall_C⦄ ∈ set evsPReqS⟧
⟹ Says C M ⦃PIDualSigned, OIData, Hash PIData⦄
    # evsPReqS ∈ set_pur
```

PIdata includes only amount.
It is hashed and signed to prevent merchant from cheating.

Forming the dual signature

Transaction details for XID

Account data is encrypted with session key to hide it from merchant.

SET Proofs are Complicated

- Massive redundancy caused by hashing and dual signatures
  - 9 copies of “purchase amount” in one message!
- Many nested digital envelopes for key dependency
  - Results in multi-page subgoals for proving key dependency theorems
- Yet insufficient redundancy leads to failure of one agreement property
  - Insufficient redundancy = lack of explicit information

Inductive Method: Pros & Cons

- Advantages
  - Reason about arbitrarily large runs, message spaces
  - Trace model close to protocol specification
  - Can “prove” protocol correct
- Disadvantages
  - Does not always give an answer
  - Failure of proof does not always yield an attack
  - Trace-based properties only
  - Labor intensive
    - Must be comfortable with higher-order logic