Distance Bounding Protocols: Computational vs. Symbolic Models

Jorge Toro Pozo
University of Luxembourg

(joint work with S. Mauw, Z. Smith and R. Trujillo)

FutureDB Workshop
Azores, Portugal - April 14, 2018

Luxembourg National Research Fund

UNIVERSITÉ DU LUXEMBOURG

Outline

1. Introduction
2. Probabilistic model based on automata
3. Symbolic model with time and location
4. Symbolic model based on causality
5. Conclusion and Future

Outline

This talk's content

- S. Mauw, J. Toro-Pozo, R. Trujillo-Rasua, "A Class of Precomputation-Based Distance-Bounding Protocols", in *EuroS&P'16*, 2016, pp. 97–111.
- S. Mauw, J. Toro-Pozo, R. Trujillo-Rasua, "Optimality Results on the Security of Lookup-Based Protocols", in *RFIDSec'16*, 2016, pp. 137–150.
- S. Mauw, Z. Smith, J. Toro-Pozo, R. Trujillo-Rasua, "Distance Bounding Protocols: Verification without Time and Location", in *S&P'18*, 2018.

Problem: Relay attack

Source: securepositioning.com

Source: securepositioning.com

Definition

A *relay attack* is a man-in-the-middle attack in which an attacker relays verbatim a message from the sender to a valid receiver.

Solution: Distance-bounding protocols

Definition

A *distance-bounding protocol* is an authentication protocol that checks that the distance between verifier and prover is below a given threshold.

Solution: Distance-bounding protocols

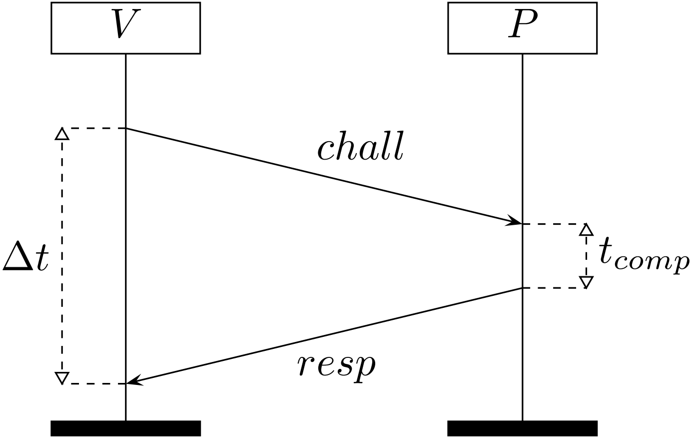
Definition

A *distance-bounding protocol* is an authentication protocol that checks that the distance between verifier and prover is below a given threshold.

How to measure (or bound) distance?

- Verifier sends a challenge.
- Prover provides corresponding response.
- Verifier measures the round-trip-time.

A challenge/response round



$$dist(V, P) = \frac{1}{2} \cdot c \cdot (\Delta t - t_{comp}) \leq \frac{1}{2} \cdot c \cdot \Delta t$$

Outline

Lookup Protocols: Motivation

- To obtain an accurate upper-bound on the distance, the computational time on the prover's side must be as short as possible.
- Solution: Pre-computing the possible responses and store them in a constant-time-access structure, such as a lookup-table.
- Protocols with a final crypto-verification phase could be outperformed by a precomputation-based protocol with more rounds, with no increase of the computational cost: $\forall n, \exists m\colon \left(\frac{1}{2}\right)^n > \left(\frac{3}{4}\right)^m$.
- Partial information can be given if the protocol gets interrupted before finishing.

Lookup Protocols

Lookup protocols are DB protocols such that:

1. In the fast phase, the responses to the challenges are the result of lookup operations from a table.

Lookup Protocols

Lookup protocols are DB protocols such that:

1. In the fast phase, the responses to the challenges are the result of lookup operations from a table.

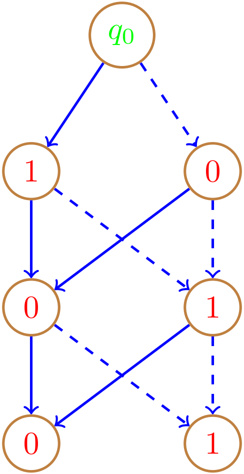
2. either do NOT have a final verification phase at all or

Lookup Protocols

Lookup protocols are DB protocols such that:

1. In the fast phase, the responses to the challenges are the result of lookup operations from a table.
2. either do NOT have a final verification phase at all or
3. having replied correctly and on time to all challenges is SUFFICIENT to pass the protocol (do not have any crypto-based verification mechanism such as opening commits, keyed hash functions, signatures...).

Protocol Representation: State-Labeled DFA

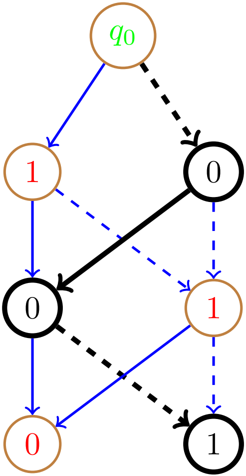


$$A = (\Sigma, \Gamma, Q, q_0, \delta, \ell)$$

Σ is the set of input symbols
Γ is the set of output symbols
Q is the set of states
$q_0 \in Q$ is the initial state
$\delta \colon Q \times \Sigma \to Q$ is the transition function
$\ell \colon Q \to \Gamma$ is the state labeling function

Protocol Representation

State-Labeled DFA

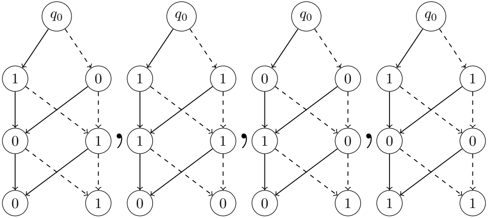


$$A = (\Sigma, \Gamma, Q, q_0, \delta, \ell)$$

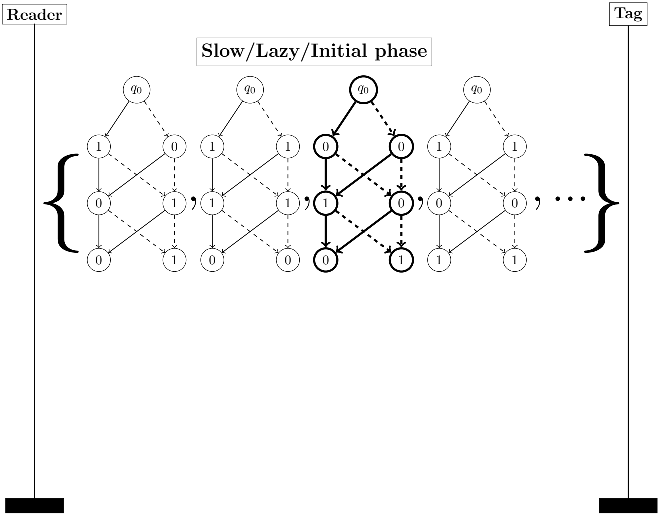
Σ is the set of input symbols
Γ is the set of output symbols
Q is the set of states
$q_0 \in Q$ is the initial state
$\delta \colon Q \times \Sigma \to Q$ is the transition function
$\ell \colon Q \to \Gamma$ is the state labeling function

$$\Omega_A\,(101) = 001$$

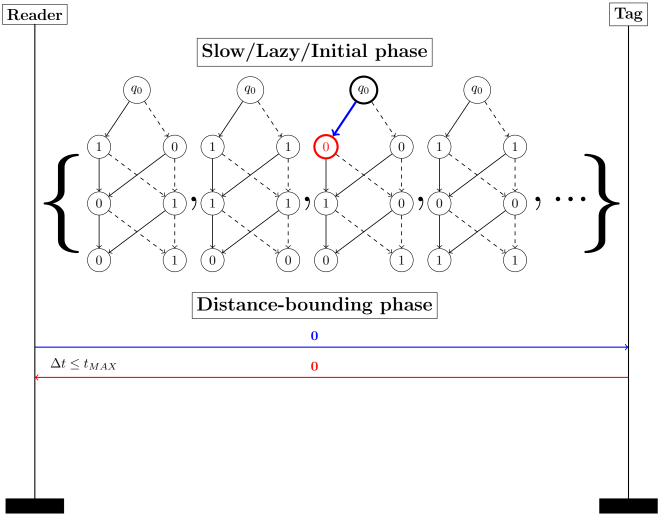
Protocol Representation

$P = \{$  $, \cdots \}$

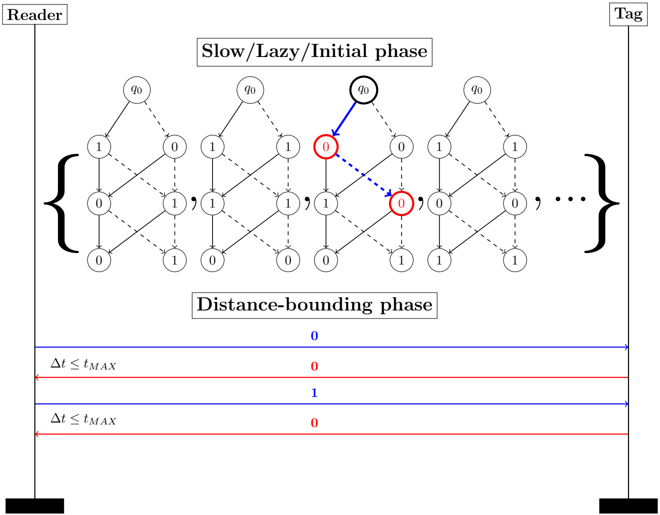
Protocol Execution



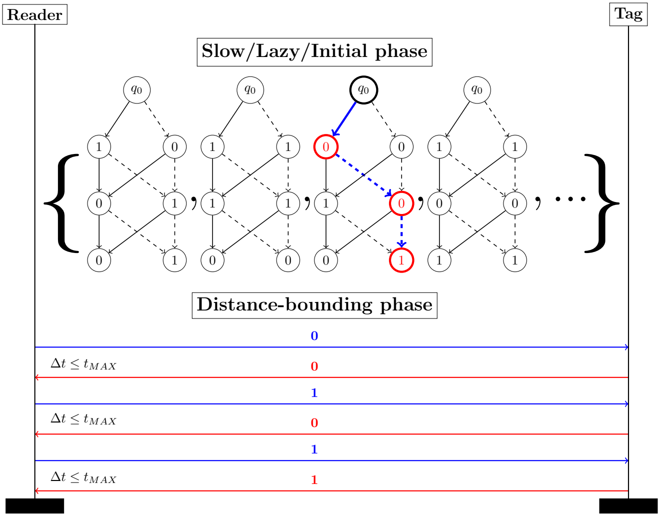
Protocol Execution



Protocol Execution



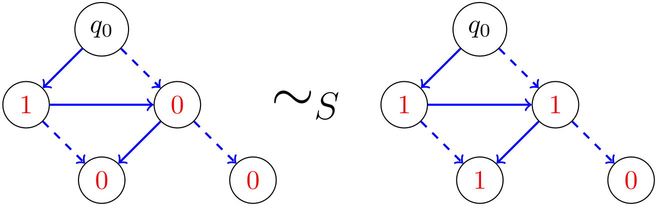
Protocol Execution



Automata Equivalence Relations

- State-label-insensitive relation ($\sim_S$)

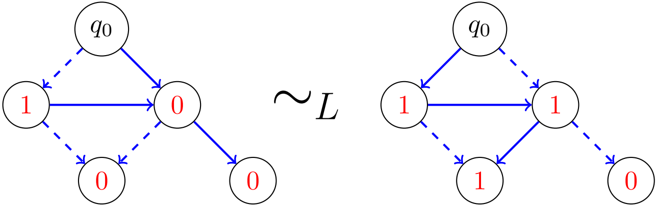
$$(\Sigma, \Gamma, Q, q_0, \delta, \ell) \sim_S (\Sigma, \Gamma, Q, q_0, \delta, \ell')$$



- Label-insensitive relation ($\sim_L$)

$$(\Sigma, \Gamma, Q, q_0, \delta, \ell) \sim_L (\Sigma, \Gamma, Q, q_0, \delta', \ell')$$

such that $\forall q \in Q : \{\delta(q,c) \mid c \in \Sigma\} = \{\delta'(q,c) \mid c \in \Sigma\}$.



Consistency and Closeness

- A protocol P is consistent w.r.t $\sim_R$ iff

$$A, A' \in P \colon A \sim_R A'$$

Consistency and Closeness

- A protocol P is consistent w.r.t $\sim_R$ iff

$$A, A' \in P \colon A \sim_R A'$$

- A protocol P is closed under $\sim_R$ iff

$$\forall (A, A') \in \sim_R \colon A \in P \implies A' \in P$$

Consistency and Closeness

- A protocol P is consistent w.r.t $\sim_R$ iff

$$A, A' \in P \colon A \sim_R A'$$

- A protocol P is closed under $\sim_R$ iff

$$\forall (A, A') \in \sim_R \colon A \in P \implies A' \in P$$

- The closure of P w.r.t $\sim_R$, denoted by P^R, is the minimal superset of P that is closed under $\sim_R$.

Some formulas

Given a layered automaton A:

$$mafia\left(\{A\}^S\right) = \frac{1}{|\Sigma|^n \cdot |\Gamma|^n} \max_{x \in \Sigma^n} \left\{ \sum_{y \in \Sigma^n} |\Gamma|^{collisions_A(x,y)} \right\}$$

$$mafia\left(\{A\}^L\right) = \frac{1}{|\Sigma|^{2n} \cdot |\Gamma|^n} \sum_{x,y \in \Sigma^n} |\Gamma|^{collisions_A(x,y)}$$

Some formulas

Given a layered automaton A:

$$mafia\left(\{A\}^S\right) = \frac{1}{|\Sigma|^n \cdot |\Gamma|^n} \max_{x \in \Sigma^n} \left\{ \sum_{y \in \Sigma^n} |\Gamma|^{collisions_A(x,y)} \right\}$$

$$mafia\left(\{A\}^L\right) = \frac{1}{|\Sigma|^{2n} \cdot |\Gamma|^n} \sum_{x,y \in \Sigma^n} |\Gamma|^{collisions_A(x,y)}$$

Trivially, $mafia\left(\{A\}^S\right) \geq mafia\left(\{A\}^L\right)$
because $\max\{u_i\} \geq \frac{1}{N}(u_1 + \cdots + u_N)$

Main theorem

Theorem

For any layered lookup protocol P *the following holds:*

$$mafia(P) \geq mafia\left(P^S\right) \geq mafia\left(\{A\}^L\right) \geq mafia(Tree),$$

for some $A \in P$.

Main theorem

Theorem

For any layered lookup protocol P the following holds:

$$mafia(P) \geq mafia\left(P^S\right) \geq mafia\left(\{A\}^L\right) \geq mafia\left(\left\{M_{size(P)}\right\}^L\right) \geq mafia(Tree),$$

for some $A \in P$.

Conclusions

- We have formalized relevant structural properties of lookup protocols that have been used in a rather intuitive way.
- We provided simple formulas for computing mafia success probability for all but one lookup protocols.
- We have addressed (partially) the security-memory trade-off problem in layered protocols.

Outline

The basis of our work

Model based on time and location

- P. Schaller, B. Schmidt, D. A. Basin, and S. Capkun, "Modeling and verifying physical properties of security protocols for wireless networks," in *CSF'09*, 2009, pp. 109–123.
- D. A. Basin, S. Capkun, P. Schaller, and B. Schmidt, "Let's get physical: Models and methods for real-world security protocols," in *TPHOLs'09*, 2009, pp. 1–22.
- C. J. F. Cremers, K. B. Rasmussen, B. Schmidt, and S. Capkun, "Distance hijacking attacks on distance bounding protocols," in *S&P'12*, 2012, pp. 113–127.

Specification and Execution

- **Agents**: the set Agent, partitioned into $\{\mathit{Honest}, \mathit{Dishonest}\}$.

Specification and Execution

- **Agents**: the set $Agent$, partitioned into $\{Honest, Dishonest\}$.
- **Messages**: the set Msg defined by:

$$m ::= atom \mid (m, m') \mid f(m) \mid \{m\}_{m'}$$

where $atom \in Nonce \cup Agent \cup Const$ and $f \in Fun$.

Specification and Execution

- **Agents**: the set *Agent*, partitioned into $\{Honest, Dishonest\}$.
- **Messages**: the set *Msg* defined by:

$$m ::= atom \mid (m, m') \mid f(m) \mid \{m\}_{m'}$$

  where $atom \in Nonce \cup Agent \cup Const$ and $f \in Fun$.
- **Events**: the set *Ev* defined by:

$$e ::= send_A(m) \mid recv_A(m) \mid claim_A(B, e', e'')$$

Specification and Execution

- **Agents**: the set *Agent*, partitioned into $\{Honest, Dishonest\}$.
- **Messages**: the set *Msg* defined by:

$$m ::= atom \mid (m, m') \mid f(m) \mid \{m\}_{m'}$$

  where $atom \in Nonce \cup Agent \cup Const$ and $f \in Fun$.

- **Events**: the set *Ev* defined by:

$$e ::= send_A(m) \mid recv_A(m) \mid claim_A(B, e', e'')$$

- **Trace**: a sequence $(t_1, e_1) \cdots (t_n, e_n)$ with $t_i \in \mathbb{R}, e_i \in Ev$.

Specification and Execution

- **Agents**: the set *Agent*, partitioned into $\{Honest, Dishonest\}$.
- **Messages**: the set *Msg* defined by:

$$m ::= atom \mid (m, m') \mid f(m) \mid \{m\}_{m'}$$

where $atom \in Nonce \cup Agent \cup Const$ and $f \in Fun$.

- **Events**: the set *Ev* defined by:

$$e ::= send_A(m) \mid recv_A(m) \mid claim_A(B, e', e'')$$

- **Trace**: a sequence $(t_1, e_1) \cdots (t_n, e_n)$ with $t_i \in \mathbb{R}, e_i \in Ev$.
- **Specification**: a set of rules defining the actions of *honest* agents.

Specification and Execution

- **Agents**: the set *Agent*, partitioned into $\{Honest, Dishonest\}$.
- **Messages**: the set *Msg* defined by:

$$m ::= atom \mid (m, m') \mid f(m) \mid \{m\}_{m'}$$

where $atom \in Nonce \cup Agent \cup Const$ and $f \in Fun$.

- **Events**: the set *Ev* defined by:

$$e ::= send_A(m) \mid recv_A(m) \mid claim_A(B, e', e'')$$

- **Trace**: a sequence $(t_1, e_1) \cdots (t_n, e_n)$ with $t_i \in \mathbb{R}, e_i \in Ev$.
- **Specification**: a set of rules defining the actions of *honest* agents.
- And some other stuff such as **message deduction**.

Specification and Execution

- **Trace**: a sequence $(t_1, e_1) \cdots (t_n, e_n)$ with $t_i \in \mathbb{R}, e_i \in Ev$.

$$\alpha = (1.3, send_{Alice}(m)) \cdot (3, recv_{Bob}(m)) \cdot (5, send_{Bob}(h(m)))$$

Specification and Execution

- **Trace**: a sequence $(t_1, e_1) \cdots (t_n, e_n)$ with $t_i \in \mathbb{R}, e_i \in Ev$.

$$\alpha = (1.3, send_{Alice}(m)) \cdot (3, recv_{Bob}(m)) \cdot (5, send_{Bob}(h(m)))$$

$$dist(Alice, Bob) \leq c \cdot (3 - 1.3)$$

Specification and Execution

- **Specification**: a set of rules defining the actions of *honest* agents.

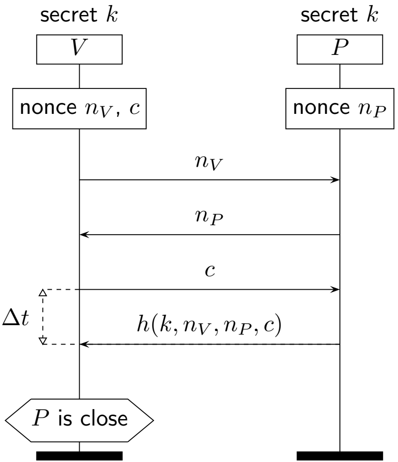
  $\mathcal{P} = \{R_1 \ldots, R_n\}$ where the R_i's have the form:

$$\frac{t \geq maxt(\alpha) \quad A \in Honest \quad cond_1 \quad \cdots \quad cond_n}{(\alpha, (t, e)) \in R_i}$$

  In words: if conditions $cond_j$ are met, then the agent A can execute the event e at time t.

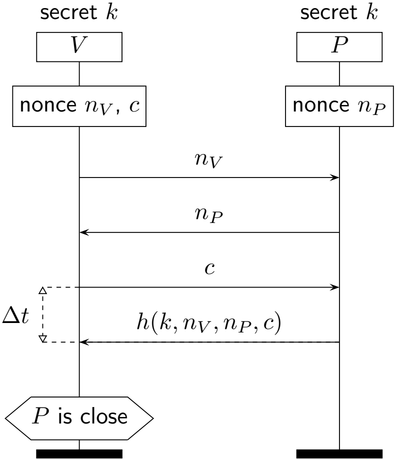
Example

Hancke and Kuhn's 2005



Example

Hancke and Kuhn's 2005



$$\mathcal{P} = \{R_1, R_2, R_3, R_4, R_5\}$$

$$\frac{V \in Hnst \quad t \geq maxt(\alpha) \quad fresh(n_V, \alpha)}{(\alpha, (t, send_V(n_V))) \in R_1}$$

$$\frac{P \in Hnst \quad t \geq maxt(\alpha) \quad (t', recv_P(n_V)) \in \alpha \quad fresh(n_P, \alpha)}{(\alpha, (t, send_P(n_P))) \in R_2}$$

$$\frac{V \in Hnst \quad t \geq maxt(\alpha) \quad (t', send_V(n_V)) \in \alpha \quad (t'', recv_V(n_P)) \in \alpha \quad fresh(c, \alpha)}{(\alpha, (t, send_V(c))) \in R_3}$$

$$\frac{P \in Hnst \quad t \geq maxt(\alpha) \quad (t', recv_P(n_V)) \in \alpha \quad (t'', send_P(n_P)) \in \alpha \quad (t''', recv_P(c)) \in \alpha \quad r = h(sh(V, P), n_V, n_P, c)}{(\alpha, (t, send_P(r))) \in R_4}$$

$$\frac{V \in Hnst \quad t \geq maxt(\alpha) \quad (t', send_V(n_V)) \in \alpha \quad (t'', recv_V(n_P)) \in \alpha \quad u = send_V(c) \quad v = recv_V(r) \quad (tu, u) \in \alpha \quad (tv, v) \in \alpha \quad r = h(sh(V, P), n_V, n_P, c)}{(\alpha, (t, claim_V(P, u, v))) \in R_5}$$

Specification and Execution

- And some other stuff such as **message deduction**.

  The set *infer* (A, α) contains all messages that A can infer from α:

$$\frac{m \in init\,(A)}{m \in infer\,(A,\alpha)} \qquad \frac{(t, recv_A(m)) \in \alpha}{m \in infer\,(A,\alpha)} \qquad \frac{(m_1, m_2) \in infer\,(A,\alpha)}{m_i \in infer\,(A,\alpha)}$$

$$\frac{\begin{array}{c} m_1 \in infer\,(A,\alpha) \\ m_2 \in infer\,(A,\alpha) \end{array}}{(m_1, m_2) \in infer\,(A,\alpha)} \qquad \frac{\begin{array}{c} m \in infer\,(A,\alpha) \\ f \in Func \setminus \{sk, \_^{-1}, sh\} \end{array}}{f(m) \in infer\,(A,\alpha)}$$

$$\frac{\begin{array}{c} m \in infer\,(A,\alpha) \\ k \in infer\,(A,\alpha) \end{array}}{\{m\}_k \in infer\,(A,\alpha)} \qquad \frac{\begin{array}{c} \{m\}_k \in infer\,(A,\alpha) \\ k^{-1} \in infer\,(A,\alpha) \end{array}}{m \in infer\,(A,\alpha)}$$

Execution model

The set of all valid traces $Tr(\mathcal{P})$ is defined by:

$$\alpha \cdot (t, e) \in Tr(\mathcal{P}) \iff \alpha \in Tr(\mathcal{P}) \land \exists R \in \mathcal{P} \cup \{Int, Net\} \colon (\alpha, (t, e)) \in R$$

where:

$$\frac{I \in Dishonest \quad t \geq maxt(\alpha) \quad m \in infer(I, \alpha)}{(\alpha, (t, send_I(m))) \in Int} \qquad \frac{t \geq maxt(\alpha) \quad (t', send_A(m)) \in \alpha \quad t \geq t' + dist(A, B)/c}{(\alpha, (t, recv_B(m))) \in Net}$$

Secure distance-bounding

Definition

A protocol $\mathcal{P}$ satisfies *secure distance-bounding* if and only if:

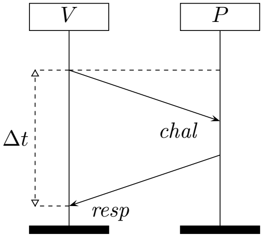
$$\forall \alpha \in Tr\left(\mathcal{P}\right), (t, claim_V(P, u, v)) \in \alpha :$$
$$\exists (tu, u), (tv, v) \in \alpha, P' \approx P : dist(V, P') \leq \frac{c \cdot (tv - tu)}{2}$$

where $\approx = \{(A, A) \mid A \in Honest\} \cup Dishonest \times Dishonest$.

Implemented in Isabelle/HOL, available at
http://www.infsec.ethz.ch/research/software/protoveriphy.html

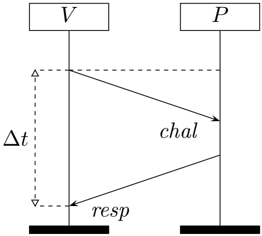
Outline

Three timing scenarios

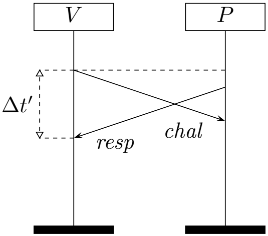


Correct timing

Three timing scenarios

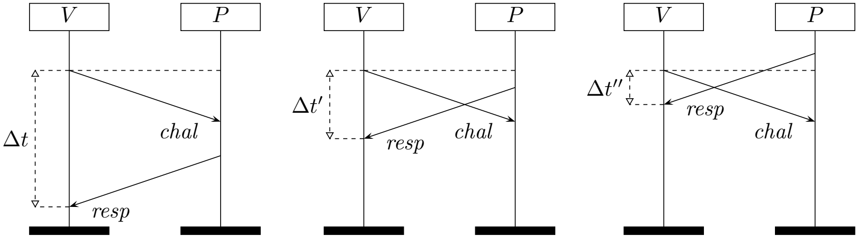


Correct timing



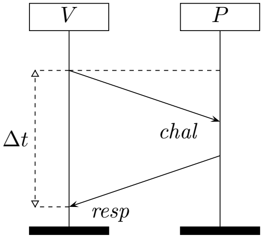
Early timing

Three timing scenarios

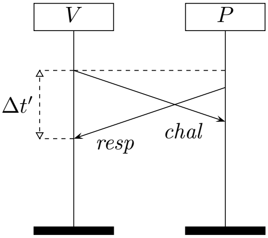


Correct timing Early timing Very early timing

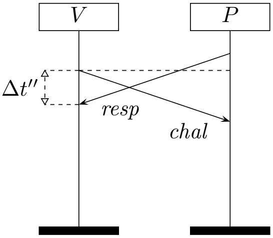
Three timing scenarios



Correct timing



Early timing



Very early timing

Claim: If there is an early timing, then there is a very early timing.

Main theorem

Theorem

A protocol $\mathcal{P}$ satisfies secure distance-bounding if and only if:

$$\forall \sigma \in \pi(Tr(\mathcal{P})), claim_V(P, u, v) \in \sigma:$$
$$\exists u \cdot e \cdot v \sqsubseteq \sigma: actor(e) \approx P$$

where $\pi(T) = \{e_1 \cdots e_n \mid (t_1, e_1) \cdots (t_n, e_n) \in T\}$.

Main theorem

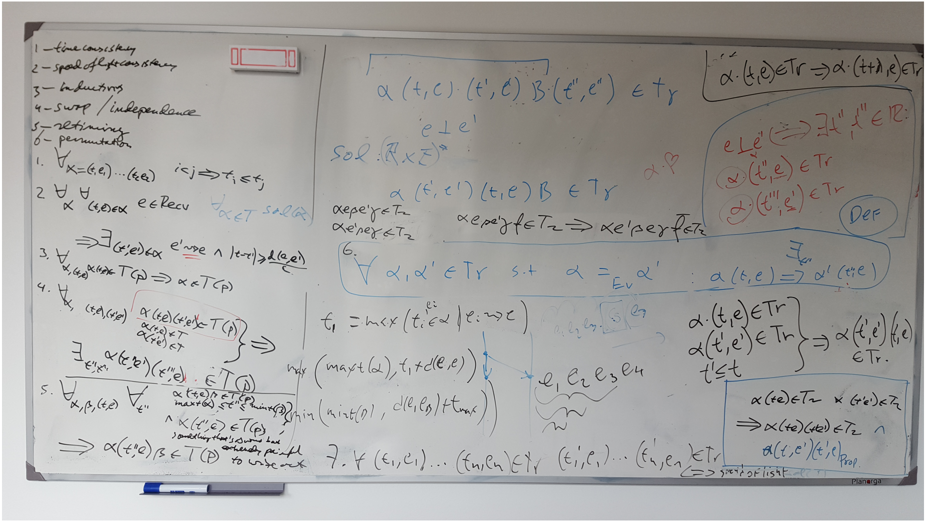
Theorem

A protocol $\mathcal{P}$ satisfies secure distance-bounding if and only if:

$$\forall \sigma \in \pi(Tr(\mathcal{P})), claim_V(P, u, v) \in \sigma :$$
$$\exists u \cdot e \cdot v \sqsubseteq \sigma : actor(e) \approx P$$

where $\pi(T) = \{e_1 \cdots e_n \mid (t_1, e_1) \cdots (t_n, e_n) \in T\}$.

Verified 12+ protocols in Tamarin, available at
http://satoss.uni.lu/software/DBVerify/



1 – time consistency
2 – speed of light consistency
3 – inductions
4 – swap / independence
5 – retiming
6 – permutation
Def

Proof idea

Characterise timed-traces model

For every $(t_1, e_1) \cdots (t_n, e_n) \in Tr(\mathcal{P})$:

Proof idea

Characterise timed-traces model

For every $(t_1, e_1) \cdots (t_n, e_n) \in Tr(\mathcal{P})$:

1. $t_1 \leq \cdots \leq t_n$

For every $(t_1, e_1) \cdots (t_n, e_n) \in Tr(\mathcal{P})$:

1. $t_1 \leq \cdots \leq t_n$
2. $t_n = recv_A(m)$ implies $i < n$ exists such that $e_i = send_B(m)$ and $t_n - t_i \geq dist(A, B)/c$

Proof idea

Characterise timed-traces model

For every $(t_1, e_1) \cdots (t_n, e_n) \in Tr(\mathcal{P})$:

1. $t_1 \leq \cdots \leq t_n$
2. $t_n = recv_A(m)$ implies $i < n$ exists such that $e_i = send_B(m)$ and $t_n - t_i \geq dist(A, B)/c$
3. if $(t'_1, e_1) \cdots (t'_n, e_n)$ satisfies (1) and (2) then $(t'_1, e_1) \cdots (t'_n, e_n) \in Tr(\mathcal{P})$

Proof idea

Characterise timed-traces model

For every $e_1 \cdots e_n \in \pi(Tr(\mathcal{P}))$:

Proof idea

Characterise timed-traces model

For every $e_1 \cdots e_n \in \pi(\mathit{Tr}(\mathcal{P}))$:

1. $e_1 \cdots e_{n-1} \in \pi(\mathit{Tr}(\mathcal{P}))$

For every $e_1 \cdots e_n \in \pi(Tr(\mathcal{P}))$:

1. $e_1 \cdots e_{n-1} \in \pi(Tr(\mathcal{P}))$
2. $e_n \notin Recv$ and $actor(e_{n-1}) \neq actor(e_n)$ then $e_1 \cdots e_{n-2} \cdot e_n \in \pi(Tr(\mathcal{P}))$

Proof idea

Characterise timed-traces model

For every $e_1 \cdots e_n \in \pi(Tr(\mathcal{P}))$:

1. $e_1 \cdots e_{n-1} \in \pi(Tr(\mathcal{P}))$
2. $e_n \notin Recv$ and $actor(e_{n-1}) \neq actor(e_n)$ then $e_1 \cdots e_{n-2} \cdot e_n \in \pi(Tr(\mathcal{P}))$
3. $e_n = send_A(m)$ implies $e_1 \cdots e_n \cdot recv_B(m) \in \pi(Tr(\mathcal{P}))$

Proof idea

Characterise timed-traces model

For every $e_1 \cdots e_n \in \pi(Tr(\mathcal{P}))$:

1. $e_1 \cdots e_{n-1} \in \pi(Tr(\mathcal{P}))$
2. $e_n \notin Recv$ and $actor(e_{n-1}) \neq actor(e_n)$ then $e_1 \cdots e_{n-2} \cdot e_n \in \pi(Tr(\mathcal{P}))$
3. $e_n = send_A(m)$ implies $e_1 \cdots e_n \cdot recv_B(m) \in \pi(Tr(\mathcal{P}))$
4. $\forall A, B \in Honest^2 \cup Dishonest^2$ it holds that $(e_1 \cdots e_n)[A \mapsto B] \in \pi(Tr(\mathcal{P}))$

The Tamarin dbsec lemma

```
lemma dbsec:
"
All P V m n #t. (
VerifierComplete(P, V, m, n)@t ) ==>
(
  Ex #tc.
    Corrupt(V)@tc
)|(
  Ex #t1 #t2 #t3.
    StartFastPhase(V, m)@t1 &
    Action(P)@t2 &
    EndFastPhase(V, m)@t3 &
    (#t1 < #t2) &
    (#t2 < #t3) &
    ( (#t3 < #t ) | (#t3 = #t) )
)|(
  Ex CAgent #t4 #t5 #t6 #t7.
    StartFastPhase(V, m)@t5 &
    EndFastPhase(V, m)@t7 &
    Corrupted(P, V)@t4 &
    CAction(CAgent)@t6 &
    (#t5 < #t6)&
    (#t6 < #t7)&
    ( (#t7 < #t) | (#t7 = #t) )
)
"
```

Verification in Tamarin

Protocol	Satisfies `dbsec`?	Attack found
BC-Signature	No	DH
BC-FiatShamir	No	DH, DF
BC-Schnorr	No	DH, DF
CRCS	No	DH
Meadows et al.	No	DH
Tree-based	Yes	-
Poulidor	Yes	-
Hancke and Kuhn	Yes	-
Uniform	Yes	-
Kim and Avoine	Yes	-
Munilla et al.	Yes	-
Reid et al.	Yes	-
Swiss-Knife	Yes	-
TREAD-PK	No	MF, DH
TREAD-SH	No	DH
PaySafe	No	DF, DH

What we achieved

- Proved that secure distance-bounding can be formulated through causality.

What we achieved

- Proved that secure distance-bounding can be formulated through causality.
- Provided a fully-automatic verification framework for DB protocols. (simply specify the protocol and click on "verify `dbsec` lemma").

What we achieved

- Proved that secure distance-bounding can be formulated through causality.
- Provided a fully-automatic verification framework for DB protocols. (simply specify the protocol and click on "verify `dbsec` lemma").
- Provided computer-verifiable (in)security proofs for a number of state-of-the-art protocols.

What we achieved

- Proved that secure distance-bounding can be formulated through causality.
- Provided a fully-automatic verification framework for DB protocols. (simply specify the protocol and click on "verify `dbsec` lemma").
- Provided computer-verifiable (in)security proofs for a number of state-of-the-art protocols.
- Identified *unreported* vulnerabilities in two published protocols: PaySafe (FC'15) and TREAD (AsiaCCS'17).

Outline

Probabilistic vs. Symbolic

- Where probabilistic models win:
  - More precise results - there's an attach that succeeds w/ prob. p
  - Arithmetic properties can be fairly-well modeled
- Where symbolic models win:
  - No need to consider each attack individually
  - Automated verification - Tamarin, ProVerif, Scyther, Isabelle
  - Computer-verifiable proofs of (in)security

Future

- Terrorist fraud?
  Requires fancy techniques for corruption modeling.

Future

- Terrorist fraud?
  Requires fancy techniques for corruption modeling.

- Automatic probabilistic analysis?
  Seems hard.

Q & A

Thank you

jorge.toro@uni.lu
http://satoss.uni.lu/jorge