Computational and Symbolic Analysis of Distance-Bounding Protocols

Jorge Toro Pozo

PhD Dissertation

May 14, 2019

Fonds National de la Recherche Luxembourg

UNIVERSITÉ DU LUXEMBOURG

Defence Committee & Supervision

Defence Committee Chair
Prof. Dr. Yves LE TRAON

Defence Committee Vice-Chair & Daily Advisor
Dr. Rolando TRUJILLO RASÚA

Defence Committee Member & Supervisor
Prof. Dr. Sjouke MAUW

Defence Committee Member
Dr. Stéphanie DELAUNE

Defence Committee Member
Dr. Ioana BOUREANU

Outline

Problem: Relay attack

Source https://securepositioning.com

Problem: Relay attack

Source https://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

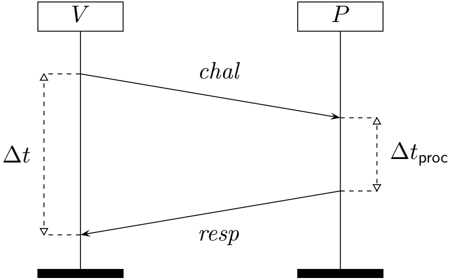
Definition

A *distance-bounding protocol* is a security protocol that, in addition to authentication, established an upper bound on the physical distance between the prover and the verifier.

Solution: Distance-bounding protocols

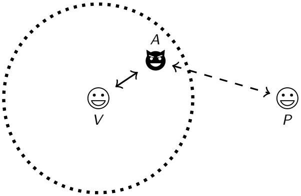
Definition

A *distance-bounding protocol* is a security protocol that, in addition to authentication, established an upper bound on the physical distance between the prover and the verifier.



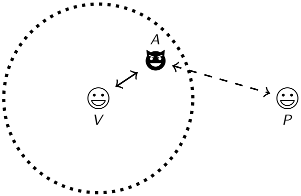
$$
\begin{aligned}
dist(V, P) &\leq \frac{c}{2}(\Delta t - \Delta t_{\text{proc}}) \\
&\leq \frac{c}{2}\Delta t
\end{aligned}
$$

Distance-bounding attacks

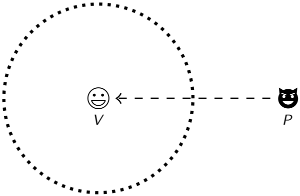


(a) Mafia fraud

Distance-bounding attacks

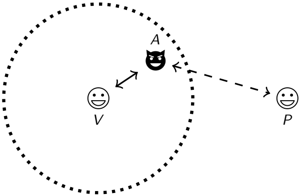


(a) Mafia fraud

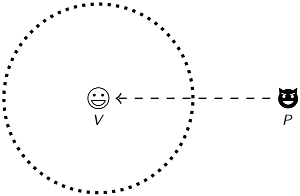


(b) Distance fraud

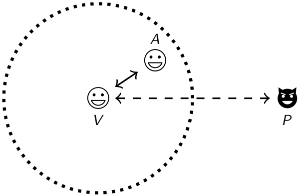
Distance-bounding attacks



(a) Mafia fraud

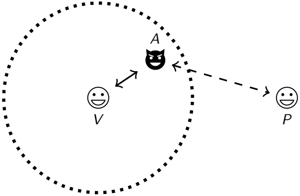


(b) Distance fraud

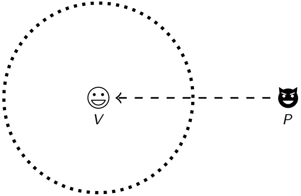


(c) Distance hijacking

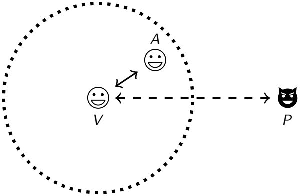
Distance-bounding attacks



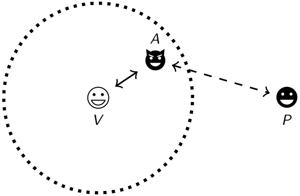
(a) Mafia fraud



(b) Distance fraud



(c) Distance hijacking



(d) Terrorist fraud

Part I

Computational Analysis of Distance-Bounding Protocols

Outline

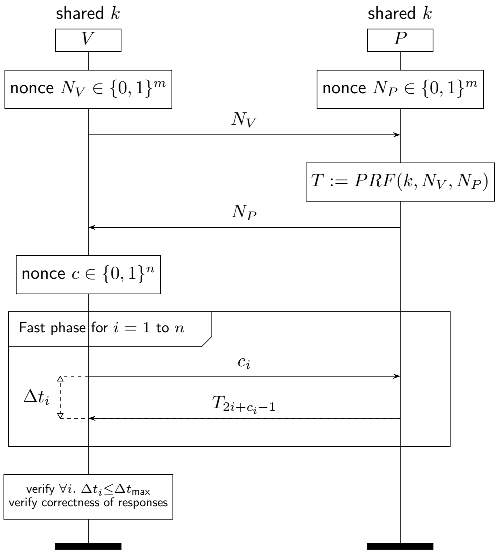
Lookup-based protocol

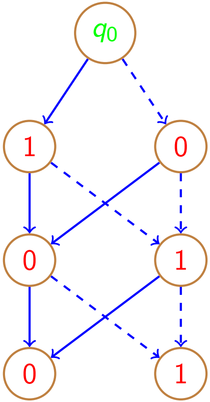
Lookup-based protocols are distance-bounding (DB) protocols such that:

1. During the fast phase, the responses to the challenges are looked-up from a table built up in the slow phase.
2. The prover does not send any messages after the fast phase has been completed.

Example

Hancke and Kuhn (HK), 2005



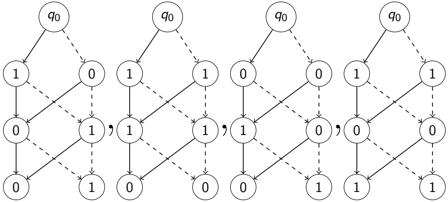


$A = (\Sigma, \Gamma, Q, q_0, \delta, \ell)$ where:

- 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

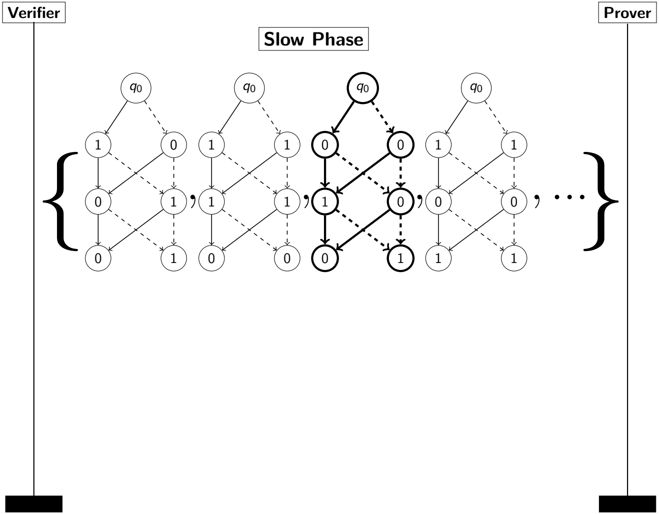
Protocol representation

$$Proto = \Big\{$$

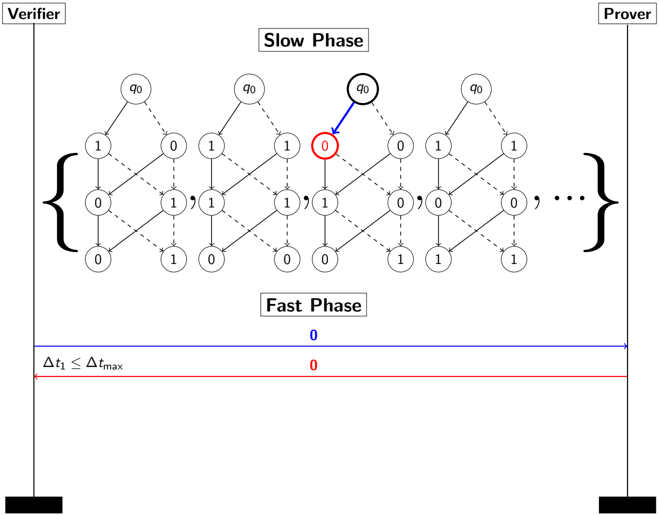


$$, \cdots \Big\}$$

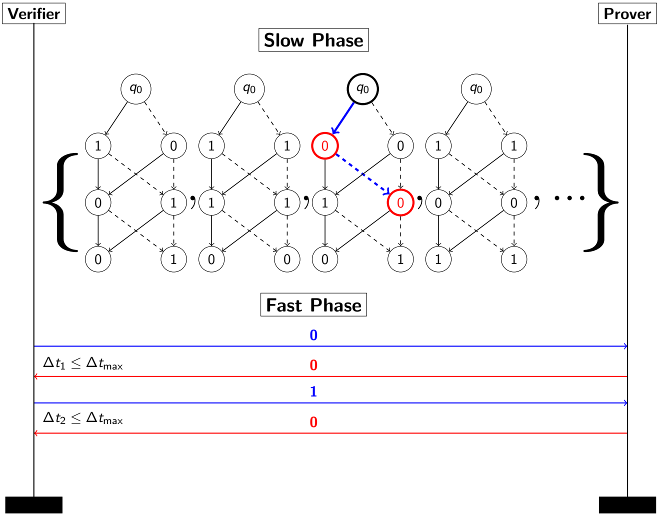
Protocol execution



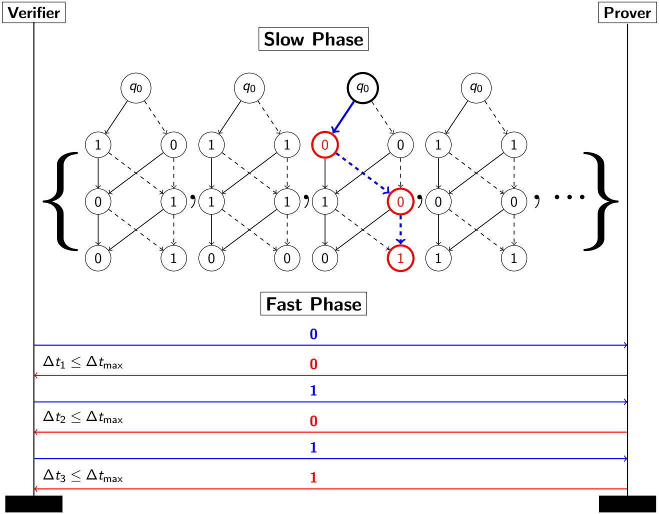
Protocol execution



Protocol execution



Protocol execution



DFA representation allowed us

- To provide an optimal adversary strategy to conduct a pre-ask mafia fraud attack against a prominent class of lookup-based protocols.
- To prove that the Tree [AT09] protocol is optimally resistant to pre-ask mafia fraud amongst all lookup-based protocols, at the cost of exponential space complexity.

Outline

The optimality goal

Definition (Optimality problem)

Given a bound h, find an *optimally resistant to mafia fraud* amongst all protocols that are layered, and random-labeled, and whose size is not larger than h.

The optimality goal

Definition (Optimality problem)

Given a bound h, find an *optimally resistant to mafia fraud* amongst all protocols that are layered, and random-labeled, and whose size is not larger than h.

- Layered is to do with two sequences of different lenghts not reaching the same state.
- Random-labeled is a property that says that there exists an automaton for any labeling of the states.
- Size is a measure of space complexity.

The optimality goal

Definition (Optimality problem)

Given a bound h, find an *optimally resistant to mafia fraud* amongst all protocols that are layered, and random-labeled, and whose size is not larger than h.

- Layered is to do with two sequences of different lenghts not reaching the same state.
- Random-labeled is a property that says that there exists an automaton for any labeling of the states.
- Size is a measure of space complexity.

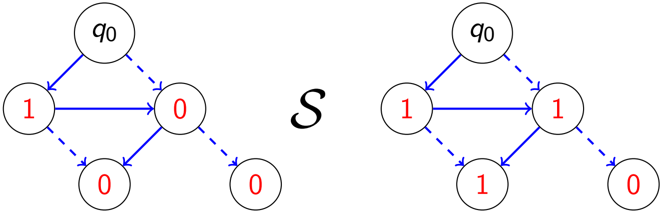
To solve the optimality problem we employed equivalence relations, and closeness and consistency in sets, and inclusion-exclusion principle.

Automata equivalence relations

Definition (State-label-insensitive relation)

The relation $\mathcal{S}$ is defined by:

$$\big((\Sigma, \Gamma, Q, q_0, \delta, \ell),\ (\Sigma, \Gamma, Q, q_0, \delta, \ell')\big) \in \mathcal{S}$$



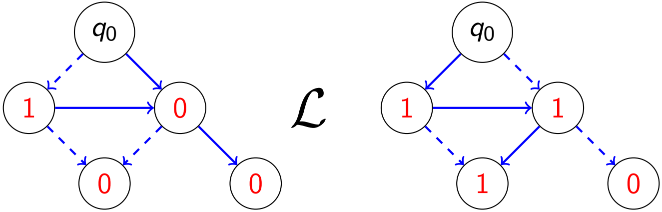
Automata equivalence relations

Definition (Label-insensitive relation)

The relation $\mathcal{L}$ is defined by:

$$\big((\Sigma, \Gamma, Q, q_0, \delta, \ell),\ (\Sigma, \Gamma, Q, q_0, \delta', \ell')\big) \in \mathcal{L}$$

such that for every $q \in Q$, a bijective function $\sigma \colon \Sigma \to \Sigma$ exists such that $\delta(q, c) = \delta'(q, \sigma(c))$ for all $c \in \Sigma$.



Consistency and closeness

- A protocol *Proto* is consistent w.r.t $\mathcal{R}$ iff

$$A, A' \in P \colon (A, A') \in \mathcal{R}$$

- A protocol *Proto* is closed under $\mathcal{R}$ iff

$$\forall (A, A') \in \mathcal{R} \colon A \in Proto \implies A' \in Proto$$

- The closure of *Proto* w.r.t $\mathcal{R}$, denoted by $Proto^{\mathcal{R}}$, is the minimal superset of *Proto* that is closed under $\mathcal{R}$.

Solving the optimality problem

Theorem ($\mathrm{Modular}$ is optimal)

For any protocol Proto that is layered and closed under $\mathcal{S}$, $A \in Proto$ exists such that:

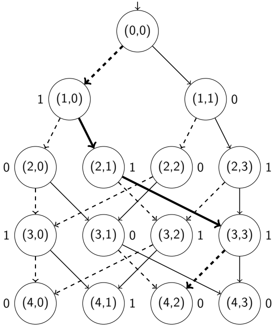
$$mafia(Proto) \geq mafia\left(\{A\}^{\mathcal{L}}\right) \geq mafia\left(\left\{M_{size(Proto)}\right\}^{\mathcal{L}}\right)$$

Solving the optimality problem

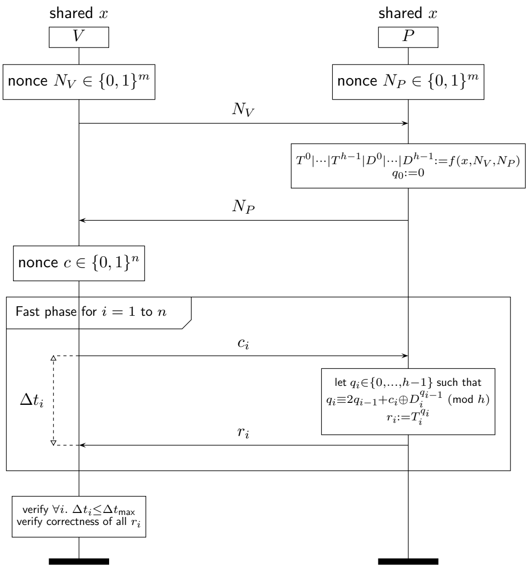
Theorem (Modular is optimal)

For any protocol Proto that is layered and closed under $\mathcal{S}$, $A \in Proto$ exists such that:

$$mafia(Proto) \geq mafia\left(\{A\}^{\mathcal{L}}\right) \geq mafia\left(\left\{M_{size(Proto)}\right\}^{\mathcal{L}}\right)$$



The Modular protocol



Summary of Part I

- Introduced a model that allows us to systematically study security and space complexity in lookup-based protocols.
- Provided formulas for computing mafia fraud success probability for most lookup-based protocols.
- Addressed (partially) the security-memory trade-off problem in a prominent class within the lookup-based protocols.
- Provided a concrete construction of a protocol that is optimally secure amongst resource-constrained protocols.

Part II

Symbolic Analysis of Distance-Bounding Protocols

Outline

Time/location model

Syntax

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

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

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

where $atom \in \mathrm{Nonce} \cup \mathrm{Agent} \cup \mathrm{Const}$ and $f \in \mathcal{F}$.

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

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

where $atom \in \text{Nonce} \cup \text{Agent} \cup \text{Const}$ and $f \in \mathcal{F}$.

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

$$e ::= \text{send}(A, m)[m'] \mid \text{recv}(A, m) \mid \text{claim}(A, B, e', e'')$$

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

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

where $atom \in \text{Nonce} \cup \text{Agent} \cup \text{Const}$ and $f \in \mathcal{F}$.

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

$$e ::= \text{send}\,(A, m)\,[m'] \mid \text{recv}\,(A, m) \mid \text{claim}\,\left(A, B, e', e''\right)$$

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

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

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

where $atom \in \text{Nonce} \cup \text{Agent} \cup \text{Const}$ and $f \in \mathcal{F}$.

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

$$e ::= \text{send}\,(A, m)\,[m'] \mid \text{recv}\,(A, m) \mid \text{claim}\,\big(A, B, e', e''\big)$$

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

Time/location model

Syntax

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

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

where $atom \in \text{Nonce} \cup \text{Agent} \cup \text{Const}$ and $f \in \mathcal{F}$.

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

$$e ::= \text{send}\,(A, m)\,[m'] \mid \text{recv}\,(A, m) \mid \text{claim}\,\left(A, B, e', e''\right)$$

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

Time/location model

Syntax

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

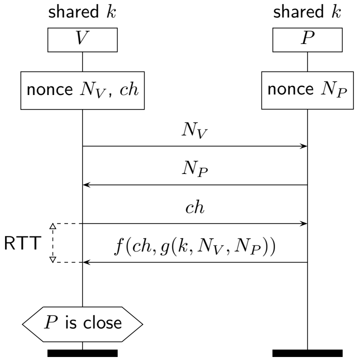
  $Proto = \{R_1 \ldots, R_n\}$ where the R_i's have the form:

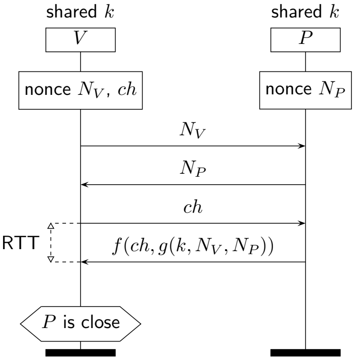
$$\frac{t \geq maxt(\alpha) \qquad A \in \text{Honest} \\ cond_1 \qquad \cdots \qquad cond_n}{(\alpha, (t, e)) \in R_i}$$

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

Time/location model

Syntax: Specifying the HK protocol





$$\frac{\alpha \in \mathit{Tr}\,(\mathit{Proto}) \quad V \in \mathsf{Honest} \quad t \geq \mathit{maxt}(\alpha) \\ N_V \in \mathsf{Nonce}_V \setminus \mathit{used}(\alpha)}{\alpha \cdot (t, \mathsf{send}\,(V, N_V)\,[\,]) \in \mathit{Tr}\,(\mathrm{HK})}$$

$$\frac{\alpha \in \mathit{Tr}\,(\mathit{Proto}) \quad P \in \mathsf{Honest} \quad t \geq \mathit{maxt}(\alpha) \\ (t', \mathsf{recv}\,(P, N_V)) \in \alpha \quad N_P \in \mathsf{Nonce}_P \setminus \mathit{used}(\alpha)}{\alpha \cdot (t, \mathsf{send}\,(P, N_P)\,[N_V]) \in \mathit{Tr}\,(\mathrm{HK})}$$

$$\frac{\alpha \in \mathit{Tr}\,(\mathit{Proto}) \quad V \in \mathsf{Honest} \quad t \geq \mathit{maxt}(\alpha) \\ (t', \mathsf{send}\,(V, N_V)\,[\,]) \in \alpha \quad (t'', \mathsf{recv}\,(V, N_P)) \in \alpha \\ ch \in \mathsf{Nonce}_V \setminus \mathit{used}(\alpha)}{\alpha \cdot (t, \mathsf{send}\,(V, ch)\,[N_V, N_P]) \in \mathit{Tr}\,(\mathrm{HK})}$$

$$\frac{\alpha \in \mathit{Tr}\,(\mathit{Proto}) \quad P \in \mathsf{Honest} \quad t \geq \mathit{maxt}(\alpha) \\ (t', \mathsf{send}\,(P, N_P)\,[N_V]) \in \alpha \quad (t'', \mathsf{recv}\,(P, ch)) \in \alpha \\ rp = f(ch, g(sh(V, P), N_V, N_P))}{\alpha \cdot (t, \mathsf{send}\,(P, rp)\,[\,]) \in \mathit{Tr}\,(\mathrm{HK})}$$

$$\frac{\alpha \in \mathit{Tr}\,(\mathit{Proto}) \quad V \in \mathsf{Honest} \quad tz \geq \mathit{maxt}(\alpha) \\ rp = f(ch, g(sh(V, P), N_V, N_P)) \\ x = \mathsf{send}\,(V, ch)\,[N_V, N_P] \quad y = \mathsf{recv}\,(V, rp) \\ (tx, x) \in \alpha \quad (ty, y) \in \alpha}{\alpha \cdot (tz, \mathsf{claim}\,(V, P, x, y)) \in \mathit{Tr}\,(\mathrm{HK})}$$

- **Message deduction**: the set $dm_A(\alpha)$ contains all messages that A can infer from α:

$$\frac{m \in initk(A)}{m \in dm_A(\alpha)} \qquad \frac{(t, \mathrm{recv}(A, m)) \in \alpha}{m \in dm_A(\alpha)} \qquad \frac{\langle m_1, m_2\rangle \in dm_A(\alpha)}{\{m_1, m_2\} \subseteq dm_A(\alpha)}$$

$$\frac{\begin{array}{c} m \in dm_A(\alpha) \\ f \in \mathcal{F} \setminus \mathcal{B} \end{array}}{f(m) \in dm_A(\alpha)} \qquad \frac{\begin{array}{c} m_1 \in dm_A(\alpha) \\ m_2 \in dm_A(\alpha) \end{array}}{\langle m_1, m_2\rangle \in dm_A(\alpha)}$$

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

Time/location model
Semantics

The set of all valid traces $Tr\,(Proto)$ is closed under the rules `Start`, `Int`, `Net` and the rules of *Proto*, where:

$$\frac{}{\epsilon \in Tr\,(Proto)}\ \texttt{Start}$$

$$\frac{\begin{array}{c} E \in \text{Dishonest} \\ t \geq maxt(\alpha) \qquad m \in dm_E\,(\alpha) \end{array}}{\alpha \cdot (t, \text{send}\,(E, m)\,[\,]) \in Tr\,(Proto)}\ \texttt{Int}$$

$$\frac{\begin{array}{c} t \geq maxt(\alpha) \\ \langle t', \text{send}\,(A, m)\,[m'] \rangle \in \alpha \\ t \geq t' + dist(A, B)/\text{c} \end{array}}{\alpha \cdot (t, \text{recv}\,(B, m)) \in Tr\,(Proto)}\ \texttt{Net}$$

Definition

A distance-bounding protocol *Proto* is secure if and only if:

$$\forall \alpha \in \mathit{Tr}\left(\mathit{Proto}\right), \left(t, \mathrm{claim}\left(V, P, x, y\right)\right) \in \alpha.$$
$$\exists (tx, x), (ty, y) \in \alpha.$$
$$\mathit{dist}(V, P) \leq \mathrm{c} \cdot \frac{ty - tx}{2}$$

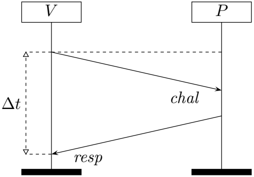
Definition

A distance-bounding protocol *Proto* is secure if and only if:

$$\forall \alpha \in \mathit{Tr}\left(\mathit{Proto}\right), \left(t, \text{claim}\left(V, P, x, y\right)\right) \in \alpha.$$
$$\exists (tx, x), (ty, y) \in \alpha, P' \in \mathit{actor}\left(\alpha\right).$$
$$\mathit{dist}(V, P') \leq \text{c} \cdot \frac{ty - tx}{2} \wedge P \approx P'$$

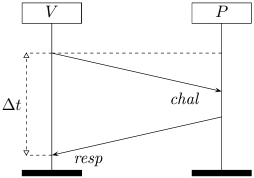
where $\approx = \{(A, A) \mid A \in \text{Honest}\} \cup \text{Dishonest} \times \text{Dishonest}.$

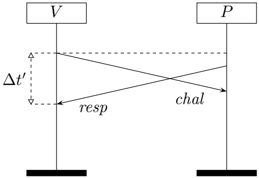
Three timing scenarios



Correct timing

Three timing scenarios

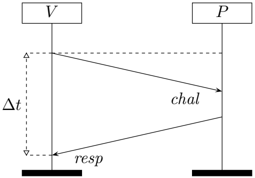




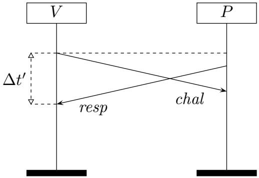
Correct timing

Early timing

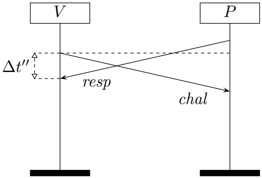
Three timing scenarios



Correct timing

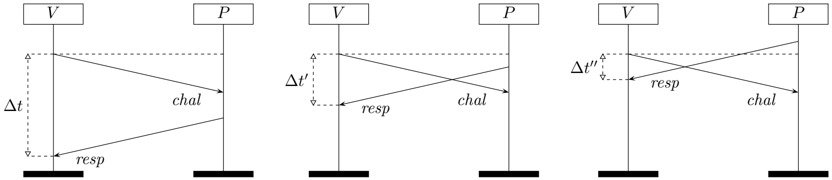


Early timing



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.

Causality-based security property

Theorem (Causality-based secure DB)

A distance-bounding protocol Proto is distance-bounding secure if and only if:

$$\forall \sigma \in \pi(Tr\,(Proto)), \mathsf{claim}\,(V, P, x, y) \in \sigma.$$
$$\exists x \cdot e \cdot y \sqsubseteq \sigma.\ actor\,(e) = P.$$

In words: Whenever V claims that P is close during the fast phase delimited by x and y, it is the case that P was alive in such phase.

Theorem (Causality-based secure DB)

A distance-bounding protocol Proto is distance-bounding secure if and only if:

$$\forall \sigma \in \pi(Tr\,(Proto)), \mathrm{claim}\,(V, P, x, y) \in \sigma.$$
$$\exists x \cdot e \cdot y \sqsubseteq \sigma.\ actor\,(e) \approx P.$$

In words: Whenever V claims that P is close during the fast phase delimited by x and y, it is the case that P was alive in such phase, or a compromised P' was, if P is compromised.

Verification results

Protocol	Satisfies `dbsec`	Attack Found
BC-Signature	×	DH
BC-FiatShamir	$\times^{(n)}$	$\text{DH}^{(n)}$, $\text{DF}^{(n)}$
BC-Schnorr	$\times^{(n)}$	$\text{DH}^{(n)}$, $\text{DF}^{(n)}$
CRCS	$\times^{(n)}$	$\text{DH}^{(n)}$
Lookup-based		
• Tree	✓	-
• Poulidor	✓	-
• Hancke-Kuhn	✓	-
• Uniform	✓	-
Meadows et al.	×	DH
Kim-Avoine	$\checkmark^{(n)}$	-
Munilla-Peinado	$\checkmark^{(n)}$	-
Reid et al.	$\checkmark^{(n)}$	-
Swiss-Knife	$\checkmark^{(n)}$	-
TREAD-PK	$\times^{(n)}$	$\text{DH}^{(n)}$, $\text{MF}^{(n)}$
TREAD-SH	$\times^{(n)}$	$\text{DH}^{(n)}$
PaySafe	$\times^{(n)}$	$\text{DF}^{(n)}$, $\text{DH}^{(n)}$

Summary on causality

- Proved that distance-bounding security can be formulated through causality, like most other security properties.
  - Led to simplification and more effective tooling.
    (e.g. BC protocol is 650 Isabelle/HOL LoC vs. 180 Tamarin LoC).
  - Provided the first fully automated verification framework.
    verification.
- Provided computer-verifiable (in)security proofs for a number of state-of-the-art protocols.
  - Identified unreported vulnerabilities in two recently published protocols: PaySafe (FC'15) and TREAD (AsiaCCS'17).

Outline

Multiset rewriting system

- The execution of a protocol starts with the empty multiset of facts, and evolves through *multiset rewriting rules*.

Multiset rewriting system

- The execution of a protocol starts with the empty multiset of facts, and evolves through *multiset rewriting rules*.
- A multiset rewriting rule is a tuple (p, a, c), written as $[p]\xrightarrow{a}[c]$, where p, a and c are sequences of facts called the *premises*, the *actions*, and the *conclusions* of the rule, respectively. E.g.

$$\begin{bmatrix}\mathrm{Funds}(\mathit{Person}, \mathit{funds}),\\ \mathrm{Price}(\mathit{Good}, \mathit{price})\end{bmatrix} \xrightarrow{\substack{\mathrm{Geq}(\mathit{funds}, \mathit{price}),\\ \mathrm{Purchase}(\mathit{Person}, \mathit{Good}),\\ \mathrm{Happy}(\mathit{Person})}} [\,\mathrm{Funds}(\mathit{Person}, \mathit{sub}(\mathit{funds}, \mathit{price}))]$$

$$\begin{bmatrix}\mathrm{Salary}(\mathit{Person}, \mathit{salary}),\\ \mathrm{PayDay}(\mathit{Person}),\\ \mathrm{Funds}(\mathit{Person}, \mathit{funds})\end{bmatrix} \xrightarrow{\substack{\mathrm{PaySalary}(\mathit{Person}),\\ \mathrm{EvenHappier}(\mathit{Person})}} [\mathrm{Funds}(\mathit{Person}, \mathit{funds} + \mathit{salary})]$$

Multiset rewriting system

- The execution of a protocol starts with the empty multiset of facts, and evolves through *multiset rewriting rules*.
- A multiset rewriting rule is a tuple (p, a, c), written as $[p]\xrightarrow{a}[c]$, where p, a and c are sequences of facts called the *premises*, the *actions*, and the *conclusions* of the rule, respectively. E.g.

$$\begin{bmatrix}\text{Funds}(\mathit{Person}, \mathit{funds}),\\ \text{Price}(\mathit{Good}, \mathit{price})\end{bmatrix} \xrightarrow[]{\substack{\text{Geq}(\mathit{funds},\mathit{price}),\\ \text{Purchase}(\mathit{Person},\mathit{Good}),\\ \text{Happy}(\mathit{Person})}} \left[\ \text{Funds}(\mathit{Person}, \mathit{sub}(\mathit{funds}, \mathit{price}))\right]$$

$$\begin{bmatrix}\text{Salary}(\mathit{Person}, \mathit{salary}),\\ \text{PayDay}(\mathit{Person}),\\ \text{Funds}(\mathit{Person}, \mathit{funds})\end{bmatrix} \xrightarrow[]{\substack{\text{PaySalary}(\mathit{Person}),\\ \text{EvenHappier}(\mathit{Person})}} [\text{Funds}(\mathit{Person}, \mathit{funds} + \mathit{salary})]$$

Consider only traces t that satisfy

$$\forall x, y.\ \text{Geq}(x, y) \in t \implies \exists z.\ y + z = x$$

Multiset rewriting system

- A set R of multiset rewriting rules defines a *multiset rewriting system*: an LTS whose set of states is $\mathcal{G}^\sharp$ and whose transition relation $\rightarrow_R \subseteq \mathcal{G}^\sharp \times \mathcal{P}(\mathcal{G}) \times \mathcal{G}^\sharp$ is defined by:

$$
\begin{aligned}
& S \xrightarrow{l}_R S' \iff \\
& \quad \exists (p, a, c) \in_E \mathit{ginsts}(R). \\
& \qquad l = \mathit{set}(a) \wedge \mathit{linear}(p) \subseteq^\sharp S \wedge \mathit{persist}(p) \subseteq \mathit{set}(S) \wedge \\
& \qquad S' = \left(S \setminus^\sharp \mathit{linear}(p)\right) \cup^\sharp \mathit{multiset}(c).
\end{aligned}
$$

Multiset rewriting system

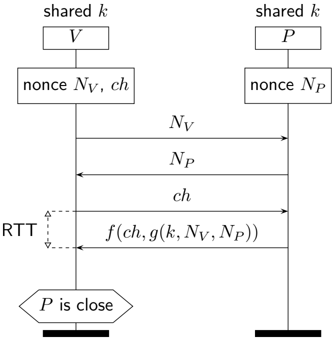
- A set R of multiset rewriting rules defines a *multiset rewriting system*: an LTS whose set of states is $\mathcal{G}^\sharp$ and whose transition relation $\rightarrow_R \subseteq \mathcal{G}^\sharp \times \mathcal{P}(\mathcal{G}) \times \mathcal{G}^\sharp$ is defined by:

$$
\begin{aligned}
&S \xrightarrow{l}_R S' \iff \\
&\quad \exists (p, a, c) \in_E ginsts(R). \\
&\qquad l = set(a) \wedge linear(p) \subseteq^\sharp S \wedge persist(p) \subseteq set(S) \wedge \\
&\qquad S' = \left(S \setminus^\sharp linear(p)\right) \cup^\sharp multiset(c).
\end{aligned}
$$

- An *execution* of R is a finite alternating sequence of states and labels $[S_0, l_1, S_1, \ldots, l_n, S_n]$ of states and labels such that:
  - $S_0 = \emptyset^\sharp$,
  - $S_{i-1} \xrightarrow{l_i}_R S_i$ for $1 \leq i \leq n$, and
  - if $S_{i+1} \setminus^\sharp S_i = \{\mathrm{Fr}(x)\}^\sharp$ for some i and x, then $j \neq i$ does not exist such that $S_{j+1} \setminus^\sharp S_j = \{\mathrm{Fr}(x)\}^\sharp$.

Protocol specification

Hancke and Kuhn, 2005



$\mathtt{KeyGen}, \mathtt{KeyRevV}, \mathtt{KeyRevP}$

$$\mathtt{DBNet} := [\mathrm{Send}(X, m)] \xrightarrow{\mathrm{Action}(Y),\ \mathrm{Recv}(Y, m)} \begin{bmatrix} \mathrm{Out}(m), \\ \mathrm{Recv}(Y, m) \end{bmatrix}$$

$$\mathtt{DBAdv} := [\mathrm{In}(m), \mathrm{KeyComp}(X)] \xrightarrow{\mathrm{Action}(X)} [\mathrm{Send}(X, m)]$$

$$\mathtt{V1} := [\mathrm{Fr}(N_V)] \xrightarrow{\mathrm{Start}(N_V)} [\mathrm{Out}(N_V), \mathrm{VerifSt1}(V, N_V)]$$

$$\mathtt{P1} := \begin{bmatrix} \mathrm{Fr}(N_P), \mathrm{In}(N_V), \\ \mathrm{Shk}(V, P, k) \end{bmatrix} \xrightarrow{\substack{\mathrm{Start}(N_P), \\ \mathrm{Action}(P)}} \begin{bmatrix} \mathrm{Send}(P, N_P), \\ \mathrm{ProvSt1}(P, k, N_P, N_V) \end{bmatrix}$$

$$\mathtt{V2} := \begin{bmatrix} \mathrm{VerifSt1}(V, N_V), \\ \mathrm{Fr}(ch), \mathrm{In}(N_P), \\ \mathrm{Shk}(V, P, k) \end{bmatrix} \xrightarrow{\mathrm{Send}(V, ch)} \begin{bmatrix} \mathrm{Out}(ch), \\ \mathrm{VerifSt2}(V, P, N_V, ch, \\ f(ch, g(k, N_V, N_P))) \end{bmatrix}$$

$$\mathtt{P2} := \begin{bmatrix} \mathrm{ProvSt1}(P, k, N_P, N_V), \\ \mathrm{In}(ch) \end{bmatrix} \xrightarrow{\substack{\mathrm{Action}(P), \\ \mathrm{End}(N_P)}} \begin{bmatrix} \mathrm{Send}(P, f(ch, \\ g(k, N_P, N_V))) \end{bmatrix}$$

$$\mathtt{V3} := \begin{bmatrix} \mathrm{VerifSt2}(V, P, N_V, ch, rp), \\ \mathrm{Recv}(V, rp) \end{bmatrix} \xrightarrow{\mathrm{DBSec}(V, P, ch, rp), \mathrm{End}(N_V)} [\ ]$$

$$\mathrm{HK} = \{\mathtt{KeyGen}, \mathtt{KeyRevV}, \mathtt{KeyRevP}, \mathtt{DBNet}, \mathtt{DBAdv}, \mathtt{V1}, \mathtt{V2}, \mathtt{V3}, \mathtt{P1}, \mathtt{P2}\}$$

Protocol execution

- The set of all executions *Proto* is $[\![Proto \cup \mathcal{I}]\!]$ where:

$$
\begin{aligned}
\mathcal{I} = \{\ & \texttt{Fresh} := [\ ] \rightarrow [\mathsf{Fr}(x\colon \mathit{fresh})]\ ,\\
& \texttt{Learn} := [\mathsf{Out}(x)] \rightarrow [\mathsf{K}(x)]\ ,\\
& \texttt{Inject} := [\mathsf{K}(x)] \xrightarrow{\mathsf{K}(x)} [\mathsf{In}(x)]\ ,\\
& \texttt{AdvFresh} := [\mathsf{Fr}(x)] \rightarrow [\mathsf{K}(x)]\ ,\\
& \texttt{Public} := [\ ] \rightarrow [\mathsf{K}(x\colon \mathit{pub})]\ ,\\
& \texttt{Funct} := [\mathsf{K}(x_1), \ldots, \mathsf{K}(x_n)] \rightarrow [\mathsf{K}(f(x_1, \ldots, x_n))]\ \}
\end{aligned}
$$

- Given an execution $[S_0, l_1, S_1, \ldots, l_n, S_n]$, the sequence $l_1 \cdots l_n$ is the *trace*.
- $Tr(Proto) = \{l_1 \cdots l_n \mid [S_0, l_1, S_1, \ldots, l_n, S_n] \in [\![Proto \cup \mathcal{I}]\!]\}$.

Security properties

Definition (Security Property)

A *security property* φ is a relation from traces to natural numbers, and $\varphi(t, i)$ means that the claims of φ in t_i are valid.

- E.g. secure distance-bounding is defined as:

$$
\begin{aligned}
& dbsec(t, l) \iff \\
& \quad \forall V, P, ch, rp.\ \mathrm{DBSec}(V, P, ch, rp) \in t_l \implies \\
& \qquad (\exists i, j, k.\ i < j < k \wedge \mathrm{Send}(V, ch) \in t_i \wedge \\
& \qquad\qquad \mathrm{Action}(P) \in t_j \wedge \mathrm{Recv}(V, rp) \in t_k) \vee \\
& \qquad (\exists b, b', i, j, k, P'. \\
& \qquad\qquad i < j < k \wedge \mathrm{Send}(V, ch) \in t_i \wedge \\
& \qquad\qquad \mathrm{Action}(P') \in t_j \wedge \mathrm{Recv}(V, rp) \in t_k \wedge \\
& \qquad\qquad \mathrm{KeyComp}(P) \in t_b \wedge \mathrm{KeyComp}(P') \in t_{b'}) \vee \\
& \qquad (\exists i.\ \mathrm{KeyComp}(V) \in t_i)
\end{aligned}
$$

Security properties

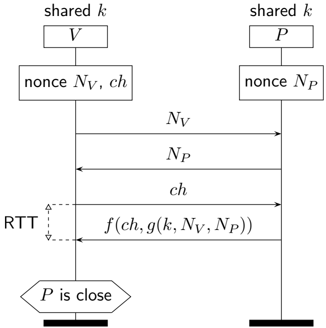
Definition (Security)

A set *Proto* of protocol rules *satisfies* a security property φ, denoted $\mathit{Proto} \models \varphi$, if $\forall t \in \mathit{Tr}\left(\mathit{Proto}\right), i \in \{1, \ldots, |t|\}.\ \varphi(t, i)$.

Security properties

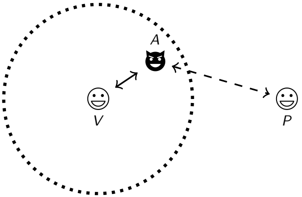
Definition (Security)

A set *Proto* of protocol rules *satisfies* a security property φ, denoted $\mathit{Proto} \models \varphi$, if $\forall t \in \mathit{Tr}\left(\mathit{Proto}\right), i \in \{1, \ldots, |t|\}.\ \varphi(t, i)$.

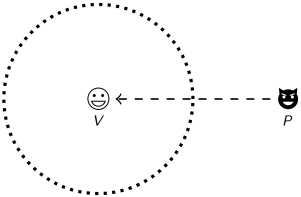


$$\mathrm{HK} \models \mathit{dbsec}$$

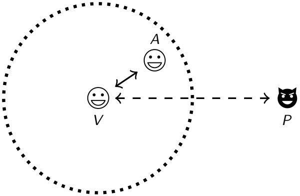
i.e. no MF, DF or DH exist



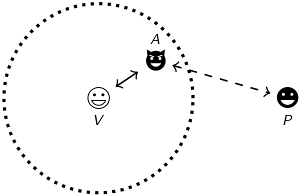
(a) Mafia fraud



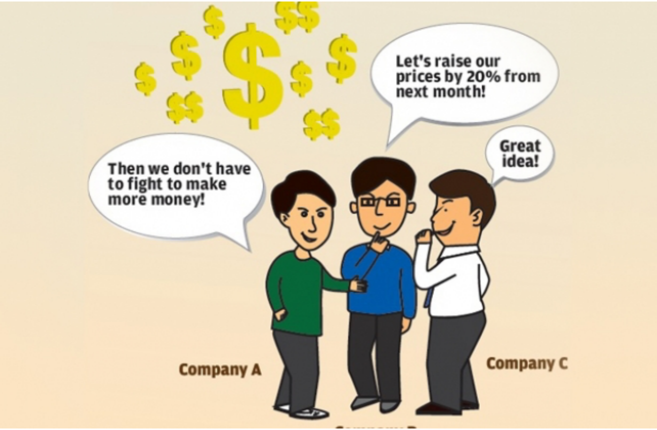
(b) Distance fraud



(c) Distance hijacking

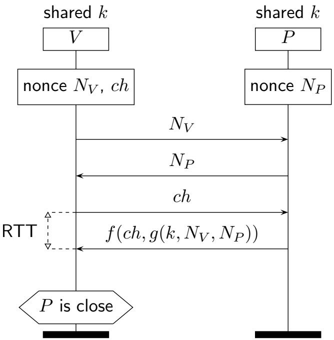


(d) Terrorist fraud



Source https://yp.scmp.com

Hancke and Kuhn, 2005



KeyGen, KeyRevV, KeyRevP

$$\texttt{DBNet} := [\mathsf{Send}(X, m)] \xrightarrow{\mathsf{Action}(Y),\ \mathsf{Recv}(Y, m)} \begin{bmatrix} \mathsf{Out}(m), \\ \mathsf{Recv}(Y, m) \end{bmatrix}$$

$$\texttt{DBAdv} := [\mathsf{In}(m), \mathsf{KeyComp}(X)] \xrightarrow{\mathsf{Action}(X)} [\mathsf{Send}(X, m)]$$

$$\texttt{V1} := [\mathsf{Fr}(N_V)] \xrightarrow{\mathsf{Start}(N_V)} [\mathsf{Out}(N_V), \mathsf{VerifSt1}(V, N_V)]$$

$$\texttt{P1} := \begin{bmatrix} \mathsf{Fr}(N_P), \mathsf{In}(N_V), \\ \mathsf{Shk}(V, P, k) \end{bmatrix} \xrightarrow{\substack{\mathsf{Start}(N_P), \\ \mathsf{Action}(P)}} \begin{bmatrix} \mathsf{Send}(P, N_P), \\ \mathsf{ProvSt1}(P, k, N_P, N_V) \end{bmatrix}$$

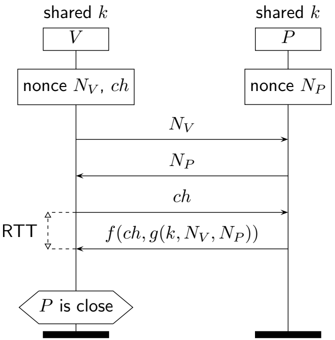
$$\texttt{V2} := \begin{bmatrix} \mathsf{VerifSt1}(V, N_V), \\ \mathsf{Fr}(ch), \mathsf{In}(N_P), \\ \mathsf{Shk}(V, P, k) \end{bmatrix} \xrightarrow{\mathsf{Send}(V, ch)} \begin{bmatrix} \mathsf{Out}(ch), \\ \mathsf{VerifSt2}(V, P, N_V, ch, \\ f(ch, g(k, N_V, N_P))) \end{bmatrix}$$

$$\texttt{P2} := \begin{bmatrix} \mathsf{ProvSt1}(P, k, N_P, N_V), \\ \mathsf{In}(ch) \end{bmatrix} \xrightarrow{\substack{\mathsf{Action}(P), \\ \mathsf{End}(N_P)}} \begin{bmatrix} \mathsf{Send}(P, f(ch, \\ g(k, N_P, N_V))) \end{bmatrix}$$

$$\texttt{V3} := \begin{bmatrix} \mathsf{VerifSt2}(V, P, N_V, ch, rp), \\ \mathsf{Recv}(V, rp) \end{bmatrix} \xrightarrow{\mathsf{DBSec}(V, P, ch, rp), \mathsf{End}(N_V)} [\ ]$$

Modeling collusion

Hancke and Kuhn, 2005



$\texttt{KeyGen}, \texttt{KeyRevV}, \texttt{KeyRevP}$

$$\texttt{DBNet} := [\mathsf{Send}(X, m)] \xrightarrow{\mathsf{Action}(Y),\ \mathsf{Recv}(Y, m)} \begin{bmatrix} \mathsf{Out}(m), \\ \mathsf{Recv}(Y, m) \end{bmatrix}$$

$$\texttt{DBAdv} := [\mathsf{In}(m), \mathsf{KeyComp}(X)] \xrightarrow{\mathsf{Action}(X)} [\mathsf{Send}(X, m)]$$

$$\texttt{V1} := [\mathsf{Fr}(N_V)] \xrightarrow{\mathsf{Start}(N_V)} [\mathsf{Out}(N_V), \mathsf{VerifSt1}(V, N_V)]$$

$$\texttt{P1} := \begin{bmatrix} \mathsf{Fr}(N_P), \mathsf{In}(N_V), \\ \mathsf{Shk}(V, P, k) \end{bmatrix} \xrightarrow{\substack{\mathsf{Start}(N_P), \\ \mathsf{Action}(P)}} \begin{bmatrix} \mathsf{Send}(P, N_P), \\ \mathsf{ProvSt1}(P, k, N_P, N_V) \end{bmatrix}$$

$$\texttt{Coll} := [\mathsf{ProvSt1}(P, k, N_P, N_V)] \xrightarrow{\mathsf{Collusion}()} \begin{bmatrix} \mathsf{ProvSt1}(P, k, N_P, N_V), \\ \mathsf{Out}(g(k, N_V, N_P)) \end{bmatrix}$$

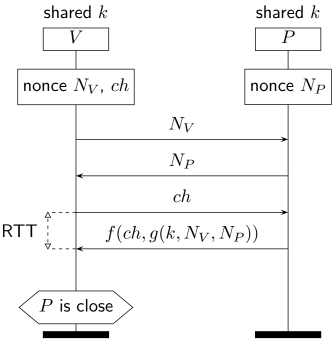
$$\texttt{V2} := \begin{bmatrix} \mathsf{VerifSt1}(V, N_V), \\ \mathsf{Fr}(ch), \mathsf{In}(N_P), \\ \mathsf{Shk}(V, P, k) \end{bmatrix} \xrightarrow{\mathsf{Send}(V, ch)} \begin{bmatrix} \mathsf{Out}(ch), \\ \mathsf{VerifSt2}(V, P, N_V, ch, \\ f(ch, g(k, N_V, N_P))) \end{bmatrix}$$

$$\texttt{P2} := \begin{bmatrix} \mathsf{ProvSt1}(P, k, N_P, N_V), \\ \mathsf{In}(ch) \end{bmatrix} \xrightarrow{\substack{\mathsf{Action}(P), \\ \mathsf{End}(N_P)}} \begin{bmatrix} \mathsf{Send}(P, f(ch, \\ g(k, N_P, N_V))) \end{bmatrix}$$

$$\texttt{V3} := \begin{bmatrix} \mathsf{VerifSt2}(V, P, N_V, ch, rp), \\ \mathsf{Recv}(V, rp) \end{bmatrix} \xrightarrow{\mathsf{DBSec}(V, P, ch, rp), \mathsf{End}(N_V)} [\ ]$$

Modeling collusion

Hancke and Kuhn, 2005



KeyGen, KeyRevV, KeyRevP

$$\mathtt{DBNet} := [\mathsf{Send}(X, m)] \xrightarrow{\mathsf{Action}(Y),\ \mathsf{Recv}(Y, m)} \begin{bmatrix} \mathsf{Out}(m), \\ \mathsf{Recv}(Y, m) \end{bmatrix}$$

$$\mathtt{DBAdv} := [\mathsf{In}(m), \mathsf{KeyComp}(X)] \xrightarrow{\mathsf{Action}(X)} [\mathsf{Send}(X, m)]$$

$$\mathtt{V1} := [\mathsf{Fr}(N_V)] \xrightarrow{\mathsf{Start}(N_V)} [\mathsf{Out}(N_V), \mathsf{VerifSt1}(V, N_V)]$$

$$\mathtt{P1} := \begin{bmatrix} \mathsf{Fr}(N_P), \mathsf{In}(N_V), \\ \mathsf{Shk}(V, P, k) \end{bmatrix} \xrightarrow{\mathsf{Start}(N_P),\ \mathsf{Action}(P)} \begin{bmatrix} \mathsf{Send}(P, N_P), \\ \mathsf{ProvSt1}(P, k, N_P, N_V) \end{bmatrix}$$

$$\mathtt{Coll} := [\mathsf{ProvSt1}(P, k, N_P, N_V)] \xrightarrow{\mathsf{Collusion}()} \begin{bmatrix} \mathsf{ProvSt1}(P, k, N_P, N_V), \\ \mathsf{Out}(g(k, N_V, N_P)) \end{bmatrix}$$

$$\mathtt{V2} := \begin{bmatrix} \mathsf{VerifSt1}(V, N_V), \\ \mathsf{Fr}(ch), \mathsf{In}(N_P), \\ \mathsf{Shk}(V, P, k) \end{bmatrix} \xrightarrow{\mathsf{Send}(V, ch)} \begin{bmatrix} \mathsf{Out}(ch), \\ \mathsf{VerifSt2}(V, P, N_V, ch, \\ f(ch, g(k, N_V, N_P))) \end{bmatrix}$$

$$\mathtt{P2} := \begin{bmatrix} \mathsf{ProvSt1}(P, k, N_P, N_V), \\ \mathsf{In}(ch) \end{bmatrix} \xrightarrow{\mathsf{Action}(P),\ \mathsf{End}(N_P)} \begin{bmatrix} \mathsf{Send}(P, f(ch, \\ g(k, N_P, N_V))) \end{bmatrix}$$

$$\mathtt{V3} := \begin{bmatrix} \mathsf{VerifSt2}(V, P, N_V, ch, rp), \\ \mathsf{Recv}(V, rp) \end{bmatrix} \xrightarrow{\mathsf{DBSec}(V, P, ch, rp), \mathsf{End}(N_V)} [\ ]$$

We obtain $\mathrm{HK} \cup \{\mathtt{Coll}\} \not\models$ *dbsec* as opposed to $\mathrm{HK} \models$ *dbsec*

Post-collusion security

Definition (Post-collusion security)

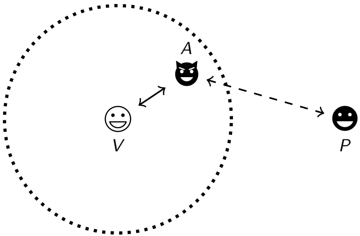
Given a protocol *Proto*, a valid extension $Proto' \supseteq Proto$ is *post-collusion secure w.r.t.* φ, denoted $Proto' \models^{\star} \varphi$, if:

$$
\begin{aligned}
&\forall t \in Tr\left(Proto'\right), e \in \{1, \ldots, |t|\}.\\
&\quad (complete(t_1 \cdots t_e) \wedge nocollusion(t_{e+1} \cdots t_{|t|}))\\
&\qquad \Longrightarrow \forall i > e.\ \varphi(t, i).
\end{aligned}
$$

Terrorist fraud

Definition (Terrorist Fraud Attack – Informal)

Terrorist fraud is an attack in which a remote and non-compromised prover P **colludes** with a close and compromised prover A to make the verifier believe that P is close. Conditionally, A must not be able to prove the same again without further collusion.



Terrorist fraud

Definition (Terrorist Fraud Attack – Informal)

Terrorist fraud is an attack in which a remote and non-compromised prover P **colludes** with a close and compromised prover A to make the verifier believe that P is close. Conditionally, A must not be able to prove the same again without further collusion.

Definition (Resistance to Terrorist Fraud)

A protocol *Proto* is *resistant to terrorist fraud* if for every valid extension $Proto' \supseteq Proto$ it holds that:

$$Proto' \not\models dbsec\_hnst \implies Proto' \not\models^{\star} dbsec\_hnst.$$

Verification results

Protocol	**Satisfies** *dbsec_hnst*	**Satisfies** *dbsec*	**Resists TF**
Brands-Chaum			
• Signature id.	✓	×	$\times^{(n)}$
• Fiat-Shamir id.	✓	×	$\times^{(n)}$
CRCS			
• Non-reveal sign.	✓	✓	×
• Reveal sign.	✓	×	×
Meadows et al.			
• $\langle N_V, P \oplus N_P \rangle$	✓	$\times^{(\neq c)}$	×
• $N_V \oplus h(P, N_P)$	$\checkmark^{(n)}$	$\checkmark^{(n)}$	$\times^{(n)}$
• $\langle N_V, P, N_P \rangle$	$\checkmark^{(n)}$	$\checkmark^{(n)}$	$\times^{(n)}$
Lookup-based			
• Tree	✓	✓	$\times^{(\neq c)}$
• Poulidor	✓	✓	$\times^{(\neq c)}$
• Hancke-Kuhn	✓	✓	$\times^{(\neq c)}$
• Uniform	✓	✓	$\times^{(\neq c)}$
Munilla-Peinado	✓	✓	$\times^{(n)}$
Kim-Avoine	✓	✓	$\times^{(n)}$

Protocol	**Satisfies** *dbsec_hnst*	**Satisfies** *dbsec*	**Resists TF**
Reid et al.	✓	✓	$\checkmark^{(n)}$
MAD (one way)	✓	$\times^{(\neq c)}$	×
DBPK	$\checkmark^{(n)}$	$\checkmark^{(n)}$	$\checkmark^{(n)}$
Swiss Knife	✓	✓	$\checkmark^{(n)}$
UWB			
• PKI	$\times^{(n)}$	$\times^{(n)}$	$\checkmark^{(n)}$
• keyed-MAC	$\times^{(n)}$	$\times^{(n)}$	$\checkmark^{(n)}$
WSBC+DB	$\checkmark^{(n)}$	$\times^{(n)}$	$\times^{(n)}$
Hitomi	$\checkmark^{(n)}$	$\checkmark^{(n)}$	$\times^{(n)}$
TREAD			
• Asymmetric	×	×	$\checkmark^{(n)}$
• Symmetric	✓	×	$\checkmark^{(n)}$
ISO/IEC 14443			
• PaySafe	✓	×	×
• MIFARE Plus	✓	×	×
• PayPass	✓	×	×

Summary of Part II

- First causality-based secure DB property.
- A concrete formalism to model collusion in security protocols.
- Introduced the notion of post-collusion security.
- Provided a formal definition of TF resistance.
- A comprehensive security survey of DB protocols.

Outline

Conclusions

- A computational model that allows for comprehensive security and space complexity analysis.
- An optimally secure protocol for a prominent class of lookup-based protocols, given an upper bound on the size.
- A causality-based, automatic symbolic framework for DB verification that accounts for the four classes of attacks.
- An extensive security survey of DB protocols, including Mastercard's PayPass protocol and NXP's MIFARE Plus protocol.

Publications

Related to Part I

- **A Class of Precomputation-Based Distance-Bounding Protocols**, with S. Mauw, and R. Trujillo-Rasua. In *1st IEEE European Symposium on Security and Privacy*, EuroS&P'16, Saarbrüecken, Germany, March 21-24, 2016. pp. 97–111.
- **Optimality Results on the Security of Lookup-Based Protocols**, with S. Mauw, and R. Trujillo-Rasua. In *Radio Frequency Identification and IoT Security*, RFIDSec'16, Hong Kong, China, Nov. 30 - Dec. 2, 2016. pp. 137–150.

Related to Part II

- **Distance-Bounding Protocols: Verification without Time and Location**, with S. Mauw, Z. Smith, and R. Trujillo-Rasua. In *39th IEEE Symposium on Security and Privacy*, S&P'18, San Francisco, California, May 21–23, 2018, USA. pp. 549–566.
- **Post-Collusion Security and Distance Bounding**, with S. Mauw, Z. Smith, and R. Trujillo-Rasua (under submission).

Not related to DB

- **Automated Identification of Desynchronisation Attacks on Shared Secrets**, with S. Mauw, Z. Smith, and R. Trujillo-Rasua. In *23rd European Symposium on Research in Computer Security*, ESORICS'18, Barcelona, Spain, Sept. 3–7, 2018. pp. 406–426.

Future Work

- Extend the computational analysis in order to account for further attacks.
- Proof of *completeness* for our TF resistance definition in relation to the Tamarin prover.
  - Seems quite complex, yet we have some promising ideas.
- Reduce the gap between computational and symbolic analysis.
  - Build "stochastic reasoning" on top of multiset rewriting.

Thank you

Gracias • Merci • Danke

Jorge Toro Pozo
jorgetp.github.io