Existential Assertions For Voting Protocols

R Ramanujam, Vaishnavi Sundararajan, S P Suresh

Introduction

- Desirable properties for voting protocols — Eligibility, Anonymity, Fairness, Receipt-Freeness etc.
- Anonymity — voter-vote relationship should be secret.
- Verifying properties: symbolically model, check for logical flaws.
- We present a system which makes verification for anonymity easier. Running example: FOO protocol.

DY83: Dolev, D.; Yao, A. C. (1983), "On the security of public key protocols", *IEEE Transactions on Information Theory*, IT-29: 198–208.

FOO Voting Protocol

- Proposed by Fujioka, Okamoto and Ohta in 1992. [FOO92]
- Voter contacts admin, who checks voter's id and authenticates.
- Authenticated voter then sends vote anonymously to collector.
- Admin should not know vote, collector should not know id.
- Terms-only model ensures this via blind signatures.

FOO92: Fujioka, A.; Okamoto, T.; Ohta, K. (1992), "A Practical Secret Voting Scheme for Large Scale Elections", *Advances in Cryptology — AUSCRYPT '92*, 244–251.

FOO Protocol: Terms-Only

$V \rightarrow A \quad : \quad V, \{\text{blind}(\{v\}_r, b)\}_{sd(V)}$

$A \rightarrow V \quad : \quad \{\text{blind}(\{v\}_r, b)\}_{sd(A)}$

$V \nrightarrow C \quad : \quad \{\{v\}_r\}_{sd(A)}$

$C \rightarrow \quad : \quad list, \{\{v\}_r\}_{sd(A)}$

$V \rightarrow C \quad : \quad r$

$$\text{unblind}(\{\text{blind}(t, b)\}_{sd(A)}, b) = \{t\}_{sd(A)}$$

FOO Protocol: What We Want

$V \rightarrow A$: $\{v\}_k$, *"V wants to vote with this term, an enc of valid vote"*

$A \rightarrow V$: *"V is eligible and wants to vote with the term shown earlier"*

$V \looparrowright C$: $\{v\}_{k'}$, *"Some eligible agent was authorised by A to vote with a valid vote, this term is a re-enc of that same vote."*

A does not have to modify *V*'s term (which contains the vote) in order to certify it!

FOO Protocol: Assertions

$V \rightarrow A \quad : \quad \{v\}_{r_A}, V \; says \; \{\exists x, r : \{x\}_r = \{v\}_{r_A} \wedge \text{valid}(x)\}$

$A \rightarrow V \quad :$

$V \looparrowright C \quad :$

FOO Protocol: Assertions

Can see structure! Both x, r visible

$V \rightarrow A \quad : \quad \{v\}_{r_A}, V \; says \; \{\exists x, r : \{x\}_r = \{v\}_{r_A} \wedge \text{valid}(x)\}$

No structure available! Just some bitstring

$A \rightarrow V \quad :$

$V \looparrowright C \quad :$

FOO Protocol: Assertions

$V \rightarrow A \quad : \quad \{v\}_{r_A}, V \textit{ says } \{\exists x, r : \{x\}_r = \{v\}_{r_A} \wedge \text{valid}(x)\}$

$A \rightarrow V \quad : \quad A \textit{ says } \big[\text{elg}(V) \wedge \text{voted}(V, \{v\}_{r_A})$
$\wedge\, V \textit{ says } \{\exists x, r : \{x\}_r = \{v\}_{r_A} \wedge \text{valid}(x)\}\big]$

$V \looparrowright C \quad :$

FOO Protocol: Assertions

$$V \rightarrow A \quad : \quad \{v\}_{r_A},\, V \textit{ says } \{\exists x, r : \{x\}_r = \{v\}_{r_A} \wedge \text{valid}(x)\}$$

$$A \rightarrow V \quad : \quad A \textit{ says } \Big[\text{elg}(V) \wedge \text{voted}(V, \{v\}_{r_A}) \wedge V \textit{ says } \{\exists x, r : \{x\}_r = \{v\}_{r_A} \wedge \text{valid}(x)\}\Big]$$

$$V \looparrowright C \quad : \quad \{v\}_{r_C}, r_C, \exists X, y, s : \Big\{A \textit{ says } \Big[\text{elg}(X) \wedge \text{voted}(X, \{y\}_s) \wedge X \textit{ says } \{\exists x, r : \{x\}_r = \{y\}_s \wedge \text{valid}(x)\}\Big] \wedge y = v\Big\}$$

Dolev-Yao Model

- Term algebra. $t := m \mid (t_1, t_2) \mid \{t\}_k$
- Intruder I can block, replay, forge terms — but not break encryption. Essentially the network.
- Send/receive by an agent governed by derivability checks.

X: set of terms

$$\frac{}{X \vdash t}\ ax \quad (t \in X)$$

$$\frac{X \vdash (t_0, t_1)}{X \vdash t_i}\ split_i \quad (i = 0, 1)$$

$$\frac{X \vdash t_0 \quad X \vdash t_1}{X \vdash (t_0, t_1)}\ pair$$

$$\frac{X \vdash \{t\}_k \quad X \vdash inv(k)}{X \vdash t}\ dec$$

$$\frac{X \vdash t \quad X \vdash k}{X \vdash \{t\}_k}\ enc$$

Dolev-Yao derivation system

Dolev-Yao Model

- Consider a communicated proof that a term is the encryption of one of two constants. Also encoded as a term, needs complex primitives!
- Logical content of such terms not immediately evident from description.
- Use "zkp" primitive [BMU08]: more readable, but no logical inference.
- From $(v = 0 \vee v = 1)$ and $(v = 0 \vee v = 2)$, agent should be able to derive $v = 0$. Impossible with zkp terms.
- Our extension to the Dolev-Yao model addresses these problems.

BMU08: Backes, M.; Hritcu C.; Maffei, M. (2008), "Type-checking zero-knowledge", *Proceedings of ACM CCS '08*, 357-370.

Enter Assertions

- Can now send "assertions" — capture basic facts about terms and communications, and allow logical inference over such facts. [RSS14]

- Important addition: existential quantifier – hides witnesses for partial knowledge proofs.

$$\alpha := t_1 = t_2 \mid \alpha_1 \vee \alpha_2 \mid \alpha_1 \wedge \alpha_2 \mid \exists x\, \alpha(x) \mid m \textit{ says } \alpha \mid \ldots$$

RSS14: Ramanujam R.; Sundararajan, V.; Suresh, S. P. (2014), "Extending Dolev-Yao with Assertions", *Proceedings of ICISS'14*, 50–68.

Assertions: Actions

- Implicitly trusted; model guarantees only true assertions are communicated — via TTP or translation into ZKPs.
- Intruder is again the network: can block, replay. But cannot forge assertions in general — *A says* α, for example, can only be sent by agent with *A*'s secret key.

Assertions: Actions

- Agents can send and receive assertions (enabling conditions similar to those for terms).
- Can branch based on assertions: confirm and deny actions. Also enabled by derivability checks.
- Can add new assertions to state: insert action. Internal action, specified by protocol description.

FOO Voting Protocol

$$V \rightarrow A \quad : \quad \{v\}_{r_A}, V \textit{ says } \{\exists x, r : \{x\}_r = \{v\}_{r_A} \wedge \text{valid}(x)\}$$

$$A \rightarrow V \quad : \quad A \textit{ says } \big[\text{elg}(V) \wedge \text{voted}(V, \{v\}_{r_A}) \wedge V \textit{ says } \{\exists x, r : \{x\}_r = \{v\}_{r_A} \wedge \text{valid}(x)\}\big]$$

$$V \looparrowright C \quad : \quad \{v\}_{r_C}, r_C, \exists X, y, s : \Big\{A \textit{ says } \big[\text{elg}(X) \wedge \text{voted}(X, \{y\}_s) \wedge X \textit{ says } \{\exists x, r : \{x\}_r = \{y\}_s \wedge \text{valid}(x)\}\big] \wedge y = v\Big\}$$

FOO Voting Protocol

$$
\begin{aligned}
V \rightarrow A \quad &: \quad \{v\}_{r_A}, V \; says \; \{\exists x, r : \{x\}_r = \{v\}_{r_A} \wedge \text{valid}(x)\} \\
A \quad &: \quad deny \;\; \exists x : \text{voted}(V, x) \\
A \rightarrow V \quad &: \quad A \; says \; \big[\text{elg}(V) \wedge \text{voted}(V, \{v\}_{r_A}) \\
&\qquad\qquad \wedge V \; says \; \{\exists x, r : \{x\}_r = \{v\}_{r_A} \wedge \text{valid}(x)\}\big] \\
V \rightsquigarrow C \quad &: \quad \{v\}_{r_C}, r_C, \\
&\quad \exists X, y, s : \Big\{A \; says \; \big[\text{elg}(X) \wedge \text{voted}(X, \{y\}_s) \\
&\qquad\qquad\qquad \wedge X \; says \; \{\exists x, r : \{x\}_r = \{y\}_s \\
&\qquad\qquad\qquad\qquad \wedge \text{valid}(x)\}\big] \\
&\qquad\qquad \wedge y = v\Big\}
\end{aligned}
$$

FOO Voting Protocol

$$
\begin{aligned}
V \rightarrow A \quad &: \quad \{v\}_{r_A}, V \textit{ says } \{\exists x, r : \{x\}_r = \{v\}_{r_A} \wedge \text{valid}(x)\} \\
A \quad &: \quad \textit{deny} \;\; \exists x : \text{voted}(V, x) \\
A \quad &: \quad \textit{insert} \;\; \text{voted}(V, \{v\}_{r_A}) \\
A \rightarrow V \quad &: \quad A \textit{ says } \big[\text{elg}(V) \wedge \text{voted}(V, \{v\}_{r_A}) \\
&\qquad \wedge V \textit{ says } \{\exists x, r : \{x\}_r = \{v\}_{r_A} \wedge \text{valid}(x)\}\big] \\
V \looparrowright C \quad &: \quad \{v\}_{r_C}, r_C, \\
&\quad \exists X, y, s : \Big\{A \textit{ says } \big[\text{elg}(X) \wedge \text{voted}(X, \{y\}_s) \\
&\qquad\qquad \wedge X \textit{ says } \{\exists x, r : \{x\}_r = \{y\}_s \\
&\qquad\qquad\qquad \wedge \text{valid}(x)\}\big] \\
&\qquad \wedge y = v\Big\}
\end{aligned}
$$

X: set of terms

Φ: set of assertions

$$\frac{X, \Phi \vdash \alpha(t)}{X, \Phi \vdash \exists x : \alpha(x)} \exists i$$

y does not appear in X, Φ or β

$$\frac{X, \Phi \vdash \exists x : \alpha(x) \quad X, \Phi \cup \{\alpha(y)\} \vdash \beta}{X, \Phi \vdash \beta} \exists e$$

$$\frac{X, \Phi \vdash \alpha \quad X \vdash_{dy} sk(A)}{X, \Phi \vdash A \; says \; \alpha} says_A$$

$$\frac{X, \Phi \vdash m = n}{X, \Phi \vdash \alpha} \perp [m, n \in \mathcal{B}, m \neq n]$$

Assertion derivation system: Key Rules

Anonymity: Setup

- Want to analyse FOO for anonymity.
- Runs need to satisfy following prerequisites.
  - At least two voters V_0 and V_1; at least two candidates *0* and *1*.
  - All voter-admin messages precede voter-collector ones.
  - Most powerful intruder — *I* controls admin *A* and collector C.

Anonymity: (Almost) Definition

We say that a protocol Pr satisfies anonymity if

for every run with a $(0, 0)$ and a $(1, 1)$ session,

there is a run with a $(1, 0)$ and a $(0, 1)$ session

such that the two runs are intruder-indistinguishable.

(i, j) session: V_i votes for j

Intruder-Indistinguishability

- Want *I* to not be able to distinguish between runs with different votes.
- Two runs are *intruder-indistinguishable* as long as *I* draws exactly the same conclusions, i.e., derives the same terms and "same" assertions, in both runs.

Intruder-Indistinguishability

ρ, ρ': two runs of a protocol.
u_i, v_i : terms communicated in i^{th} action in ρ and ρ' respectively.
(X,Φ), (X', Φ'): respective states of I at the end of the runs.

We say that ρ and ρ' are I-indistinguishable (denoted $\rho \sim_I \rho'$)

if for all

assertions $\alpha(\vec{x})$ and all sequences $\vec{u}$ and $\vec{v}$ of matching actions:

$$X,\Phi \vdash \alpha(\vec{u}) \quad \text{iff} \quad X',\Phi' \vdash \alpha(\vec{v})$$

Anonymity: Analysis for FOO

- $V \rightarrow A$: voter id is public, vote encrypted. V *says* assertion quantifies out value of vote.
- $V \rightarrow$ C: vote revealed, but sent anonymously. Existential assertion hides voter's id.
- Intuitively, no way for the intruder to link the voter's id to their vote (no ∃e possible). FOO satisfies anonymity!

Conclusions & Future Work

- Presented a new framework that sends assertions along with terms. Analyzed FOO protocol for anonymity.
- Passive intruder problem (checking $X, \Phi \vdash \alpha$): coNP-complete without quantifiers. Need to pin down complexity with quantifiers.
- Formalize other properties, integrate into tools for automation.
- Translation between terms-only and assertions-based protocols.

Thank You!