A Theory of Assertions for Dolev-Yao Models

R Ramanujam
IMSc, Chennai

Vaishnavi Sundararajan
CMI, Chennai

S P Suresh
CMI, Chennai

Formal Methods Update Meeting, Goa
20th July 2018

Introduction

- Security protocol: a pattern of communications to achieve a security goal in an insecure environment.
- Each communication is of the form $A \rightarrow B$: m.
- A and B are agents participating in the protocol, and m is some message.
- Malicious intruder can play havoc when many messages are being communicated, by mixing-and-matching (even without breaking cryptography).
- Need formal analysis of protocols to guarantee security goals!

Logical Flaws: Example

$A \to B : \{m\}_{pk(B)}$

$B \to A : \{m\}_{pk(A)}$

$A \to \quad : \{m\}_{pk(B)}$

$I \to B : \{m\}_{pk(B)}$

$B \to I : \{m\}_{pk(I)}$

$\to A : \{m\}_{pk(A)}$

Dolev-Yao Model

- Framework for analysis of security protocols.
- Messages are abstract terms rather than bit strings.
- Encryption, hashing etc. abstract functions on terms.
- Cryptography assumed to be perfect, no cryptanalysis!
- Formalize properties, verify.

Dolev-Yao Model: Intruder

Intruder *I* cannot break encryption, but can

- see any message
- block any message
- redirect any message
- generate messages — according to set rules!
- send messages in someone else's name
- initiate new communication according to the protocol

Dolev-Yao Model: Actions

- Two types of actions, send and receive.
- Each communication $A \longrightarrow B$ separated out into a send action ($+A$) and a 'corresponding' receive action ($-B$).
- Every sent term assumed to be received by I.
- Each received term assumed to come from I.
- Ties in well with intuition of I being the network!

Dolev-Yao model: Term syntax

$$t = m \mid \mathrm{pk}(\mathrm{k}) \mid \mathrm{pair}(t_0, t_1) \mid \mathrm{senc}(t, t') \mid \mathrm{aenc}(t, r, k)$$

- Term algebra as in picture.
- Derivation rules of the following form.

$$\frac{X \vdash t \quad X \vdash u}{X \vdash \mathrm{senc}(t, u)}\, senc \qquad \frac{X \vdash \mathrm{senc}(t, u) \quad X \vdash u}{X \vdash t}\, sdec$$

More about Dolev-Yao

- Dolev-Yao treats all messages as "terms".
- What if protocol involves certificates? For authorization, delegation etc.
- Encoded as terms in Dolev-Yao — bit commitment, protocol-specific tagging etc.
- Not always concise/readable!

ZKP Terms [BHM08]

- Extend the Dolev-Yao model with "zero-knowledge proof terms".
- Zero-knowledge proof term: $ZK_{p,q}(P_1,\ldots,P_p\,;\,Q_1,\ldots,Q_q\,;\,F)$.
- *P*s: private; *Q*s: public; *F* defines link between *P*s and *Q*s.
- Presents the certificate in a more readable format than encoding into terms.

$$A \to B : ZK_{2,3}\big(m, k\,;\,\{m\}_k, a, b\,;\,\beta_1 = enc(\alpha_1, \alpha_2) \wedge (\alpha_1 = \beta_2 \vee \alpha_1 = \beta_3)\big)$$

BHM08: Backes, M.; Hritcu, C.; Matteo, M. (2008) "Type-checking zero-knowledge." In *Proc.* CCS '08, 357–370.

ZKP Terms (Contd.)

- Sounds great! So why reinvent the wheel?
- Consider two certificates as follows: $\{m = a \text{ or } m = b\}$ and $\{m = a \text{ or } m = c\}$, with $b \neq c$.
- Ideally, should be able to derive $m = a$ from these two.
- One cannot do derivations on ZKP terms. Cannot infer $m = a$ from these certificates in this system.

Overall Idea

- Extend the Dolev-Yao model with a class of abstract objects called 'assertions' which capture certification.
- Protocol descriptions are readable. Assertions are distinct from terms, and clearly specify the statements of the certificates they model.
- Inference on assertions is possible, independent of underlying implementation.

Assertions

- Assertions have the following syntax.

  $$\alpha := t_1 = t_2 \mid P(t) \mid \alpha_1 \wedge \alpha_2 \mid \alpha_1 \vee \alpha_2 \mid \exists x.\ \alpha \mid A \textit{ says } \alpha$$

- The *says* connective allows agents to "sign" an assertion as coming from them.

- P is any application-specific predicate.

- Existential quantification lets agents hide witnesses.

- Earlier example now looks as follows:

  $$A \to B : \{m\}_k, \exists xy. [\{m\}_k = \{x\}_y \wedge (x = a \vee x = b)]$$

Existential Quantification

- When exactly can one existentially quantify out a term from an assertion?
- m from $m = t$? m from $\{m\}_k = t$?
- Quantification becomes complicated in the presence of encryption!

Abstractability

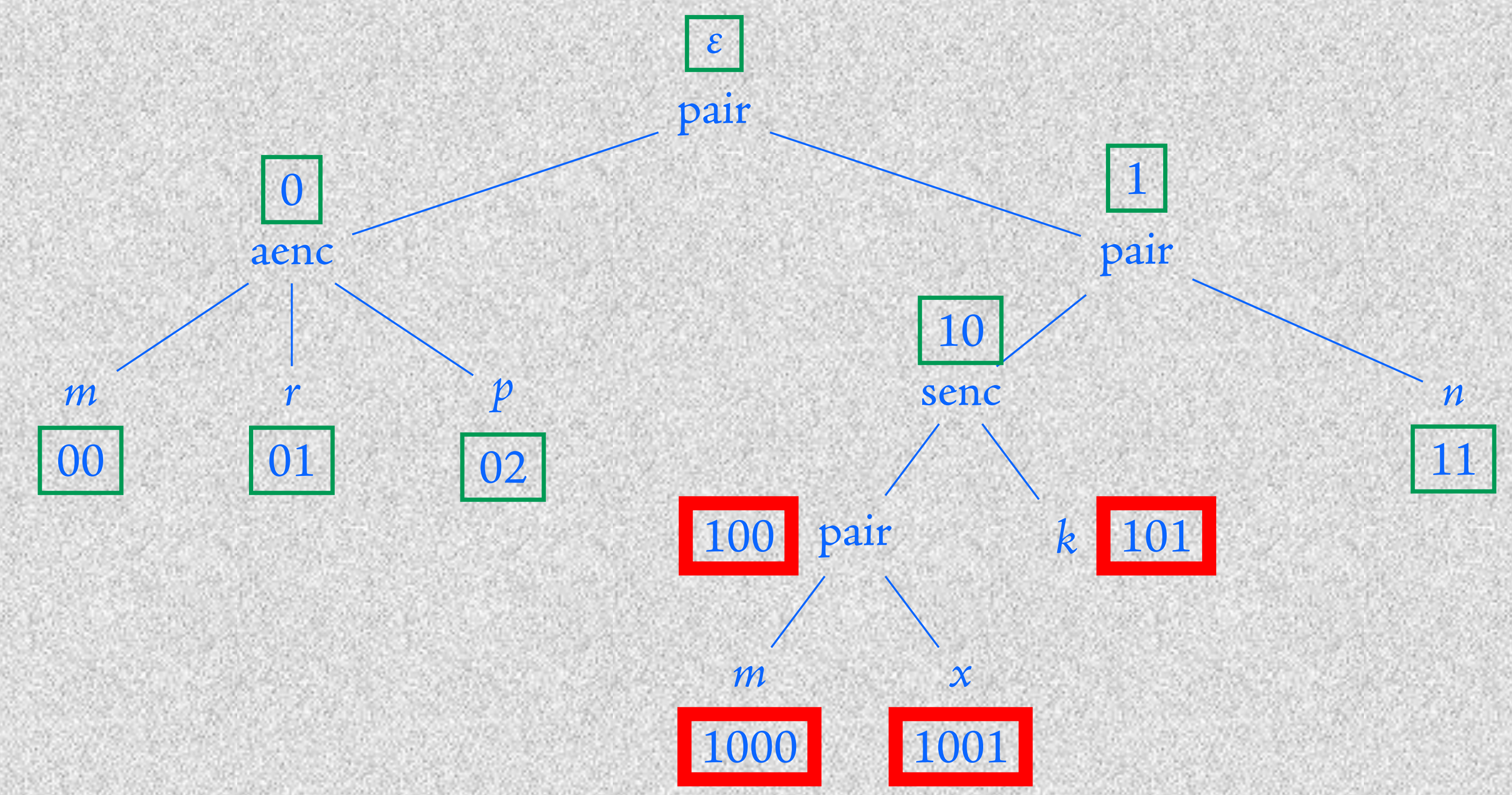
- Informally, a position p is 'abstractable' inside a term t if we can replace the subterm at p with something else and build the rest of t back up.

- We consider a notion of abstractability w.r.t. a set of terms S, if we can use (some of the) terms in S to build the relevant parts of t.

- abs(S, t): Set of abstractable positions of t w.r.t S.

Abstractability

- $X = \{m, r, p, \mathrm{pair}(\mathrm{senc}(\mathrm{pair}(m, x), k), n)\}$
- $t = \mathrm{pair}(\mathrm{aenc}(m, r, p), \mathrm{pair}(\mathrm{senc}(\mathrm{pair}(m, x), k), n))$
- $\mathrm{abs}(X, t) = \{\varepsilon, 0, 00, 01, 02, 1, 10, 11\}$

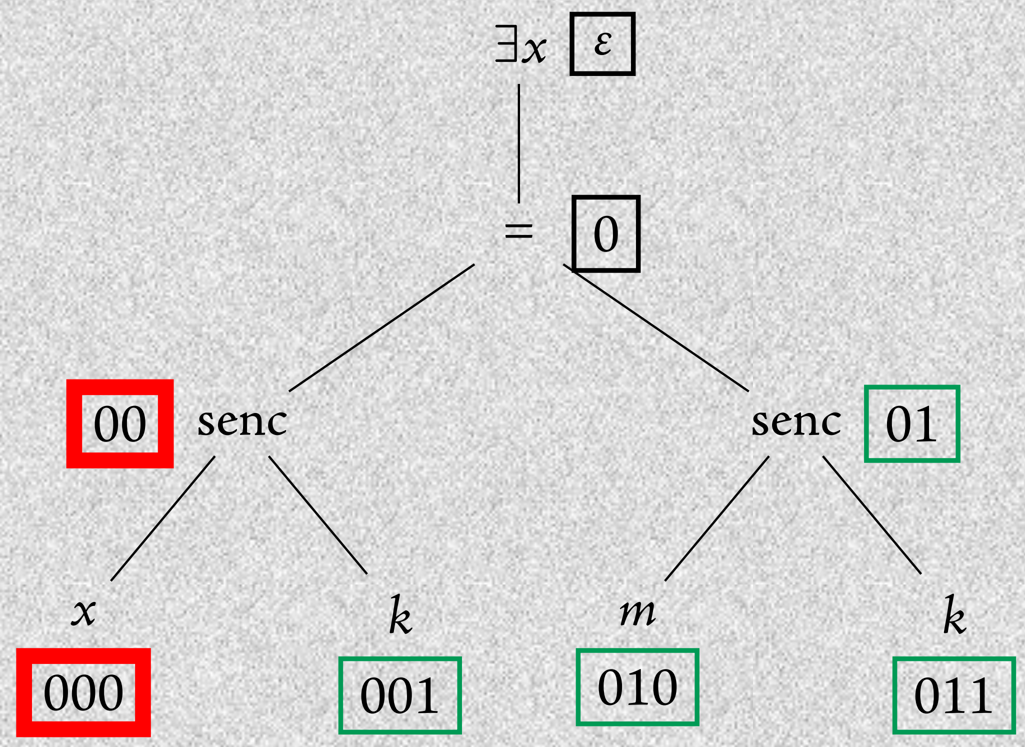


Abstractability: Assertions

- Can provide a similar definition of abstractability for assertions.
- A term-position p is abstractable from an assertion α if we can replace the term at p with something else and build the rest of α back up. Consider abs(S, α) as earlier.
- But what if assertion is already quantified — of the $\exists x.\alpha$ form? What positions can one remove then?

Abstractability: Assertions

- $X = \{\text{senc}(m, k), k\}$
- $\alpha = \exists x.[\text{senc}(x, k) = \text{senc}(m, k)]$
- $\text{abs}(X, \alpha) = \{001, 01, 010, 011\}$



Inference system for Assertions

- Sequents now of the form $S; A \vdash \alpha$.
- Simple equality rule: if t derivable from S, can state $t = t$.
- Some rules for manipulating equality make use of abstractability.

Inference system for Assertions

- Abstractability used by projection, substitution, existential introduction etc.
- Can go from $\alpha(t)$ to $\alpha(u)$ if all occurrences of t abstractable from α w.r.t. the set of terms S.
- Restricted contradiction rule: two terms t and u such that the structure of t and u can be determined (maybe using abstractability!) to be different, but $S; A \vdash t = u$.

$$\frac{}{S; A \cup \{\alpha\} \vdash \alpha}\ ax$$

$$\frac{S \vdash_{dy} t}{S; A \vdash t = t}\ eq$$

$$\frac{S; A \vdash \mathsf{f}(t_1, \ldots, t_r) = \mathsf{f}(u_1, \ldots, u_r)}{S; A \vdash t_i = u_i}\ proj_i \quad [t_i, u_i \text{ abstractable w.r.t. } S]$$

$$\frac{S; A \vdash t = u}{S; A \vdash \alpha}\ \bot \quad [S \Vdash t \bot u]$$

$$\frac{S; A \vdash \alpha[t]_P \quad S; A \vdash t = u}{S; A \vdash \alpha[u]_P}\ subst \quad [t \text{ abstractable w.r.t. } S,\ S \vdash_{dy} u]$$

Inference system for Assertions

- *A says* is essentially a signature with *A*'s private key, can be removed by an *unsay* rule.
- Rules for logical operators ∧, ∨ and ∃ are as in standard intuitionistic logic (caveat of abstractability for ∃i).

$\dfrac{S \vdash_{dy} k \quad S; A \vdash \alpha}{S; A \vdash \mathrm{pk}(k)\ \mathit{says}\ \alpha}\ \mathit{says}$	$\dfrac{S; A \vdash k\ \mathit{says}\ \alpha}{S; A \vdash \alpha}\ \mathit{unsay}$
$\dfrac{S; A \vdash \alpha_0 \quad S; A \vdash \alpha_1}{S; A \vdash \alpha_0 \wedge \alpha_1}\ \wedge\mathrm{i}$	$\dfrac{S; A \vdash \alpha_0 \wedge \alpha_1}{S; A \vdash \alpha_i}\ \wedge\mathrm{e}_i$
$\dfrac{S; A \vdash \alpha_i}{S; A \vdash \alpha_0 \vee \alpha_1}\ \vee\mathrm{i}$	$\dfrac{S; A \vdash \alpha \vee \beta \quad S; A \cup \{\alpha\} \vdash \delta \quad S; A \cup \{\beta\} \vdash \delta}{S; A \vdash \delta}\ \vee\mathrm{e}$
$\dfrac{S; A \vdash \alpha[t]_P}{S; A \vdash \exists x.\alpha}\ \exists\mathrm{i} \quad [t \text{ abstractable w.r.t. } S]$	$\dfrac{S; A \vdash \exists x.\alpha[x]_P \quad S \cup \{y\}; A \cup \{\alpha[y]_P\} \vdash \delta}{S; A \vdash \delta}\ \exists\mathrm{e} \quad [y \text{ is "fresh"}]$

Assertions: Actions

- As with terms, agents can send and receive assertions.
- Can now branch based on the derivability of assertions: confirm and deny actions.
- Can add new instances of predicates: insert action. Internal action, specified by protocol description.

Runtime Model

- An *A*-action is a send, receive, confirm or deny by *A*.
- Actions specified with as much pattern as possible for terms, with variables standing for unknowns.
- An *A*-role is a sequence of *A*-actions.

Runtime Model (Contd.)

- Each agent accumulates terms and assertions generated and received, in a knowledge state $(X; \Phi)$.
- Represent by $(X_A; \Phi_A)$ the knowledge state of agent A.
- Represent by $(X_I; \Phi_I)$ the knowledge state of the intruder I.
- Knowledge states used to enable actions, and possibly updated after performing actions.

Enabling & Updates

Action	Enabling conditions	Updates
A sends t, α with new nonces $\vec{m}$	$X_A \cup \{\vec{m}\} \vdash_{dy} t$ $X_A; \Phi_A \vdash \alpha$	$X'_A = X_A \cup \{\vec{m}\}$ $X'_I = X_I \cup \{t\}$ $\Phi'_I = \Phi_I \cup \{\alpha\}$
A receives t, α	$X_I \vdash_{dy} t$ $X_I; \Phi_I \vdash \alpha$	$X'_A = X_A \cup \{t\}$ $\Phi'_A = \Phi_A \cup \{\alpha\}$
$A : \mathit{confirm}\ \alpha$	$X_A; \Phi_A \vdash \alpha$	No update
$A : \mathit{deny}\ \alpha$	$X_A; \Phi_A \nvdash \alpha$	No update

Runtime Model (Contd.)

- A protocol is just a set of roles.
- Can consider various instantiations of roles — sessions.
- A run is an admissible (according to enabling conditions!) interleaving of such sessions.
- One can think of a transition system with states that keep track of agents' knowledge and all the sessions in progress, where enabled actions induce transitions.

Example: FOO e-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

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

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

$V \looparrowright C \quad : \quad \{\{v\}_r\}_{sg(A)}$

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

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

$V \looparrowright C \quad : \quad r$

FOO Protocol: What we want

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

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

$V \nrightarrow C$: $\{v\}_{k'}$, *"Some eligible agent was authorized by A to vote with a valid vote, this term is a re-encryption 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

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

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

$$V \looparrowright C \quad :$$

FOO Protocol: Assertions

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

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

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

FOO Protocol: Assertions

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

$$A \quad : \quad \textit{deny}\ \ \exists x : \text{voted}(V, x)$$

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

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

FOO Protocol: Assertions

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

$$A \quad : \quad \mathit{deny}\ \ \exists x : \mathrm{voted}(V, x)$$

$$\mathit{insert}\ \ \mathrm{voted}(V, \{v\}_{r_A})$$

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

$$V \looparrowright C \quad : \quad \{v\}_{r_C}, r_C,$$

$$\exists X, y, s : \Big\{A\ \mathit{says}\ \big[\mathrm{elg}(X) \wedge \mathrm{voted}(X, \{y\}_s) \wedge X\ \mathit{says}\ \{\exists x, r : \{x\}_r = \{y\}_s \wedge \mathrm{valid}(x)\}\big] \wedge y = v\Big\}$$

Anonymity: Setup

- Want to analyze 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. FOO satisfies anonymity!

Verification

- Derivability problem: Given a finite set of terms X, a finite set of assertions Φ, and an assertion α, is it the case whether $X; \Phi \vdash \alpha$?

- Insecurity problem: Given a protocol Pr and a designated secret assertion α, is there a run of Pr at the end of which $X_I, \Phi_I \vdash \alpha$?

Conclusions & Future Work

* Presented an abstract model for security protocols involving certification. Analyzed FOO protocol for anonymity.
* Implementation and tool support.
* Translation between terms-only and assertions-based protocols.

References

- Existential assertions for voting protocols
  R Ramanujam, Vaishnavi Sundararajan and S P Suresh
  Proc. FC 2017 Workshops (Voting '17), Springer LNCS vol. 10323, 337–352.
- The complexity of disjunction in intuitionistic logic
  R Ramanujam, Vaishnavi Sundararajan and S P Suresh
  Proc. LFCS 2016, Springer LNCS vol. 9537, 349–363.
- Extending Dolev-Yao with assertions
  R Ramanujam, Vaishnavi Sundararajan and S P Suresh
  Proc. ICISS 2014, Springer LNCS vol. 8880, 50–68.

Thank you!