Multiagent System-based Verification of Security and Privacy

Ioana Boureanu

Imperial College London
Department of Computing

September 2015

Outline

Outline

Model Checking MAS

1. Model Checking in Theory
2. Model Checking MAS in Practice
3. Logic-based Languages
4. MAS-based Models

Model Checking In Theory

- *Model checking* [Clarke et al., 1999] is a verification technique
- $M \models \varphi$, given a model M for a system and a specification φ encoding one of the system's properties

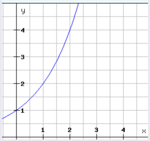
Our Example of Models & Specifications

- M — a **formal** semantics for multiagent systems
- φ — **knowledge**, **joint abilities** beliefs, intentions, . . ., to express fault-tolerance, diagnosability, **security ...**

Model Checking in Practice

Real World Verification

An explicit modelling → state-space exponential in the size of the input



An optimised, much simplified model for onion routing has 3.03439e+58 reachable states!

We need efficient methods and tools!

Model Checking in Practice

Pbs & Solutions

- state explosion pb: explicit encodings of state/action in M
  - one solution: efficient/**symbolic** encodings, e.g., via binary decision diagrams (BDDs)

(More) Pbs & Solutions

- MC algorithms over BDD-encoded specifications & tools
  - solution: MAS symbolic model-checking techniques [Lomuscio and Raimondi, 2006]

(More) Pbs & Solutions

- there's always a need for optimisations
  - solutions: cut-offs, abstractions [Lomuscio and Kouvaros, 2015], etc.
    and/in a robust tool MCMAS [Lomuscio et al., 2015]

Model Checking MAS in Practice

MCMAS [Lomuscio et al., 2015]

- Support for epistemic specifications, ATL (uniformity and fairness), CTL, deontic modalities
- Dedicated modelling language (`ISPL`)
- BDD-based (via CUDD). Sequential and parallel MC
- Eclipse GUI
- Support for witnesses, counterexamples, etc
- Open source
- Used for robotic swarms, web-services, **security**...

Logic-based Languages

A Stop At Epistemic Specifications

- $S5_n$
- $\varphi = p \mid \neg\varphi \mid \varphi \wedge \varphi \mid K_i\varphi$
- readings:
  - $K_i\varphi$ – “agent *i* knows that φ”

MAS-based Models

Interpreted Systems

- Multiagent-based models
  [Lodaya et al., 1995, Fagin et al., 1995]
  - $A = \{1, \ldots, n\}$ agents and ***E**nvironment* agent;
  - $\forall i \in A \cup \mathbf{E}$: L_i – possible *local states*, Act_i – *local actions*, $P_i : L_i \to 2^{Act_i}$ – *protocol function* (actions enabled at l_i);
  - $t_i(l_i, a_1, \ldots, a_n, a_{\mathbf{E}}) = l'_i$ – *local evolution function*;
  - G – *global states*, $\overline{P}$ – *joint protocol*, Act – *joint actions*, T *global evolution function* — by composition;
  - $IS = \left\langle G, \overline{P}, T, I, V \right\rangle$ – *interpreted system*,
    where $I \subset G$ – *initial global states* and
    $V : G \to 2^{AP}$ – *valuation* function;

MAS-based Models

MAS Induced-Models

The *induced model of IS* is a tuple $\mathcal{M}_{\mathcal{IS}} = (S, T, \{\sim_i\}_{i \in \{1 \dots n\}}, V)$ where:

- $S \subseteq L_0 \times \cdots \times L_n$ is the set of *global states reachable from I via T*
- T encodes the temporal evolution;
- $\{\sim_i\}_{i \in Ag \setminus \mathbf{E}} \subseteq S \times S$ is a set of equivalence relations encoding epistemic accessibility

MAS-based Models

State Indistinguishability

- $l \in L_i$ and $l' \in L_i$ are *i-indistinguishable*, $l \approx_i l'$ if -in general- $\approx_i \subseteq L_i \times L_i$ is an equivalence relation over L_i
  - standard:
    $\approx_i$ is the equality relation: $l_i(g) \approx_i l_i(g')$ iff $l_i(g) = l_i(g')\}$
  - non-standard:
    $\approx_i$ is a bespoke equiv. relation

    e.g., $\mathbf{l} \equiv \{\mathbf{m_1}\}_{\mathbf{k_1}}$ and $\mathbf{l'} \equiv \{\mathbf{m_2}\}_{\mathbf{k_2}}$

    (assuming l containing just the encryption of a term with a key and l' containing yet just the encryption of another term with another key)
    $\Rightarrow \quad \mathbf{l} \approx_\mathbf{i} \mathbf{l'}$
- $s, s' \in S$ are *i-indistinguishable*, $s \sim_i s'$, if $l_i(s) \approx_i l_i(s)$

MAS-based Models

Satisfaction of Formulae on MAS Models

- CTL and ATL fragments as usual
- $(M, s) \models K_i\phi$ iff $\forall s' \in S$ if $s \sim_i s'$ then $(M, s') \models \phi$

Outline

Outline

Joint work

Based on:

- previous joint work at Imperial College London
  - I. B., M. Cohen, A. Lomuscio, “Automatic Verification of Temporal-Epistemic Properties of Cryptographic Protocols”, Journal of Applied Non-Classical Logics, 2009
  - I. B., A. Lomuscio, M. Cohen,“Model Checking Detectability of Attacks in Multiagent Systems”, AAMAS 2010
  - I. B,. A. Jones, A. Lomuscio, “Automatic Verification of Temporal-Epistemic Logic under Convergent Equational Theories”, AAMAS 2012
- I. B., “Model checking security protocols: a multi-agent system approach”, PhD Thesis, Imperial College London, 2011
- ongoing joint work with A. Lomuscio and the VAS group at Imperial College London
- H2020 “Logic-based Verification of Privacy-Preservation in Europe’s 2020 ICT”

Motivation...

- “Protocols ... are prone to extremely subtle errors that are unlikely to be detected in normal operation.”

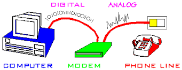
  (Needham and Schroeder, 1978)

- VeriSign spent $> \$10^8$ in 2009–2010 to upgrade the *.com* `DNS` servers

- more interconnected devices, more conversative apps, more security threats

Motivation...

- "Protocols ... are prone to extremely subtle errors that are unlikely to be detected in normal operation."
  (Needham and Schroeder, 1978)
- VeriSign spent $> \$10^8$ in 2009–2010 to upgrade the *.com* DNS servers
- more interconnected devices, more conversative apps, more security threats

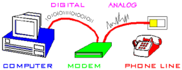


Motivation...

- "Protocols ... are prone to extremely subtle errors that are unlikely to be detected in normal operation."
  (Needham and Schroeder, 1978)

- VeriSign spent $> \$10^8$ in 2009–2010 to upgrade the *.com* DNS servers

- more interconnected devices, more conversative apps, more security threats



Symbolic Security Attacks

- Example: the Woo-Lam authentication protocol:

$$
\begin{array}{llll}
1. & A \rightarrow B & : & A \\
2. & B \rightarrow A & : & N_b \\
3. & A \rightarrow B & : & \{A, B, N_b\}_{K_{AS}} \\
4. & B \rightarrow S & : & \{A, B, \{A, B, N_b\}_{K_{AS}}\}_{K_{BS}} \\
5. & S \rightarrow B & : & \{A, B, N_b\}_{K_{BS}}
\end{array}
$$

Symbolic Security Attacks

- Example: the Woo-Lam authentication protocol:

$$
\begin{array}{llll}
1. & A \rightarrow B & : & A \\
2. & B \rightarrow A & : & N_b \\
3. & A \rightarrow B & : & \{A, B, N_b\}_{K_{AS}} \\
4. & B \rightarrow S & : & \{A, B, \{A, B, N_b\}_{K_{AS}}\}_{K_{BS}} \\
5. & S \rightarrow B & : & \{A, B, N_b\}_{K_{BS}}
\end{array}
$$

Example: an attack against the Woo-Lam protocol:

$$
\begin{array}{llll}
1'. & I_A \rightarrow B & : & A \\
2'. & B \rightarrow I_A & : & N_b \\
3'. & I_A \rightarrow B & : & N_b \\
4'. & B \rightarrow I_S & : & \{A, B, N_b\}_{K_{BS}} \\
5'. & I_S \rightarrow B & : & \{A, B, N_b\}_{K_{BS}}
\end{array}
$$

Security Goals

‘Well-established’ Requirements

- flavours of: secrecy, authentication, key-agreement, etc.

Application-Level Privacy Requirements

privacy of application-data

- vote-privacy, receipt-freeness, coercion-resistance

Data-transport privacy

- origin anonymity, destination anonymity, unlinkability within routing

Fault-Diagnosability Requirements

- attack (un)detectability

Symbolic Verification of Cryptographic Protocols

SYMBOLIC = cryptographic messages are algebraic terms;
cryptography is perfect/un-tamperable
NO ppt. capabilities on protocol parties

- logic-based formalisms (BAN logics, Horn clauses);
  inductive methods;
  rewriting-based formalisms process-algebra formalisms (CSP, spi-calculus, pi-calculus);

  ...
- agent-based formalism
  - sound knowledge of participants;
  - natural expression of state-based properties (anonymity, non-repudiation etc.)

Challenges in (MAS) Security Specification/Verification

- even secrecy in the unbounded setting is undecidable; need to design good/sound bounded security formalisms [Tiplea et al., 2009]
- mechanise cryptographic operations in MAS formalisms, i.e., no inherent intermediate, algebra/arithmetics-based language
- encapsulate standard threat models (e.g., at least Dolev-Yao [D.Dolev and A.Yao, 1983]) in MAS formalisms
- get sound cryptography-driven indistinguishability relations & cryptography-aware epistemic modalities
- do any/all of the above in a systematic/automatable way

Challenges in (MAS) Security Specification/Verification

- even secrecy in the unbounded setting is undecidable; need to design good/sound bounded security formalisms [Tiplea et al., 2009]
- mechanise cryptographic operations in MAS formalisms, i.e., no inherent intermediate, algebra/arithmetics-based language
- encapsulate standard threat models (e.g., at least Dolev-Yao [D.Dolev and A.Yao, 1983]) in MAS formalisms
- get sound cryptography-driven indistinguishability relations & cryptography-aware epistemic modalities
- do any/all of the above in a systematic/automatable way

Challenges in (MAS) Security Specification/Verification

- even secrecy in the unbounded setting is undecidable; need to design good/sound bounded security formalisms [Tiplea et al., 2009]
- mechanise cryptographic operations in MAS formalisms, i.e., no inherent intermediate, algebra/arithmetics-based language
- encapsulate standard threat models (e.g., at least Dolev-Yao [D.Dolev and A.Yao, 1983]) in MAS formalisms
- get sound cryptography-driven indistinguishability relations & cryptography-aware epistemic modalities
- do any/all of the above in a systematic/automatable way

Challenges in (MAS) Security Specification/Verification

- even secrecy in the unbounded setting is undecidable; need to design good/sound bounded security formalisms [Tiplea et al., 2009]
- mechanise cryptographic operations in MAS formalisms, i.e., no inherent intermediate, algebra/arithmetics-based language
- encapsulate standard threat models (e.g., at least Dolev-Yao [D.Dolev and A.Yao, 1983]) in MAS formalisms
- get sound cryptography-driven indistinguishability relations & cryptography-aware epistemic modalities
- do any/all of the above in a systematic/automatable way

Challenges in (MAS) Security Specification/Verification

- even secrecy in the unbounded setting is undecidable; need to design good/sound bounded security formalisms [Tiplea et al., 2009]
- mechanise cryptographic operations in MAS formalisms, i.e., no inherent intermediate, algebra/arithmetics-based language
- encapsulate standard threat models (e.g., at least Dolev-Yao [D.Dolev and A.Yao, 1983]) in MAS formalisms
- get sound cryptography-driven indistinguishability relations & cryptography-aware epistemic modalities
- do any/all of the above in a systematic/automatable way

Protocol Executions as MAS Models

Security Protocols

the Needham-Schroeder Public Key (NSPK) protocol

an actual A is *alice*: e.g., a customer

an actual B is *bob*, e.g., a bank-server

1. $A \rightarrow B : \{A, N_A\}_{pub(B)}$

2. $B \rightarrow A : \{N_A, N_B\}_{pub(A)}$

3. $A \rightarrow B : \{N_B\}_{pub(B)}$

- *alice* could have, in the same time, a session from her mobile device and another session from her PC
- there could be other servers, but *bob*, that *alice* could connect to
- if this was, e.g., a contract-signing protocol, *alice* could have two, simultaneous running sessions: in one she could be auctioning (A-role) and in the other she could be a buyer (B-role)

Protocol Executions as (Simple) MAS Models (I)

MAS Mapping

- each role instance $((A, alice)^1, (A, alice)^2 or (A, bob)^3$ etc.) $\rightarrow$ an agent (of the IS)
- a (Dolev-Yao) intruder $\rightarrow$ the Environment agent, modelled purposedly

Protocol Executions as (Simple) MAS Models (II)

— some details :

- describe a (honest) instantiated role:
  - **views** – ordered map ⟨ var,value ⟩ ⇒ agents' local states with typed, un-deciphered values, ⊥, à la [Rogaway 2001]
    $(A : alice, B : bob, k_A : pvk_{alice}, k_B : pbk_{bob}, n_A : r_1, \mathbf{n_b} : \bot)$ or,
- describe a DY insider ⇒ local state of the Environment:
  - knowledge-set – ordered multimap ⟨ term,value ⟩
    $X = [\{A, na\}_{k_B} : \{alice, r_1\}_{pbk_{\mathbf{bob}}},$
    $\{A, na\}_{k_B} : \{alice_2, r_2\}_{pbk_{\mathbf{greg}}}, A : alice, A : alice_2, B : bob]$
  - history of actions
    $H = [ag_A.send\,\{alice, r_1\}_{pbk_{\mathbf{bob}}},$
    $ag'_A.send\,\{alice_2, r_2\}_{pbk_{\mathbf{greg}}}, \ldots]$

Protocol Executions as (Simple) MAS Models (III)

protocol role instantiated under $\rho \rightarrow$

- **evolution function**
  - simple agents' local state update
    e.g, "matching receive" of message $M = \{x, f(x), y\}_{K_{alice}}$ for the symbolic $\{n_a, n, n_b\}_{K_a}$ & agent i has previously set n_a:
    — $out\_match(view_i, M) = true \text{ iff } x = ag.n_a$

    — $in\_match(M, i) =$
    $true$, iff consitstency checks inside M hold; e.g., $n == f(n_a)$

    — $set(view, n_b)$: $n_b := y$ if $in\_match(\ldots) = true$ and $out\_match(\ldots) = true$
  - Env.'s local state update (e.g., DY deductions of the insider):
    $\tilde{a}_E = interceptM$, $\tilde{a}_{ag_A} = sendM$,
    $t_E((X, H), \tilde{a}) = (X \cup M \cup \{t | \{X \cup M\} \vdash t\}, H \cup ag_A.send\ M)$.

Protocol Executions as (Simple) MAS Models (III)

protocol role instantiated under $\rho \rightarrow$

- **evolution function**
  - simple agents' local state update
    e.g, "matching receive" of message $M = \{x, f(x), y\}_{K_{alice}}$ for the symbolic $\{n_a, n, n_b\}_{K_a}$ & agent i has previously set n_a:
    — $out\_match(view_i, M) = true$ *iff* $x = ag.n_a$

    — $in\_match(M, i) =$
    $true$, *iff* consitstency checks inside M hold; e.g., $n == f(n_a)$

    — $set(view, n_b)$: $n_b := y$ *if* $in\_match(\ldots) = true$ *and* $out\_match(\ldots) = true$
  - Env.'s local state update (e.g., DY deductions of the insider):
    $\tilde{a}_E = interceptM$, $\tilde{a}_{ag_A} = sendM$,
    $t_E((X, H), \tilde{a}) = (X \cup M \cup \{t | \{X \cup M\} \vdash t\}, H \cup ag_A.send\ M)$.

Security goals to CTLK specification (I)

- atomic goal `agree A : B :` $\overline{\mathtt{VAR}}$

$$\theta(\mathtt{agree\ A\ :\ B\ :\ }\overline{\mathtt{VAR}}) = \bigwedge_{i \in A} AG(end(i) \rightarrow \bigvee_{j \in B} agree(i, j, \overline{\mathtt{VAR}}))$$

i – agents ag_A mappings of *A–role* instance
j – agents ag_B mappings of *B–role* instance

$$agree(i, j, \overline{\mathtt{VAR}}) := \bigwedge_{Var \in \overline{\mathtt{VAR}}} (i.Var = j.Var)$$

- epistemic goal `Knows A :` γ

$$\theta(\mathtt{Knows\ A\ :\ }\gamma) = \bigwedge_{i \in A} AG(end(i) \rightarrow K_i\, \theta^{\mathbf{i}}(\gamma))$$

$\theta^{\mathbf{i}}(\gamma)$ –an appropriate translation of γ from the perspective of agent i:

$$\theta^{\mathbf{i}}(\mathtt{holds\ A\ :\ }\overline{\mathtt{VAR}}) = \bigvee_{j \in A} (i.PartnerA = j.Id \wedge agree(i, j, \overline{\mathtt{VAR}}))$$

Security goals to Specifications — One Example

- Doxastic authentication goal:

$$\texttt{Believes}\, B : \texttt{holds}\, A : \texttt{K}$$

- translation 1:

$$\bigwedge_{i \in B} AG(i.step = 3 \rightarrow K_i\, \theta^{\mathbf{i}}(\texttt{holds A : K}))$$

—$\theta^{\mathbf{i}}(\texttt{holds A : K}) :=$

$$\bigvee_{j \in A} (i.PartnerA = j.Id \wedge i.\texttt{K} = j.\texttt{K})$$

— $\theta^{\mathbf{i}}(\texttt{holds A : K})$ $\Rightarrow$

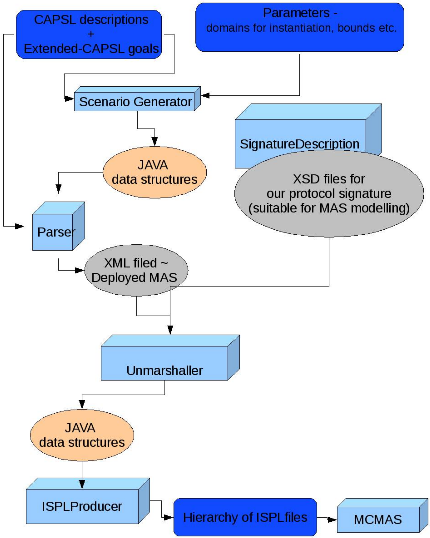
$$\bigwedge_{\mathbf{i} \in \mathbf{B}} \mathbf{AG}(\mathbf{i.step} = \mathbf{3} \rightarrow \mathbf{K_i} \bigvee_{\mathbf{j} \in \mathbf{A}} (\mathbf{i.PartnerA} = \mathbf{j.Id} \wedge \mathbf{i.}\texttt{K} = \mathbf{j.}\texttt{K}))$$

Security Protocols to MAS and CTLK

- translate different types of authentication, secrecy, key-exchange and their goals into CTLK formulas
- undetectability of attacks $\rightarrow$ new MAS formalism and hierarchy of CTLK formulas
- MAS formalisms proven correct w.r.t. trace properties, i.e., aligned with established security specification formalisms (MSR)
- done automatically from library of protocols in CAPSL to ISPL, into MCMAS

(Simple) MAS Modelling for Security

Security Protocols to MAS and CTLK – PD2IS



(Not So Simple) MAS Models for Security (I)

Intricate Cryptography, MAS and Epistemic

- cryptographic primitives can be complicated (e.g., blind signatures, trapdoor commitments, etc.)
- un-decipharable yet typed data requires attentive modelling (e.g., values in local states)
- local evolutions (e.g., checks to be made) become convoluted
- systematisation/automation possible per classes of primitives only
- need for sound epistemic modalities to be interpreted over these

(Not So Simple) MAS Models for Security (I)

Intricate Cryptography, MAS and Epistemic

- cryptographic primitives can be complicated (e.g., blind signatures, trapdoor commitments, etc.)
- un-decipharable yet typed data requires attentive modelling (e.g., values in local states)
- local evolutions (e.g., checks to be made) become convoluted
- systematisation/automation possible per classes of primitives only
- need for sound epistemic modalities to be interpreted over these

$$open(tdcommit(x,y,z),y) \rightarrow x$$
$$open(tdcommit(x,y,z),f(x,y,z,x')) \rightarrow x'$$
$$tdcommit(x',f(x,y,z,x'),z) \rightarrow tdcommit(x,y,z)$$
$$f(x',f(x,y,z,x'),z,x'') \rightarrow f(x,y,z,x'')$$

(Not So Simple) MAS Models for Security (I)

Intricate Cryptography, MAS and Epistemic

- cryptographic primitives can be complicated (e.g., blind signatures, trapdoor commitments, etc.)
- un-decipharable yet typed data requires attentive modelling (e.g., values in local states)
- local evolutions (e.g., checks to be made) become convoluted
- systematisation/automation possible per classes of primitives only
- need for sound epistemic modalities to be interpreted over these

$$open(tdcommit(x, y, z), y) \to x$$
$$open(tdcommit(x, y, z), f(x, y, z, x')) \to x'$$
$$tdcommit(x', f(x, y, z, x'), z) \to tdcommit(x, y, z)$$
$$f(x', f(x, y, z, x'), z, x'') \to f(x, y, z, x'')$$

(Not So Simple) MAS Models for Security (I)

Intricate Cryptography, MAS and Epistemic

- cryptographic primitives can be complicated (e.g., blind signatures, trapdoor commitments, etc.)
- un-decipharable yet typed data requires attentive modelling (e.g., values in local states)
- local evolutions (e.g., checks to be made) become convoluted
- systematisation/automation possible per classes of primitives only
- need for sound epistemic modalities to be interpreted over these

$$open(tdcommit(x,y,z),y) \rightarrow x$$
$$open(tdcommit(x,y,z),f(x,y,z,x')) \rightarrow x'$$
$$tdcommit(x',f(x,y,z,x'),z) \rightarrow tdcommit(x,y,z)$$
$$f(x',f(x,y,z,x'),z,x'') \rightarrow f(x,y,z,x'')$$

(Not So Simple) MAS Models for Security (I)

Intricate Cryptography, MAS and Epistemic

- cryptographic primitives can be complicated (e.g., blind signatures, trapdoor commitments, etc.)
- un-decipharable yet typed data requires attentive modelling (e.g., values in local states)
- local evolutions (e.g., checks to be made) become convoluted
- systematisation/automation possible per classes of primitives only
- need for sound epistemic modalities to be interpreted over these

$$open(tdcommit(x, y, z), y) \rightarrow x$$
$$open(tdcommit(x, y, z), f(x, y, z, x')) \rightarrow x'$$
$$tdcommit(x', f(x, y, z, x'), z) \rightarrow tdcommit(x, y, z)$$
$$f(x', f(x, y, z, x'), z, x'') \rightarrow f(x, y, z, x'')$$

(Not So Simple) MAS Models for Security (II)

Intricate Cryptography, MAS and Epistemics

- for **cryptographic primitives expressed as subterm convergent rewriting**, we give a MAS modelling
- we augment agents with logical predicates to encode the cryptographic data they hold
- we soundly approximate cryptographic indistinguishability/knowledge $\sim_i$ via indistinguishability/knowledge modulo these predicates
- we implement this in MCMAS and extend PD2IS to **automatically verify e-voting modelled as MAS, against CTLK formulae for vote-privacy, receipt-freeness**, etc.

Future Avenues for Security Apps as MAS

- soundness of such MAS methodologies w.r.t. state-based properties (e.g., privacy) remains to be proven
- many properties not captured by these models, e.g., data-origin, origin-privacy, etc.
- new MAS optimisation techniques (abstraction [Lomuscio and Michaliszyn, 2014], cut-off techniques and parametrised MC [Lomuscio and Kouvaros, 2014, 2015] can help improve these MAS-based security specification/verification methodologies
- newer applied logics (ATL, strategy logics [Cermak et al., 2013]) can be used to verify tighter requirements and more properties (e.g., privacy in e-auctioning protocols, shared resources in IoT, multi-party computations)

Future Avenues for Security Apps as MAS

Thank you!

Future Avenues for Security Apps as MAS