# Privacy Analysis of a Hidden Friendship Protocol

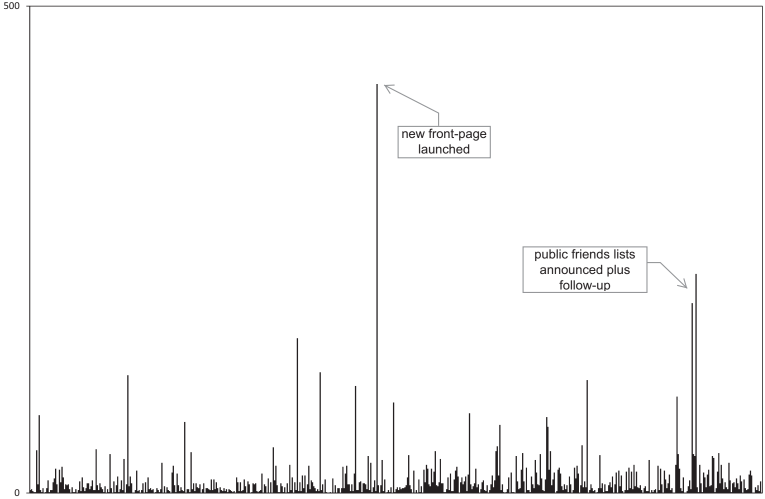
Florian Kammüller and Sören Preibusch

Middlesex University London & Microsoft Research Cambridge

DPM London, September 2013

# Hidden friendship relations in social networks

- Users of social networks are concerned about privacy



Increase of posts on German social network operators site following the announcements that friends lists would be publicly visible by default.

# Hidden friendship relations in social networks

- Idea: hidden friendship protocol to increase privacy [9]
- Problem: establishment and enforcement may enable privacy attacks

⇒ Apply solutions from protocol verification

⇒ Analysis of hidden friendship protocol shows attack risks and improvements

# Overview

1. The Hidden Friendship Protocol

2. MCMAS Modelchecking the Protocol

3. Summary and Discussion Points

# The FOAF Standard

- Friend Of A Friend (FOAF) standard [4]
  - Machine-readable (XML-like) format to publish friends lists

```
<?xml version="1.0" encoding="utf-8"?>
<rdf:RDF xmlns:rdf= ...
<foaf:Person rdf:ID="soeren">
<foaf:name>Soeren Preibusch</foaf:name>
...
<foaf:knows><foaf:Person>
<foaf:name>Alice Allington</foaf:name>
</foaf:Person>
</rdf:RDF>
```

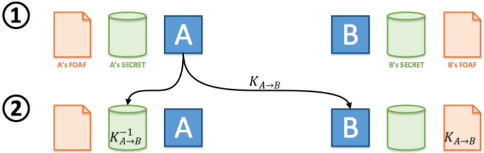
- Public and Hidden Friends in compact notation

$$\mathit{foaf}_A = (\mathit{name}_A, t, \{\mathit{name}_C, \mathit{name}_D, K_{B\to A}\})$$

- I.e., instead of hidden friend’s name, a friendship specific *Key*: $K_{B\to A}$

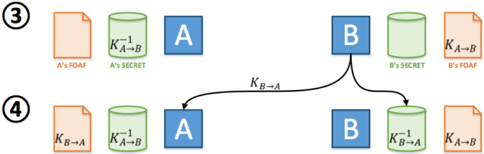
# Establishing a hidden friendship relation

- $A$ and $B$ want to share files privately
- They generate key pairs; keep private keys secret
- $B$ receives $K_{A\to B}$



$$foaf_B = (name_A, t_2, \{name_C, name_D, K_{A\to B}\})$$

- $A$ receives $K_{B\to A}$



$$foaf_A = (name_B, t_3, \{name_E, name_F, K_{B\to A}\})$$

# Using Hidden Friendship Relation

Part 2: *B* wants to retrieve data from *A* – his hidden friend

*I* sends along $foaf_B$ containing $name_B$ and $K_{A \to B}$.

*II* *A* receives *B*'s request and extracts $name_B$ from it.

*III* *A* then applies her secret key $K^{-1}_{A \to B}$ to verify $K_{A \to B}$. On success, *B* will be granted access to the file.

⇒ Nice, simple idea. But does it preserve privacy? Or security?

# Overview

1. The Hidden Friendship Protocol
2. MCMAS Modelchecking the Protocol
3. Summary and Discussion Points

# Model Checking

**ACM Turing Award Honors Founders of Automatic Verification Technology**

Researchers Created Model Checking Technique for Hardware and Software Designers

- Turing Award for Model Checking (2007)
- Fully automated technique for mathematical verification of state based systems
- Specify Model as a finite state transition system
- Natural specification of system properties with so-called “temporal logics”

AG send ⇒ (X receive) : send is always followed by receive

⇒ Various modalities possible: e.g., probabilites or epistemic logic, i.e. “beliefs”

# Modelchecking Friendship Protocol

- BAN Logic [1] for expressing and analysing security protocols
- Success story: Lowe Attack on NSPK with FDR Modelchecker [6]
- MCMAS [5]: Model Checking Multi Agent Systems
- MCMAS contains subset of CTL$^\star$ augmented with epistemic logic
- We can express for example:
  *If Bob has connected to Alice then the Intruder knows that Alice and Bob are hidden friends.*

  ```
  BconnectedA -> K(Intruder, AknowsB);
  ```

# MCMAS Model

- Modeling Agents Alice, Bob and Intruder, e.g.,

```
Agent Alice
Vars:
 initialpermission : { none };
 currentpermission: { accesstoB, accesstoI, none };
end Vars
Actions = { wait, openaccessB, openaccessI, sendfoaftoB };
Protocol:
 currentpermission = none : { wait };
...
Evolution:
 currentpermission = accesstoB if
 (Environment.foafcontainsKAB = true and Bob.Action = sendfoaftoA);
end Evolution
end Agent
```

- General assumption: Dolev-Yao model, i.e. all channels are insecure

# Two Attacks Found

- Define propositions

```
IhiddenfriendA if Intruder.IwithA = true;
BconnectedA if (Bob.currentconnection = alice);
```

- MCMAS verifies the following `Formulae`
- Security attack: intruder breaks access control of friendship relation

```
EF(IhiddenfriendA);
AF(BconnectedA -> IhiddenfriendA);
```

- Privacy attack: intruder learns who the involved parties of a friendship are

```
BconnectedA -> K(Intruder, AknowsB);
AG(!BconnectedA -> !K(Intruder, AknowsB));
```

# Fixing the Protocol

- To fix the security bug introduce *authentication*
- Instead of:

$$B \rightarrow A|\, t_1 : (name_B, t_0, \{K_{A\rightarrow B}\}) = foaf_B$$

- Have now

$$B \rightarrow A|\, t_1 : \left(name_B, t_0, K^{-1}_{B\rightarrow A}(t_1), \{K_{A\rightarrow B}(K_S)\}\right) = foaf_B.$$

- *foaf*$_B$ has signed timestamp $K^{-1}_{B\rightarrow A}(t_1)$
- Session key $K_S$ serves to encrypt $A$'s data for $B$'s download.

⇒ Overkill for friendship page download security?

⇒ Doesn't overcome privacy issue! Need unobservability.

# Overview

1. The Hidden Friendship Protocol
2. MCMAS Modelchecking the Protocol
3. Summary and Discussion Points

# Summary and Discussion Points

- Hidden friendship protocol serves privacy
- MCMAS analysis allows fixing security
- Privacy issue still an open challenge
- Modelling and verification of friendship protocol attacks not to *prove properties* but to *detect attacks* and improve
- Future Work: Integrated approach with Isabelle's Inductive Approach [7,8] similar to DNSsec analysis [3]

# References I

[1] M. Burrows, M. Abadi, and R. Needham. A logic of authentication. *ACM Transactions on Computer Systems*, 8:18–36, 1990.

[2] E. M. Clarke, O. Grumberg, S. Jha, Y. Lu, and H. Veith, Counterexample-guided abstraction refinement, *CAV*. LNCS 1855, Springer, 2000.

[3] F. Kammüller, Y. Kirsal-Ever, X. Cheng. DNSsec in Isabelle – Replay Attack and Origin Authentication. Submited to *IEEE SMC 2013*.

[4] FOAF project. *The Friend of a Friend (FOAF) project*, `www.foaf-project.org`. Accessed 10.8.2013.

[5] A. Lomuscio, H. Qu, and F. Raimondi, Mcmas: A model checker for the verification of multi-agent systems, *CAV*, LNCS 5643,Springer, 2009.

[6] G. Lowe. An attack on the needham-schroeder public-key authentication protocol.

*Information Processing Letters*, 56:131–133, 1995.

# References II

[7] T. Nipkow, L. C. Paulson, and M. Wenzel. *Isabelle/HOL – A Proof Assistant for Higher-Order Logic*, 2283 LNCS. Springer-Verlag, 2002.

[8] L. C. Paulson, The inductive approach to verifying cryptographic protocols, *Journal of Computer Security*, vol. 6, no. 1–2, pp. 85–128, 1998.

[9] S. Preibusch and A. R. Beresford. Establishing distributed hidden friendship relations. *Seventeenth International Workshop on Security Protocols*, 2009.

[10] C. W. Probst, R. H. Hansen, and F. Nielson. Where Can an Insider Attack? *Formal Aspects of Security and Trust, FAST'06.* LNCS 4691, Springer 2006.

[11] C. W. Probst, J. Hunker, D. Gollmann, and M. Bishop, Eds., *Insider Threats in Cybersecurity.* Springer, 2010.

# Add on to Fixing the Protocol

$$B \to A|\, t_1 : \Big( name_B, t_0, K^{-1}_{B\to A}(t_1), \{K_{A\to B}(K_S)\} \Big) = foaf_B.$$

When $A$ receives this request she uses the key $K_{B\to A}$ in her $foaf_A$ received from $B$ in the Establishment phase (see page 6) to first restore the time stamp

$$K_{B\to A}(K^{-1}_{B\to A}(t_1)) = t_1$$

and then verify its timeliness, i.e., $|t_1 - \text{current time}| \leq \epsilon$ where $\epsilon$ is a threshold. The threshold $\epsilon$ must be chosen such that it admits reasonable latency in distributed systems while being small enough to exclude successful observation and replay by an Intruder. Assuming synchronised clocks this simple authentication mechanism authenticates $B$ to $A$ and avoids the replay attack within the boundaries of reasonable assumptions, e.g. times for threshold $\epsilon$.

# Friendship Protocol MCMAS – Agent Alice

```
Agent Alice
  Vars:
    initialpermission : { none };
    currentpermission: { accesstoB, accesstoI, none };
  end Vars
    Actions = { wait, openaccessB, openaccessI, sendfoaftoB };
  Protocol:
    currentpermission = none : { wait };
    currentpermission = accesstoB : { openaccessB };
    currentpermission = accesstoI: { openaccessI };
  end Protocol
  Evolution:
    currentpermission = accesstoB if
    (Environment.foafcontainsKAB = true and Bob.Action = sendfoaftoA);
    currentpermission = accesstoI if
    (Environment.foafcontainsKAB = true
     and Intruder.Action = sendfoaftoA);
  end Evolution
  end Agent
```

# MCMAS – Agent Bob and Environment

```
Agent Bob
 Vars:   initialconnection : { none };
  currentconnection: { alice, none };
 end Vars
 Actions = { wait, sendfoaftoA };
 Protocol:
  currentconnection = none : { sendfoaftoA };
  currentconnection = alice : { wait };
 end Protocol
 Evolution:
  currentconnection = alice if (Alice.Action = openaccessB);
 end Evolution
end Agent
Agent Environment
 Obsvars:
  foafcontainsKAB: boolean;
 end Obsvars
 Evolution:
  (foafcontainsKAB = true) if (Bob.Action = sendfoaftoA);
 end Evolution
end Agent
```

# MCMAS – Agent Intruder

```
Agent Intruder
Vars:
 initialstate: { noKey };
 currentstate: { noKey, seenfoafB, seenfoafA };
- I manages to connect with A
 IwithA: boolean;
end Vars
 Actions = { listen, sendfoaftoA, sendfoaftoB };
Protocol:
 currentstate = noKey: { listen };
 currentstate = seenfoafA: { sendfoaftoA };
 currentstate = seenfoafB: { sendfoaftoB };
end Protocol
Evolution:
 currentstate = seenfoafA if (Bob.Action = sendfoaftoA);
 currentstate = seenfoafB if (Alice.Action = sendfoaftoB);
 IwithA = true if (Alice.Action = openaccessI);
end Evolution
end Agent
```

# MCMAS – Propositions and Proved Assertions

```
Evaluation
 IhiddenfriendA if Intruder.IwithA = true;
 BconnectedA if (Bob.currentconnection = alice);
 AadmittedB if (Alice.currentpermission = accesstoB);
 AknowsB if (Alice.currentpermission = accesstoB and
    Bob.currentconnection = alice);
end Evaluation
Formulae
 EF(IhiddenfriendA);
 AF(BconnectedA -> IhiddenfriendA);
 AF(AadmittedB -> IhiddenfriendA);
end Formulae
```

# RDF

RDF (Resource Description Framework), which is the standard for encoding metadata and other knowledge on the Semantic Web. In the Semantic Web, computer applications make use of structured information spread in a distributed and decentralized way throughout the current web. RDF is an abstract model, a way to break down knowledge into discrete pieces, and while it is most popularly known for its RDF/XML syntax, RDF can be stored in a variety of formats.
source: `http://www.rdfabout.com/intro/`