Reasoning about Accountability within Delegation

Bruno Crispo (Cryptomathic, Italy)
and
Giancarlo Ruffo (University of Turin, Italy)

Overview

- Definition of accountability and delegation;
- Symbols;
- Assumptions and postulates;
- Analysis of delegation protocols;
- Case studies;
- Application fields and Conclusion.

Accountability

- *The property whereby the association of a principal with an object, an action or a right can be proved by a third party.*
- In electronic transactions we have to guarantee accountability as in conventional transactions.

Delegation

- We are focusing on delegation protocols;
- Delegation is used to transfer accountabilities among principals;
- Though a lot of analysis schemas have been proposed, delegation protocols were not considered.

Previous work

- Logic of authentication [Burrows, Abadi, Needham, 1990]
- Provability framework [Kailar, 1995]

Provability

- The ability of partecipants in a protocol to prove a statament *x* to a third party;
- The *prove* of *x* is a set of statements produced from known assumptions that convince another principal about *x*;
- In practice, we need to convince a particular third party (e.g.; a judge).

Provability (2)

- The logic of *beliefs* focuses on what can be proved only by partecipants of the protocol…
- But the external point of view is essential to accountability;
- We adopt the Kailar's framework based on the notion of *provability*.

Symbols and Concepts

- *A, B, …*: **principals** involved in a communication protocol;
- *x, y, …*: **statements**, i.e., interpretations of messages exchanged between principals;
- **Proof of x**: something that convinces another principal of *x;*

Symbols and Concepts (2)

- **Right**: right to use a resource or right to perfom an action;
- Ω, Δ**: set of rights**;
- We will not introduce other symbols, just common phrases written concatenated (i.e., CanProve or CanExercise, …).

Symbols and Concepts (3)

- **A CanProve x** – to any third party;
- **K Authenticates A** – we can associate A to any statement encrypted with K^{-1}
- **x in m** – *x* is the interpretation of message *m* (or of a field of *m*);
- **A Says x** – A is accountable of x

$$\frac{\textbf{A Says (x, y)}}{\textbf{A Says x}}$$

Symbols and Concepts (4)

- **A Receives m SignedWith K^{-1}** – we can use the following postulate:

$$\frac{\textbf{A Receives m SignedWith } K^{-1} \textbf{ ; x in m}}{\textbf{A Receives x SignedWith } K^{-1}}$$

- **A isTrustedOn x** – A has the authority to endorse *x*.

A new framework symbol: CanExercise

- **A CanExercise** Ω – to associate principals to a set of rights;
- **A CanExercise** Ω **with K** – to specify the authentication key that a principal uses to exercise her rights;

Assumptions

- Signature algorithms are assumed to be strong enough
  - to be undisputably associated with a singol user;
  - to resist against the search of another principal's provate key;
  - to withstand birthdays attacks;

Assumptions (2)

- Signature algorithms provides message origin authentication, message content integrity and non-repudiation;
- Principals are trusted not to share their private keys with whom they do not wish to be accountable;
- A principal trusts a statement if she is an authority of it or if she is convinced on the validity of it by a trusted party;

Assumptions (3)

- **Message integrity**: not possible to fake a signed message;
- **Availability of service**: if *A CanProve x* then *A* has the ability to send all the message for proving *x;*
- **Certificate revocation**: statements proved by revoked key are valid if they were signed when the related certs were valid.

Postulates in the form: $\frac{\mathbf{P;\ Q}}{\mathbf{R}}$

- Conjunction:

$$\frac{\mathbf{A\ CanProve\ x;\ A\ CanProve\ y}}{\mathbf{A\ CanProve\ (x,\ y)}} \quad [\text{Conj}]$$

- Inference:

$$\frac{\mathbf{A\ CanProve\ x;\ x \Rightarrow y}}{\mathbf{A\ CanProve\ y}} \quad [\text{Inf}]$$

Accountability property of digital signatures

$$[\text{Sign}]\ \frac{\textbf{A Receives m SignedWith } K^{-1}\textbf{; x in m; A CanProve (K Authenticates B)}}{\textbf{A CanProve (B Says x)}}$$

Trust Relationships

$$[\text{Trust}]\ \frac{\mathbf{A\ CanProve\ (B\ Says\ x);}\quad \mathbf{A\ CanProve\ (B\ isTrustedOn\ x);}}{\mathbf{A\ CanProve\ x}}$$

A specification of the framework: CanExercise postulates

- A principal can exercise a set of rights if another principal gave her the related permissions.
- Permissions can be given by:
  - A trusted authority (e.g., a system admin.)
  - A principal (e.g., **delagator**).

CanExercise postulates

[CanEx1]

$$\frac{\textbf{A CanExercise } \Omega\textbf{;}\quad \textbf{A Says (}\text{delegation of } \Omega \textbf{ to B);}\quad \textbf{K}_{\textbf{Del}} \textbf{ Authenticates B}}{\textbf{B CanExercise } \Omega \textbf{ with K}_{\textbf{Del}}}$$

[CanEx2]

$$\frac{\textbf{A CanExercise } \Omega \textbf{ with K}}{\textbf{A CanExercise } \Omega}$$

Analysis of a delegation protocol

- Step 1: Protocol Description;
- Step 2: Protocol Reformulation (using this framework);
- Step 3: Protocol analysis, i.e. reaching the following goals:

[G1] *D* CanProve (*D* CanExercise Ω with K_{Del})

[G2] *G* CanProve (K_{Del} Authenticates *D*)

Where D is the delegate and G is the delegator.

Analysis of SPX protocol with support for delegation [Tardo and Agalappan, 91]

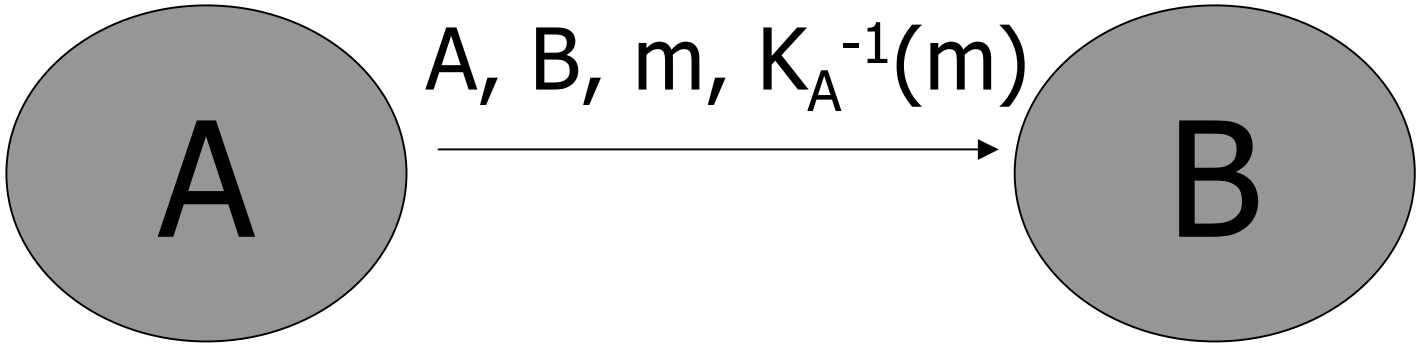
- Principals use *authentication tokens* to securely exchange session keys;
- Kailar already analyzed this protocol;
- SPX permits grantor to delegate grantee the possibility to act on grantor's behave;
- Grantor securely *shares* with grantee delegation key $\mathbf{K_{Del}}$.

Analysis of SPX protocol with support for delegation (2)

- Kailar proved:
  D CanProve (**K_{Del}** Authenticates *G*)
- Delegate can exercise transferred rights, but Grantor is still accountable for them.

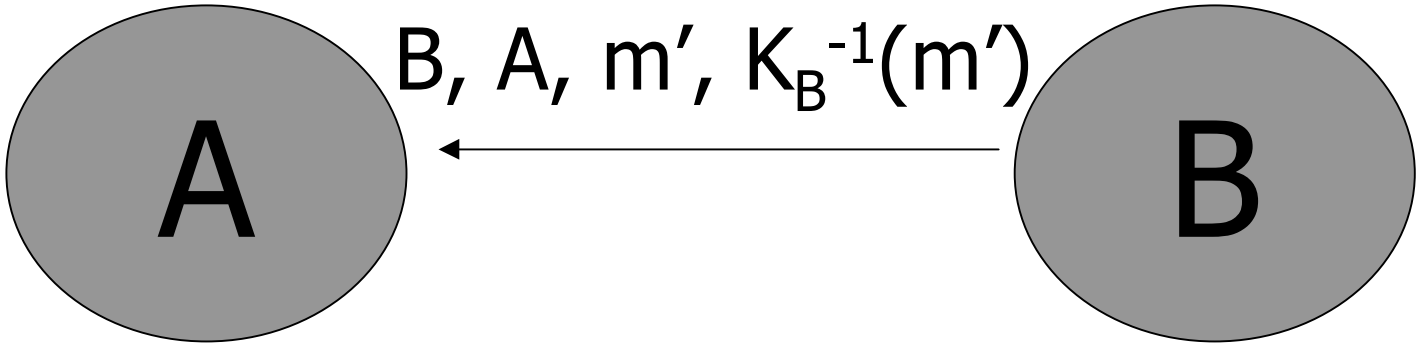
The Delegation of Accountability protocol

- It is based on *delegation tokens*;
- Protocol description:



m = "A wishes to delegate to B accountability for Ω"

The Delegation of Accountability protocol

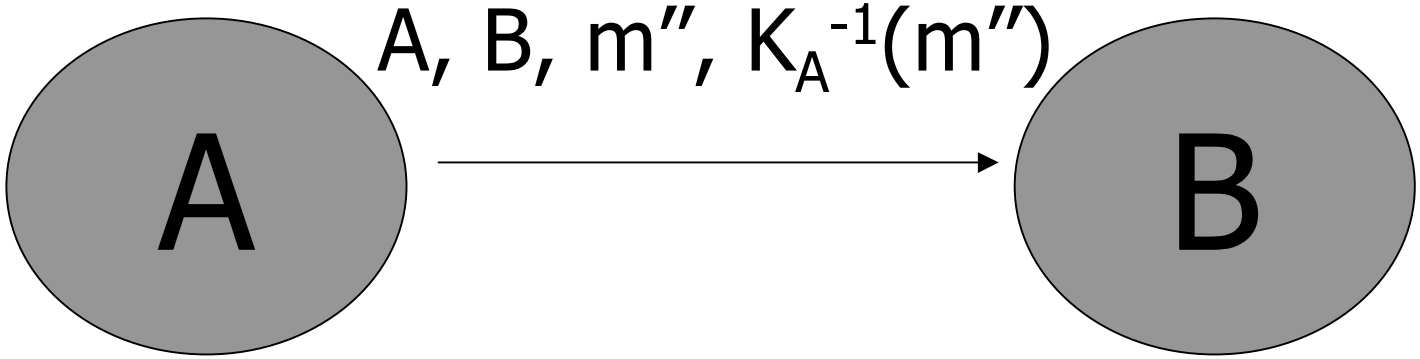


m′ = "B accepts Ω and she will exercise Ω using K_{Del}"

The Delegation of Accountability protocol

Delegation token

A, B, m″, K_A^{-1}(m″)



m″ = "Ω, time span, K_A, K_{Del}"

Reformulating the protocol

- B Receives ((A wishes to delegate to B accountability for Ω) SignedWith K_A^{-1})
- A Receives ((K_{Del} Authenticates B) SignedWith K_B^{-1})
- B Receives ((delegation of Ω to B) SignedWith K_A^{-1})

Initial Assumptions

[A1] A CanProve (K_A Authenticates A)

[A2] B CanProve (K_B Authenticates B)

[A3] B CanProve (K_{Del} Authenticates B)

[A4] B CanProve (A CanExercise Ω)

[A5] A CanProve (B isTrustedOn (K_{Del} Authenticates B)

[A6] A CanProve (K_B Authenticates B)

[A7] B CanProve (K_A Authenticates A)

Analysis and Goal (complete proof in the paper)

- Applying our postulates from initial assumptionsl, we can reach these goals:

[G1] B CanProve (B CanExercise Ω with K_{Del})

[G2] A CanProve (K_{Del} Authenticates B)

- This delegation protocol respects accountability property.

Further work and Conclusion

- Delegation of Access Control rights in a file system needs accountability; a prototype will be implemented;
- Other applications are under study;
- Collaborations are welcome!

Contacts

- Giancarlo Ruffo (University of Turin, Italy) – ruffo@di.unito.it
- Bruno Crispo (Cryptomathic, Italy) – bruno.crispo@cryptomathicit

- **Security Group**, Dept. of Computer Science, University of Turin – http://security.di.unito.it/