Graphical Modeling of Security Ceremonies

Cristian Prisacariu
(thanks to Audun Jøsang)

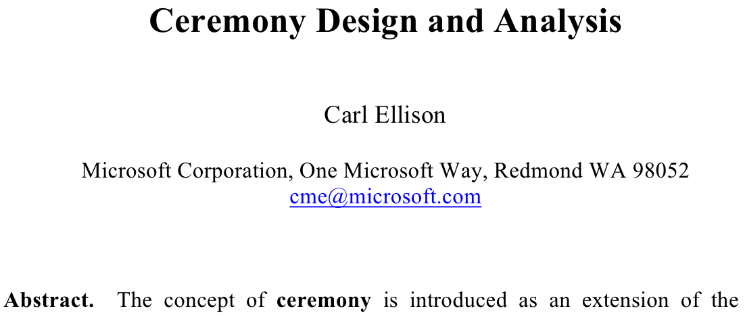
Precise Modeling and Analysis group (PMA),
University of Oslo

at
FRISC Winter School

6 May 2014, Finse, Norway.

What are Security Ceremonies

The 2007 paper that introduced the concept.



Ceremony Design and Analysis

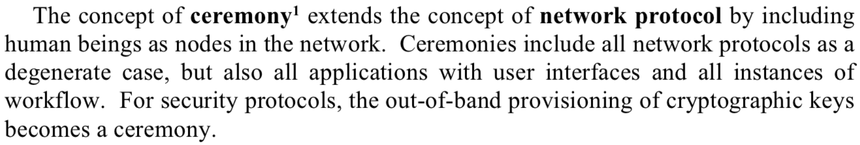
Carl Ellison

Microsoft Corporation, One Microsoft Way, Redmond WA 98052
cme@microsoft.com

Abstract. The concept of **ceremony** is introduced as an extension of the

What are Security Ceremonies

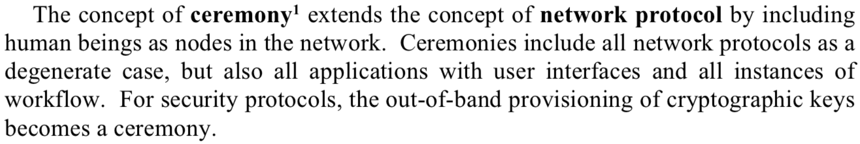
One definition:



The concept of **ceremony**[1] extends the concept of **network protocol** by including human beings as nodes in the network. Ceremonies include all network protocols as a degenerate case, but also all applications with user interfaces and all instances of workflow. For security protocols, the out-of-band provisioning of cryptographic keys becomes a ceremony.

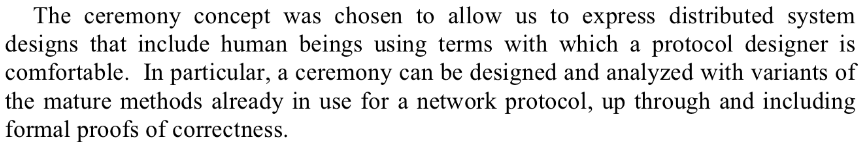
What are Security Ceremonies

One definition:



The concept of **ceremony**[1] extends the concept of **network protocol** by including human beings as nodes in the network. Ceremonies include all network protocols as a degenerate case, but also all applications with user interfaces and all instances of workflow. For security protocols, the out-of-band provisioning of cryptographic keys becomes a ceremony.

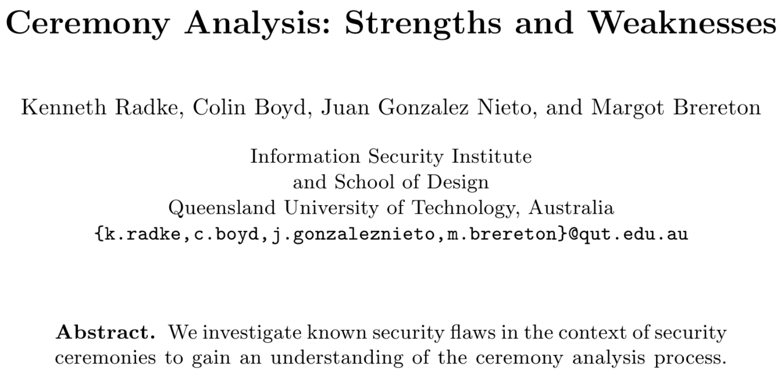
The next paragraph continues as:



The ceremony concept was chosen to allow us to express distributed system designs that include human beings using terms with which a protocol designer is comfortable. In particular, a ceremony can be designed and analyzed with variants of the mature methods already in use for a network protocol, up through and including formal proofs of correctness.

What are Security Ceremonies

Another paper on Ceremonies from 2011:



Ceremony Analysis: Strengths and Weaknesses

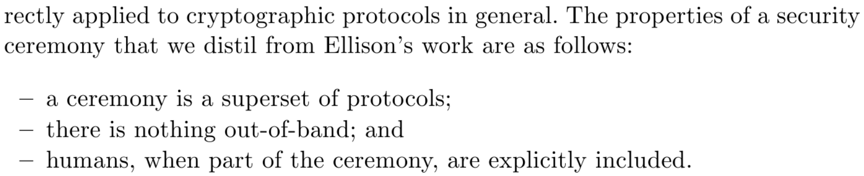
Kenneth Radke, Colin Boyd, Juan Gonzalez Nieto, and Margot Brereton

Information Security Institute
and School of Design
Queensland University of Technology, Australia
{k.radke,c.boyd,j.gonzaleznieto,m.brereton}@qut.edu.au

Abstract. We investigate known security flaws in the context of security ceremonies to gain an understanding of the ceremony analysis process.

What are Security Ceremonies

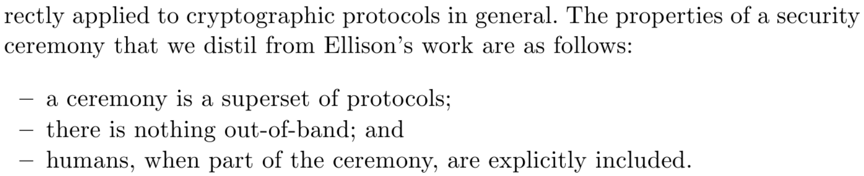
Understand Security Ceremonies as:


rectly applied to cryptographic protocols in general. The properties of a security ceremony that we distil from Ellison's work are as follows:

- a ceremony is a superset of protocols;
- there is nothing out-of-band; and
- humans, when part of the ceremony, are explicitly included.

What are Security Ceremonies

Understand Security Ceremonies as:


rectly applied to cryptographic protocols in general. The properties of a security ceremony that we distil from Ellison's work are as follows:

– a ceremony is a superset of protocols;
– there is nothing out-of-band; and
– humans, when part of the ceremony, are explicitly included.

No formal definitions and no formalism in any of these two papers.
Just case analyses and informal expository definitions.

What is a good formalism for Security Ceremonies?

Two main technical challenges in Security Ceremonies

1. Composition of Protocols

   in parallel, sequential, vertical, etc.

2. Incorporating the Human nodes

   in such a way to make analysis and security proofs possible.

Two main technical challenges in Security Ceremonies

1. Composition of Protocols

   in parallel, sequential, vertical, etc.

   - from process algebras approach
   - or Protocol Composition Logic (Stanford & ASECOlab)

2. Incorporating the Human nodes

   in such a way to make analysis and security proofs possible.

   - from psychology/cognition analyses (at Univ. College London)
   - or sociology (Actor-network theory book of B.Latour)

Two main technical challenges in Security Ceremonies

1. Composition of Protocols

   in parallel, sequential, vertical, etc.

   - from process algebras approach
   - or Protocol Composition Logic (Stanford & ASECOlab)

2. Incorporating the Human nodes

   in such a way to make analysis and security proofs possible.

   - from psychology/cognition analyses (at Univ. College London)
   - or sociology (Actor-network theory book of B.Latour)

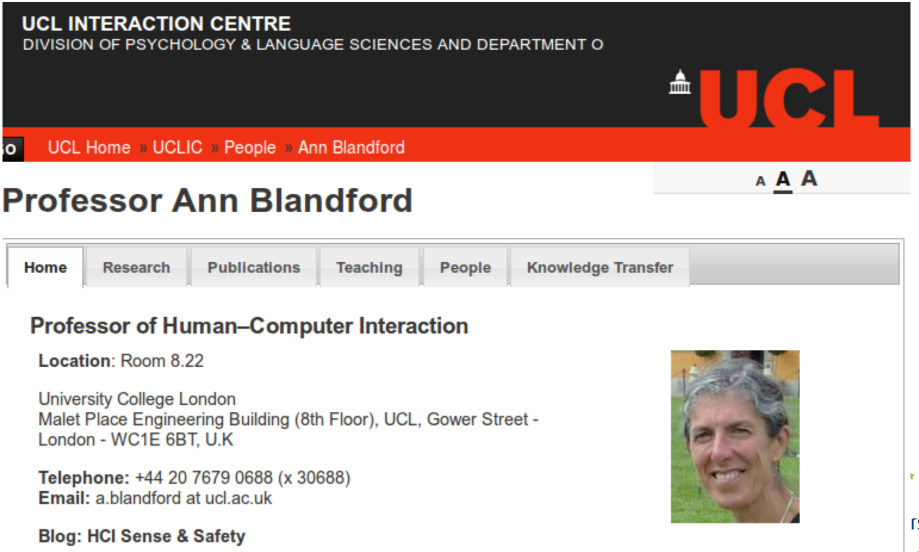
Conclusion on Ceremonies

- Security Ceremonies ask to
  - model the human nodes,
  - combine protocols,
  - be explicit about the assumptions.
- Security Ceremonies
  - have no satisfactory formalization (my opinion)
  - subsume security/network protocols (thus their formalization must subsume existing formalisms for analysing such protocols)
- Inspiration from varied fields should be sought, like
  - sociology (actor network theory)
  - psychology (investigations of cognitive aspects of humans)
  - probabilistic/stochastic models for humans
  - security/network protocol formalisms like process algebras, logics
  - existing tools like theorem provers, model checkers,
  - graphical languages (ceremony developers will come from various fields)

Modeling the Human nodes in Security

Modeling the Human nodes in Security

1. An approach from Psychology



UCL INTERACTION CENTRE
DIVISION OF PSYCHOLOGY & LANGUAGE SCIENCES AND DEPARTMENT O

UCL

UCL Home » UCLIC » People » Ann Blandford

A A A

Professor Ann Blandford

Home | Research | Publications | Teaching | People | Knowledge Transfer

Professor of Human–Computer Interaction

Location: Room 8.22

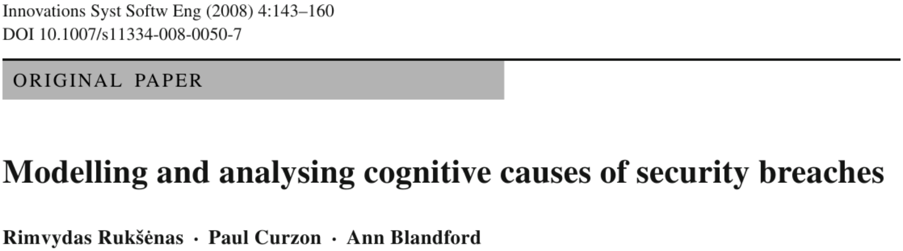
University College London
Malet Place Engineering Building (8th Floor), UCL, Gower Street - London - WC1E 6BT, U.K

Telephone: +44 20 7679 0688 (x 30688)
Email: a.blandford at ucl.ac.uk

Blog: HCI Sense & Safety

Modeling the Human nodes in Security

1. An approach from Psychology



Innovations Syst Softw Eng (2008) 4:143–160
DOI 10.1007/s11334-008-0050-7

ORIGINAL PAPER

Modelling and analysing cognitive causes of security breaches

Rimvydas Rukšėnas · Paul Curzon · Ann Blandford

- The user is modeled as non-deterministic choice of plausible actions
- Modeling done within higher order logic and specify a transition system (i.e., very close to how the information protocol is modeled)

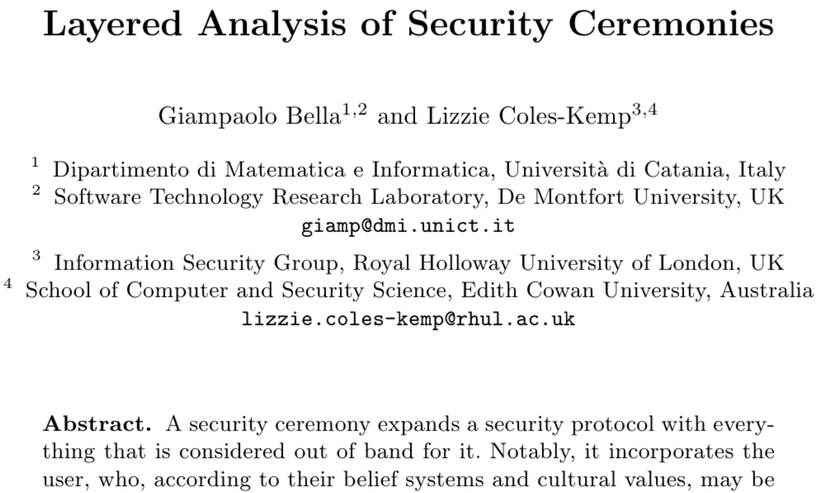
Modeling the Human nodes in Security

Modeling the Human nodes in Security

2. An approach from Sociology

Modeling the Human nodes in Security

2. An approach from Sociology



Layered Analysis of Security Ceremonies

Giampaolo Bella[1,2] and Lizzie Coles-Kemp[3,4]

[1] Dipartimento di Matematica e Informatica, Università di Catania, Italy
[2] Software Technology Research Laboratory, De Montfort University, UK
giamp@dmi.unict.it

[3] Information Security Group, Royal Holloway University of London, UK
[4] School of Computer and Security Science, Edith Cowan University, Australia
lizzie.coles-kemp@rhul.ac.uk

Abstract. A security ceremony expands a security protocol with everything that is considered out of band for it. Notably, it incorporates the user, who, according to their belief systems and cultural values, may be

Modeling the Human nodes in Security

2. An approach from Sociology

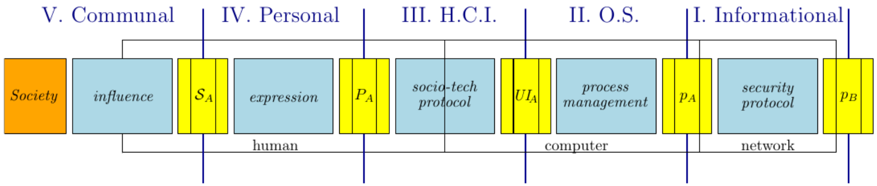
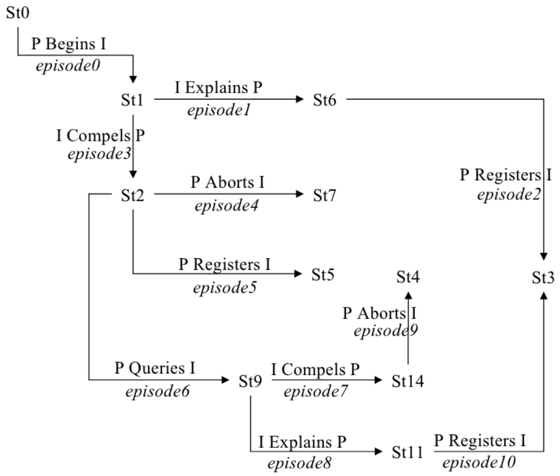


Fig. 1. The security ceremony model underlying our formal analysis

Modeling the Human nodes in Security

2. An approach from Sociology



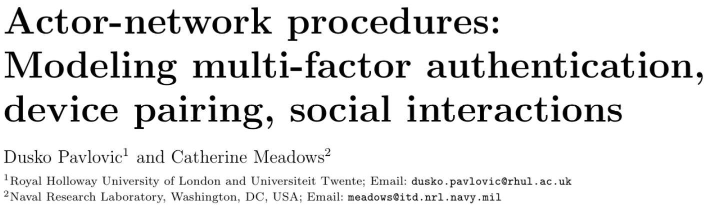
My focus

- Look for a graphical formal language for analyzing security ceremonies
- like Statecharts are for reactive systems, used in avionics or automotive

Actor Network Procedures

Actor Network Procedures

©2011 (draft under development)



Actor-network procedures: Modeling multi-factor authentication, device pairing, social interactions

Dusko Pavlovic[1] and Catherine Meadows[2]

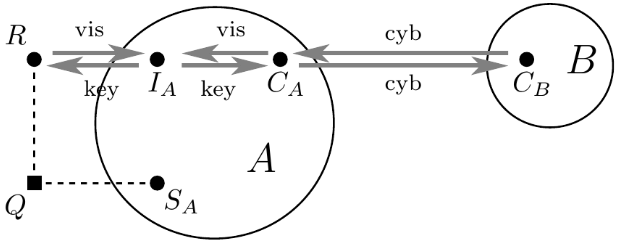
[1]Royal Holloway University of London and Universiteit Twente; Email: dusko.pavlovic@rhul.ac.uk
[2]Naval Research Laboratory, Washington, DC, USA; Email: meadows@itd.nrl.navy.mil

The Structure of an Actor Network Procedure

Structure Description Language essentially uses:

- a tree-like containment relation of (minimal) configurations, each being controlled by a principal of the ceremony
  (inspired from location models as ambient calculi or bigraphs)
- channels between two configurations

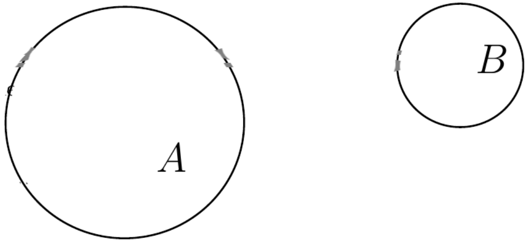
The Structure of an Actor Network Procedure



The Structure of an Actor Network Procedure

Definition 2.1. An *actor-network* consists of the following sets:

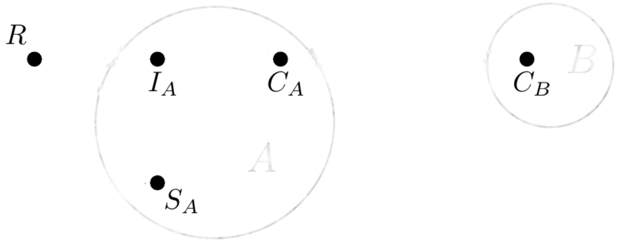
- *identities*, or *principals* $\mathcal{J} = \{A, B, \ldots\}$,



The Structure of an Actor Network Procedure

Definition 2.1. An *actor-network* consists of the following sets:

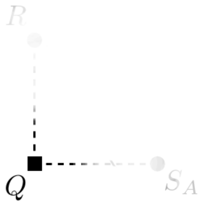
- *identities*, or *principals* $\mathcal{J} = \{A, B, \ldots\}$,
- *nodes* $\mathcal{N} = \{M, N, \ldots\}$,



The Structure of an Actor Network Procedure

Definition 2.1. An *actor-network* consists of the following sets:

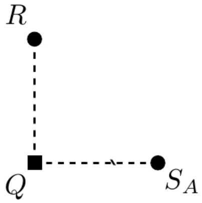
- *identities*, or *principals* $\mathcal{J} = \{A, B, \ldots\}$,
- *nodes* $\mathcal{N} = \{M, N, \ldots\}$,
- *configurations* $\mathcal{P} = \{P, Q, \ldots\}$, where a configuration can be
  - a finite set of nodes, or
  - a finite set of configurations;



The Structure of an Actor Network Procedure

Definition 2.1. An *actor-network* consists of the following sets:

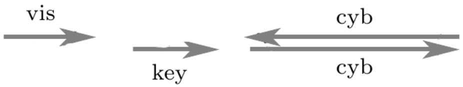
- *identities*, or *principals* $\mathcal{J} = \{A, B, \ldots\}$,
- *nodes* $\mathcal{N} = \{M, N, \ldots\}$,
- *configurations* $\mathcal{P} = \{P, Q, \ldots\}$, where a configuration can be
  - a finite set of nodes, or
  - a finite set of configurations;



The Structure of an Actor Network Procedure

Definition 2.1. An *actor-network* consists of the following sets:

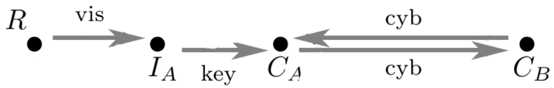
- *identities*, or *principals* $\mathcal{J} = \{A, B, \ldots\}$,
- *nodes* $\mathcal{N} = \{M, N, \ldots\}$,
- *configurations* $\mathcal{P} = \{P, Q, \ldots\}$, where a configuration can be
  - a finite set of nodes, or
  - a finite set of configurations;
- *channels* $\mathcal{C} = \{f, g, \ldots\}$, and
- *channel types* $\Theta = \{\tau, \varsigma, \ldots\}$



The Structure of an Actor Network Procedure

Definition 2.1. An *actor-network* consists of the following sets:

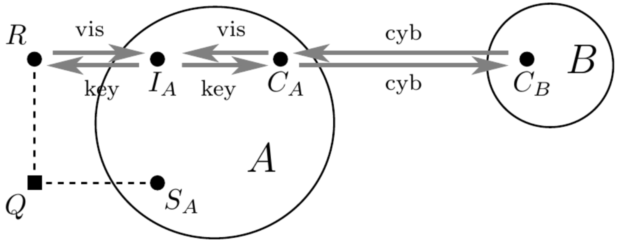
- *identities*, or *principals* $\mathcal{J} = \{A, B, \ldots\}$,
- *nodes* $\mathcal{N} = \{M, N, \ldots\}$,
- *configurations* $\mathcal{P} = \{P, Q, \ldots\}$, where a configuration can be
  - a finite set of nodes, or
  - a finite set of configurations;
- *channels* $\mathcal{C} = \{f, g, \ldots\}$, and
- *channel types* $\Theta = \{\tau, \varsigma, \ldots\}$



The Structure of an Actor Network Procedure

Definition 2.1. An *actor-network* consists of the following sets:

- *identities*, or *principals* $\mathcal{J} = \{A, B, \ldots\}$,
- *nodes* $\mathcal{N} = \{M, N, \ldots\}$,
- *configurations* $\mathcal{P} = \{P, Q, \ldots\}$, where a configuration can be
  - a finite set of nodes, or
  - a finite set of configurations;
- *channels* $\mathcal{C} = \{f, g, \ldots\}$, and
- *channel types* $\Theta = \{\tau, \varsigma, \ldots\}$



The Structure of an Actor Network Procedure

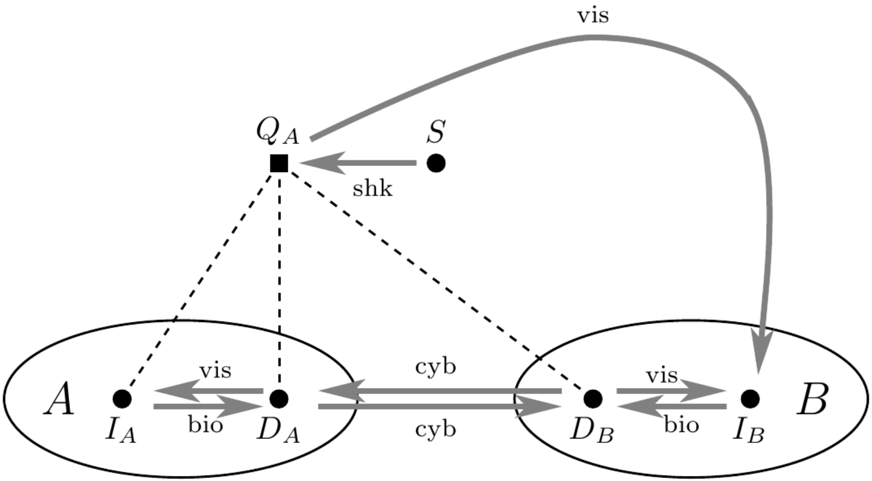


Fig. 4. Actor-network for device handshake

The Behavior

The Behavior

3.1. Computation and communication

- Computation consists of events, localized at nodes or configurations. An event that is controlled by a principal is an action.

The Behavior

3.1. Computation and communication

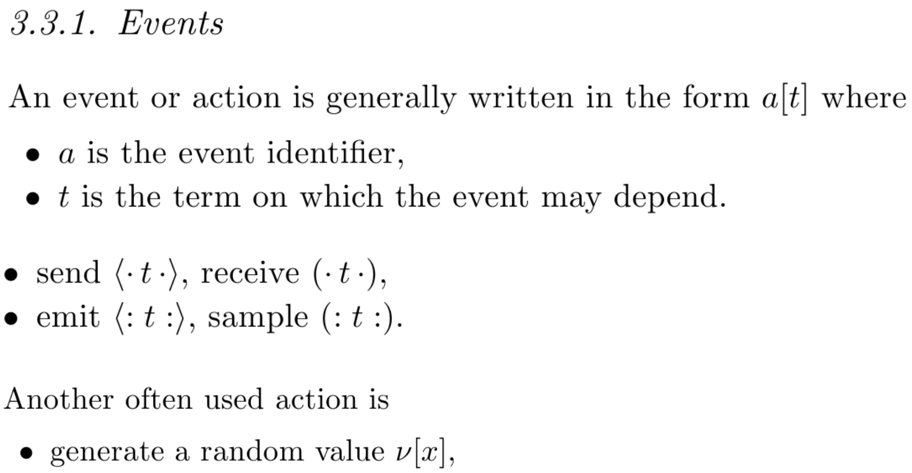
- Computation consists of events, localized at nodes or configurations. An event that is controlled by a principal is an action.
- Communication consists of information flows along the channels. Each flow corresponds to a pair of events:
  - a *write* event at the entry of the channel, and
  - a *read* event at the exit of the channel.

The Behavior

3.1. Computation and communication

- Computation consists of events, localized at nodes or configurations. An event that is controlled by a principal is an action.
- Communication consists of information flows along the channels. Each flow corresponds to a pair of events:
  - a *write* event at the entry of the channel, and
  - a *read* event at the exit of the channel.
- Flows can be:
  - messages, which consist of a send action at the entry of the channel, and a receive coaction at the exit;
- messages are described as terms (as in proc. alg. approach)
- actions can be parametrized by messages/data

The Behavior



3.3.1. Events

An event or action is generally written in the form $a[t]$ where

- a is the event identifier,
- t is the term on which the event may depend.

- send $\langle\cdot\, t\,\cdot\rangle$, receive $(\cdot\, t\,\cdot)$,
- emit $\langle:\, t\,:\rangle$, sample $(:\, t\,:)$.

Another often used action is

- generate a random value $\nu[x]$,

The nodes are also capable of performing various local operations, eg.: to execute the standard pseudo-code commands, like comparisons (t = s) or assignments (t := s).

The Behavior – Example

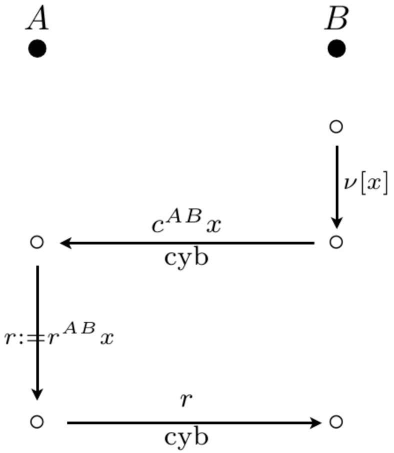


Fig. 6. Challenge-Response (CR) protocol template

Procedures – examples

Procedures – examples

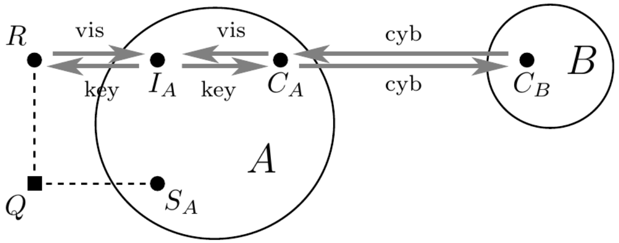


Fig. 3. A pervasive network: Online banking with a smart card reader

Procedures – examples

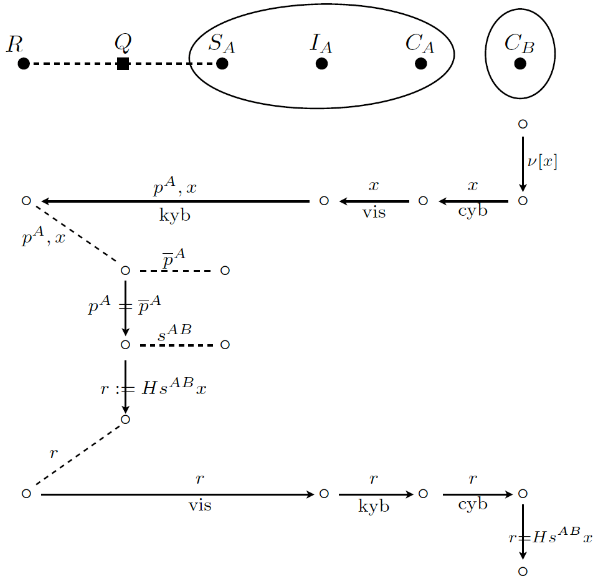


Fig. 7. Chip Authentication Program (CAP) procedure

Procedures – examples

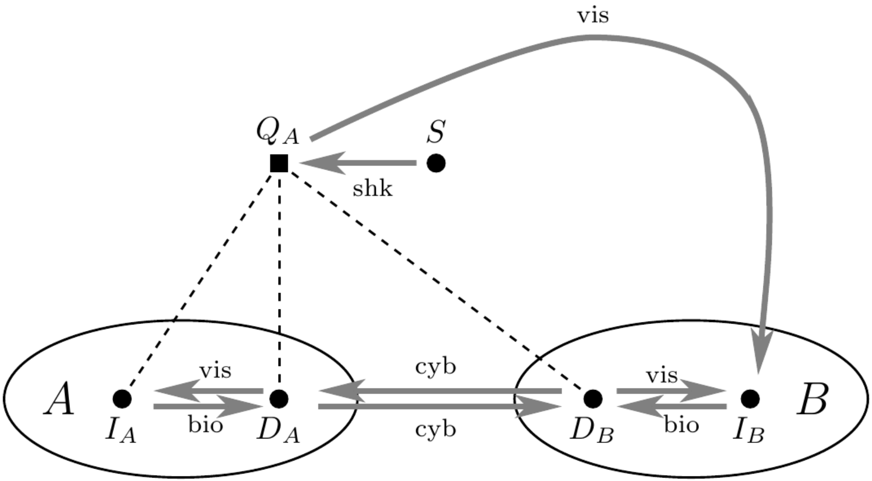


Fig. 4. Actor-network for device handshake

Procedures – examples

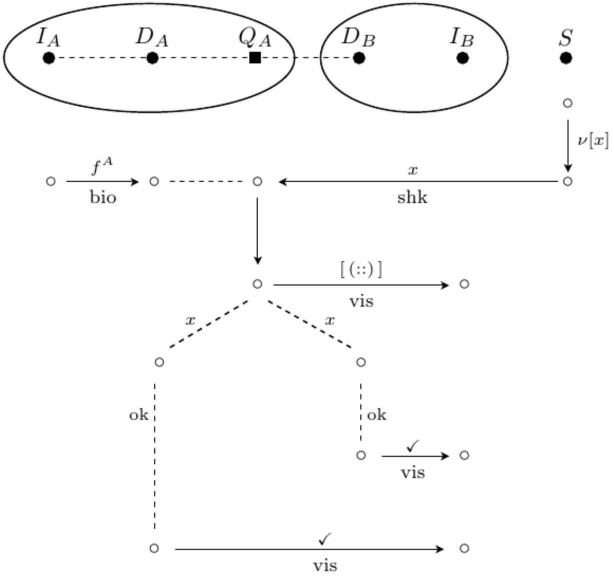


Fig. 8. Device handshake procedure

Conclusions on Actor Networks Procedures

- Are intended as a graphical formalism for describing and reasoning about security ceremonies
- Are inspired by
  - process algebras through the representation of messages as terms
  - strand spaces for the description of the runs
  - actor networks from sociology for the representation of configurations
  - protocol composition through the use of the PDL logic for reasoning about the models
- Acknowledge the need of concurrency when several parties interact
- There is room for improving the existing formalism.
- Correlations with existing formalisms are needed.
- Case studies are needed.
- Are general enough to be used outside security, with any communication/interaction ceremony.

Thank you for your attention!

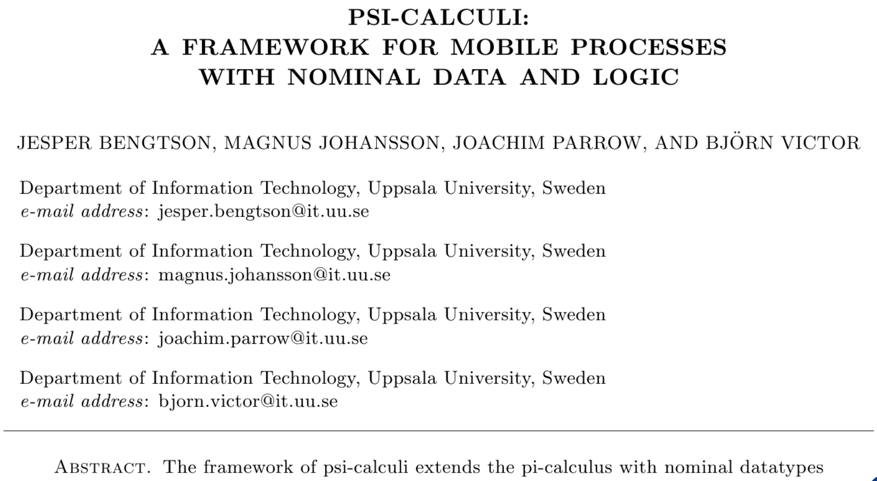
Psi-calculi semantics for ANPs

- Verification tools for security, like AVISPA, FDR, ProVerif, etc. use as input modeling languages based on process algebras like applied pi-calculus.
  These are textual.
- The aim is to do automatic verification of ANP ceremonies.
- A first step here toward this goal:

  give semantics to ANPs
  using the recent expressive psi-calculi (from Uppsala Univ.)
  as the underlying semantic framework
- Several benefits:
  - connect to the algebraic languages used by the tools;
  - investigate better the ANP graphical formalism;

Psi-calculi

©2011



PSI-CALCULI:
A FRAMEWORK FOR MOBILE PROCESSES
WITH NOMINAL DATA AND LOGIC

JESPER BENGTSON, MAGNUS JOHANSSON, JOACHIM PARROW, AND BJÖRN VICTOR

Department of Information Technology, Uppsala University, Sweden
e-mail address: jesper.bengtson@it.uu.se

Department of Information Technology, Uppsala University, Sweden
e-mail address: magnus.johansson@it.uu.se

Department of Information Technology, Uppsala University, Sweden
e-mail address: joachim.parrow@it.uu.se

Department of Information Technology, Uppsala University, Sweden
e-mail address: bjorn.victor@it.uu.se

ABSTRACT. The framework of psi-calculi extends the pi-calculus with nominal datatypes

Psi-calculi

- Are a framework where many other calculi can be defined
- e.g.: Milner's pi-calculus, cryptographic calculi like spi- and applied pi, location calculi like the distributed pi, polyadic calculi.
- typed psi-calculus exists
  thus subsuming many existing typed calculi
- For reasoning with psi-calculi one uses proof assistant tools like Isabell.

Psi-calculi

Definition 1 (Psi-calculus parameters)**.** A psi-calculus requires the three (not necessarily disjoint) nominal datatypes:

$\mathbf{T}$	the (data) terms, ranged over by M, N
$\mathbf{C}$	the conditions, ranged over by φ
$\mathbf{A}$	the assertions, ranged over by Ψ

and the four equivariant operators:

$\dot{\leftrightarrow} : \mathbf{T} \times \mathbf{T} \to \mathbf{C}$	Channel Equivalence
$\otimes : \mathbf{A} \times \mathbf{A} \to \mathbf{A}$	Composition
$\mathbf{1} : \mathbf{A}$	Unit
$\vdash \subseteq \mathbf{A} \times \mathbf{C}$	Entailment

and substitution functions $[\widetilde{a} := \widetilde{M}]$, substituting terms for names, on all of $\mathbf{T}$, $\mathbf{C}$ and $\mathbf{A}$.

Psi-calculi

To capture the standard pi-calculus instantiate as:

$\mathbf{T}$	the (data) terms, ranged over by M, N
$\mathbf{C}$	the conditions, ranged over by φ
$\mathbf{A}$	the assertions, ranged over by Ψ

$\dot{\leftrightarrow} : \mathbf{T} \times \mathbf{T} \to \mathbf{C}$	Channel Equivalence
$\otimes : \mathbf{A} \times \mathbf{A} \to \mathbf{A}$	Composition
$\mathbf{1} : \mathbf{A}$	Unit
$\vdash \subseteq \mathbf{A} \times \mathbf{C}$	Entailment

$$
\begin{aligned}
\mathbf{T} &\stackrel{\text{def}}{=} \mathcal{N} \\
\mathbf{C} &\stackrel{\text{def}}{=} \{a = b : a, b \in \mathbf{T}\} \\
\mathbf{A} &\stackrel{\text{def}}{=} \{1\} \\
\dot{\leftrightarrow} &\stackrel{\text{def}}{=} = \\
\otimes &\stackrel{\text{def}}{=} \lambda\Psi_1, \Psi_2.\, 1 \\
\mathbf{1} &\stackrel{\text{def}}{=} 1 \\
\vdash &\stackrel{\text{def}}{=} \{(1, a = a) : a \in \mathcal{N}\}
\end{aligned}
$$

Psi-calculi

Agents are defined as:

Definition 6 (psi-calculus agents)**.** Given valid psi-calculus parameters as in Definitions 1 and 3, the psi-calculus *agents*, ranged over by $P, Q, \ldots$, are of the following forms.

$\mathbf{0}$	Nil
$\overline{M}N\,.\,P$	Output
$\underline{M}(\lambda\widetilde{x})N\,.\,P$	Input
$\mathbf{case}\ \varphi_1 : P_1 \,[]\, \cdots \,[]\, \varphi_n : P_n$	Case
$(\nu a)P$	Restriction
$P \mid Q$	Parallel
$!P$	Replication
$(\!\vert\Psi\vert\!)$	Assertion

ANP semantics as Psi-calculi

- Nodes are seen as minimal configurations, i.e., without sub-configs
- Structure of configurations is modeled through lists of ancestors

$$[l_1, \cdots, l_n] \in \mathbf{T} \text{ for } l_i \in \mathcal{N}_C.$$

- Channel names are attached to configurations
- Any term can be communicated on such a channel

$$\overline{[l_1, \cdots, l_n]c}\langle N \rangle \text{ as Output, or } \quad \underline{[l_1, \cdots, l_n]c}\langle (\nu \tilde{a}) N \rangle \text{ as Input}$$

- Identities can control configurations to which channels are attached

$$i[l_1, \cdots, l_n]c \in \mathbf{T} \text{ for } i \in \mathcal{N}_I, l_i \in \mathcal{N}_A, c \in \mathcal{N}_C.$$

- Internal actions are communications on channels local to a config
- Sharing is also communication on channels shared inside a config
- Tests are done with the case constructs and term equality

ANP semantics as Psi-calculi

Example 3.1 *The local generation of the fresh value fr by Bank's computer $B[l_C]$, which is then communicated on the cyber channel is thus modeled as:*

$$(\nu\, loc)((\nu fr)\overline{B[l_C]loc}\langle fr\rangle \mid \underline{B[l_C]loc}\langle(\lambda x)x\rangle.\overline{B[l_C]cyb}\langle x\rangle).$$

The entailment relation defines when two terms are equal wrt. some assertion:

$$\Psi \vdash M = N \text{ iff } \vdash_\Sigma M = N$$

where Σ is the signature over which the message terms are built, and $\vdash_\Sigma$ is the equational logic entailment relation wrt.

- The behavior is given by the operational semantics of psi-calculi.

The Behavior of ANPs – some Questions

- Are all actions, communications (input/output)?
- Are all communications the same (input/output), only the channel name being different (and the message types)?
- Can all actions/events be observed? (Observation is interaction)
- Should we be concerned with unobservable actions/events?
- What is a channel type? (? random noise binary channel ?)

Processes / Procedures

3.3.2. Processes

Definition 3.2. A *process* $\mathcal{F}$ is a partially ordered multiset of localized events, i.e.

$$\mathcal{F} \;=\; \langle \mathcal{F}_{\mathbb{E}}, \mathcal{F}_{\mathcal{P}} \rangle : \mathbb{F} \to \mathbb{E} \times \mathcal{P}$$

where

- $(\mathbb{F}, \to)$ is a well-founded partial order, representing the structure time,
- $\mathbb{E}$ is a family of events, and
- $(\mathcal{P}, \subseteq)$ the partial order of configurations,

and they satisfy the requirements that

(a) if $\mathcal{F}_{\mathbb{E}}\phi$ is an action, then ©$(\mathcal{F}_{\mathcal{P}}\phi)$ is well defined, and
(b) if $\phi \to \psi$ in $\mathbb{F}$ then $\mathcal{F}_{\mathcal{P}}\phi \subseteq \mathcal{F}_{\mathcal{P}}\psi$ or $\mathcal{F}_{\mathcal{P}}\phi \supseteq \mathcal{F}_{\mathcal{P}}\psi$ in $\mathcal{P}$.

Processes / Procedures

3.3.2. Processes

Definition 3.2. A *process* $\mathcal{F}$ is a partially ordered multiset of localized events, i.e.

$$\mathcal{F} = \langle \mathcal{F}_{\mathbb{E}}, \mathcal{F}_{\mathcal{P}} \rangle : \mathbb{F} \to \mathbb{E} \times \mathcal{P}$$

where

- $(\mathbb{F}, \to)$ is a well-founded partial order, representing the structure time,
- $\mathbb{E}$ is a family of events, and
- $(\mathcal{P}, \subseteq)$ the partial order of configurations,

and they satisfy the requirements that

(a) if $\mathcal{F}_{\mathbb{E}}\phi$ is an action, then $©(\mathcal{F}_{\mathcal{P}}\phi)$ is well defined, and
(b) if $\phi \to \psi$ in $\mathbb{F}$ then $\mathcal{F}_{\mathcal{P}}\phi \subseteq \mathcal{F}_{\mathcal{P}}\psi$ or $\mathcal{F}_{\mathcal{P}}\phi \supseteq \mathcal{F}_{\mathcal{P}}\psi$ in $\mathcal{P}$.

Notation: The points in time are denoted by events.
(b) if $a[t]_P \to b[s]_Q$ then $P \subseteq Q$ or $P \supseteq Q$.

Flows, Runs, Procedures

Definition 3.4. For configurations $P, Q \in \mathcal{P}$, a *flow channel* $P \xrightarrow{\tau} Q$ can be either

- a channel $P \xrightarrow{\tau} Q$, or
- a flow channel $P \xrightarrow{\tau} Q'$, where $Q' \in Q$, or
- a flow channel $P' \xrightarrow{\tau} Q$, where $P' \in P$, or
- a flow channel $P' \xrightarrow{\tau} Q'$, where $P' \in P$ and $Q' \in Q$.

A *flow* $a[t]_P \xrightarrow{\tau} b[s]_Q$ is given by

- a flow channel $P \xrightarrow{\tau} Q$, and
- an interaction pair $a[t], b[s]$, i.e. a pair where
  - either $a[t] = \langle \cdot\, t \,\cdot \rangle$ and $b[s] = (\cdot\, s \,\cdot)$,
  - or $a[t] = \langle : t : \rangle$, and if $b[s] = (: s :)$.

A flow $a[t]_P \xrightarrow{\tau} b[s]_Q$ is *complete* if $s = t$.

Flows, Runs, Procedures

Definition 3.5. Let $\mathcal{F}$ be a process. A *run*, or *execution* $\mathcal{E}^{\mathcal{F}}$ of $\mathcal{F}$ is an assignment for each coaction $b[s]_Q$ of a unique flow $a[t]_P \xrightarrow{\tau} b[s]_Q$, which is required to be *sound*, in the sense that $b[s]_Q \not\to a[t]_P$ in $\mathcal{F}$.

A run is *complete* if all of the flows that it assigns are complete: the terms that are received are just those that were sent, and the inspections find just those terms that were submitted.

Flows, Runs, Procedures

Definition 3.5. Let $\mathcal{F}$ be a process. A *run*, or *execution* $\mathcal{E}^{\mathcal{F}}$ of $\mathcal{F}$ is an assignment for each coaction $b[s]_Q$ of a unique flow $a[t]_P \xrightarrow{\tau} b[s]_Q$, which is required to be *sound*, in the sense that $b[s]_Q \not\rightarrow a[t]_P$ in $\mathcal{F}$.

A run is *complete* if all of the flows that it assigns are complete: the terms that are received are just those that were sent, and the inspections find just those terms that were submitted.

Definition 3.6. A *network procedure* $\mathcal{L}$ is a pair $\mathcal{L} = \langle \mathcal{F}_{\mathcal{L}}, E_{\mathcal{L}} \rangle$ where

- $\mathcal{F}_{\mathcal{L}}$ is a process, and
- $E_{\mathcal{L}} = \{\mathcal{E}_1^{\mathcal{F}_{\mathcal{L}}}, \mathcal{E}_2^{\mathcal{F}_{\mathcal{L}}}, \mathcal{E}_3^{\mathcal{F}_{\mathcal{L}}} \ldots\}$ is a set of runs of $\mathcal{F}_{\mathcal{L}}$.

The elements of $E_{\mathcal{L}}$ are called *secure* runs. All other runs are *insecure*.

Flows, Runs, Procedures

Definition 3.5. Let $\mathcal{F}$ be a process. A *run*, or *execution* $\mathcal{E}^{\mathcal{F}}$ of $\mathcal{F}$ is an assignment for each coaction $b[s]_Q$ of a unique flow $a[t]_P \xrightarrow{\tau} b[s]_Q$, which is required to be *sound*, in the sense that $b[s]_Q \not\to a[t]_P$ in $\mathcal{F}$.

A run is *complete* if all of the flows that it assigns are complete: the terms that are received are just those that were sent, and the inspections find just those terms that were submitted.

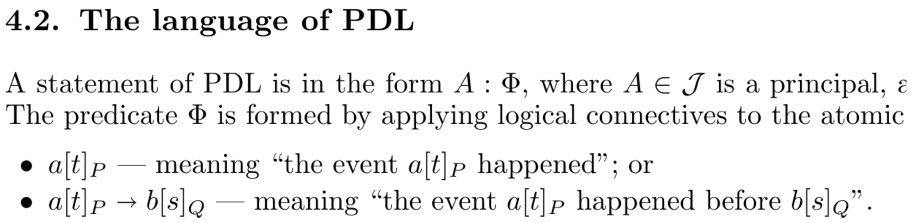
Definition 3.6. A *network procedure* $\mathcal{L}$ is a pair $\mathcal{L} = \langle \mathcal{F}_{\mathcal{L}}, E_{\mathcal{L}} \rangle$ where

- $\mathcal{F}_{\mathcal{L}}$ is a process, and
- $E_{\mathcal{L}} = \{\mathcal{E}_1^{\mathcal{F}_{\mathcal{L}}}, \mathcal{E}_2^{\mathcal{F}_{\mathcal{L}}}, \mathcal{E}_3^{\mathcal{F}_{\mathcal{L}}} \ldots\}$ is a set of runs of $\mathcal{F}_{\mathcal{L}}$.

The elements of $E_{\mathcal{L}}$ are called *secure* runs. All other runs are *insecure*.

- as $a[t]_P \to b[s]_Q$, saying that $a[t]_P$ precedes $b[s]_Q$ in the partial ordering $(\mathcal{F}, \to)$,
- $a[t]_P \xrightarrow{\tau} b[s]_Q$, denoting a flow of type τ from $a[t]_P$ to $b[s]_Q$ in a run $\mathcal{E}^{\mathcal{F}}$.

Procedure Description Logic

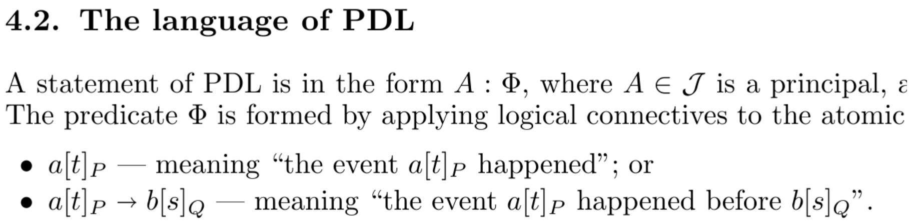


4.2. The language of PDL

A statement of PDL is in the form $A : \Phi$, where $A \in \mathcal{J}$ is a principal,
The predicate Φ is formed by applying logical connectives to the atomic

- $a[t]_P$ — meaning "the event $a[t]_P$ happened"; or
- $a[t]_P \rightarrow b[s]_Q$ — meaning "the event $a[t]_P$ happened before $b[s]_Q$".

Procedure Description Logic



4.2. The language of PDL

A statement of PDL is in the form $A : \Phi$, where $A \in \mathcal{J}$ is a principal,
The predicate Φ is formed by applying logical connectives to the atomic

- $a[t]_P$ — meaning "the event $a[t]_P$ happened"; or
- $a[t]_P \to b[s]_Q$ — meaning "the event $a[t]_P$ happened before $b[s]_Q$".

No clear syntax nor semantics.

But axioms are added and used for reasoning.

Procedure Description Logic – Axioms

Origination:

$$
\begin{aligned}
\text{©}P &: (\cdot\, t \,\cdot)_P \Longrightarrow \exists X.\ \langle \cdot\, t \,\cdot \rangle_X \rightarrow (\cdot\, t \,\cdot)_P \\
\text{©}P &: (:\, t \,:)_P \Longrightarrow \exists X.\ \langle :\, t \,: \rangle_X \rightarrow (:\, t \,:)_P
\end{aligned}
$$

Procedure Description Logic – Axioms

Origination:

$$\text{©}P \;:\; (\cdot\, t\, \cdot)_P \Longrightarrow \exists X.\ \langle \cdot\, t\, \cdot \rangle_X \rightarrow (\cdot\, t\, \cdot)_P$$
$$\text{©}P \;:\; (:t:)_P \Longrightarrow \exists X.\ \langle : t : \rangle_X \rightarrow (: t :)_P$$

Freshness:

- *freshly generated* by an action $\nu[x]$ before it is used anywhere; and
- that it can only be used elsewhere after it has passed in a message or a source.

which formally becomes

$$\begin{aligned}
\text{©}P :\quad & a[t.x]_P \Longrightarrow \exists X.\ \nu[x]_X \rightarrow a[t.x]_P \\
\text{©}P : \neg\nu[x]_P \wedge\ & a[t.x]_P \Longrightarrow \exists X.\ \left(\nu[x]_X \rightarrow \sqrt{\langle\langle \cdot\ x\ \cdot \rangle\rangle_X} \rightarrow ((\cdot\ x\ \cdot))_P \rightarrow a[t.x]_P\right) \\
& \qquad \vee\ \left(\nu[x]_X \rightarrow \sqrt{\langle\langle :\ x\ : \rangle\rangle_X} \rightarrow ((:\ x\ :))_P \rightarrow a[t.x]_P\right)
\end{aligned}$$

where, using the easy subterm order $\sqsubseteq$ from Sec. 3.2,

- $\langle\langle \cdot\ x\ \cdot \rangle\rangle_X$ abbreviates $\exists t.\ x \sqsubseteq t\ \wedge\ \langle \cdot\, t\, \cdot \rangle_X$,
- $((\cdot\ x\ \cdot))_X$ abbreviates $\exists t.\ x \sqsubseteq t\ \wedge\ (\cdot\, t\, \cdot)_X$, etc.

Procedure Description Logic – Axioms

4.4.2. Challenge-response authentication

The challenge-response axiom is in the form

$$©P \;:\; \mathsf{Local}_P \Longrightarrow \mathsf{Global}_{PQ}$$

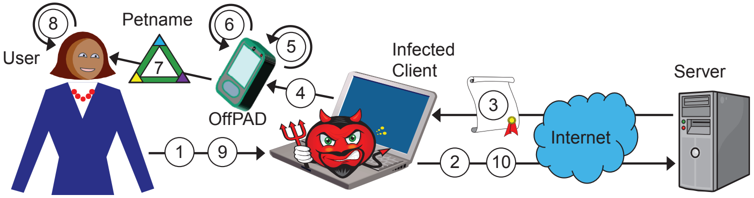
where, using the notation from Sec. 4.3.2

$$\begin{aligned}
\mathsf{Local}_P &= \nu[x]_P \to \left\langle \cdot\, c^{PQ}x \,\cdot \right\rangle_P \qquad\qquad \to \qquad\qquad \left(\cdot\, r^{PQ}x \,\cdot\right)_P \\
\mathsf{Global}_{PQ} &= \nu[x]_P \to \left\langle \cdot\, c^{PQ}x \,\cdot \right\rangle_P \to \left((\cdot\; c^{PQ}x\; \cdot)\right)_Q \to \sqrt{\langle\langle \cdot\; r^{PQ}x\; \cdot \rangle\rangle_Q} \to \left(\cdot\, r^{PQ}x \,\cdot\right)_P
\end{aligned}$$

Graphical Modeling of Security Ceremonies in OffPAD

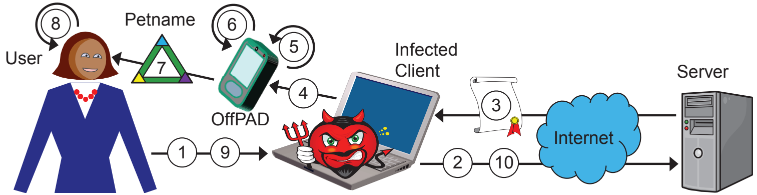
Graphical Modeling of Security Ceremonies in OffPAD

An initial ceremony for authenticating the server to the user:



Graphical Modeling of Security Ceremonies in OffPAD

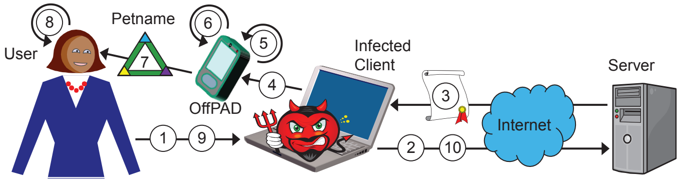
An initial ceremony for authenticating the server to the user:



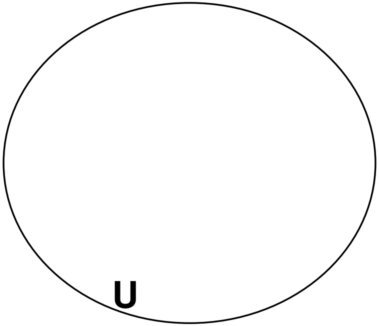
Nr.	Message / action description
1.	User initiates secure TLS connection through client platform
2.	Client platform contacts server
3:	Server returns server certificate containing public key
4.	Server certificate is forwarded to OffPAD
5.	Server certificate is validated (syntactic server authentication)
6.	Server certificate is mapped to petname
7.	Petname is presented to user
8.	User performs cognitive server authentication
9.	User approves server authentication
10.	TLS connection established between client and server

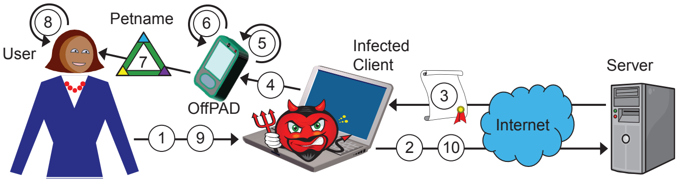
Table 3. Sequence of messages and actions for server authentication ceremony

ANP structure of the Ceremony

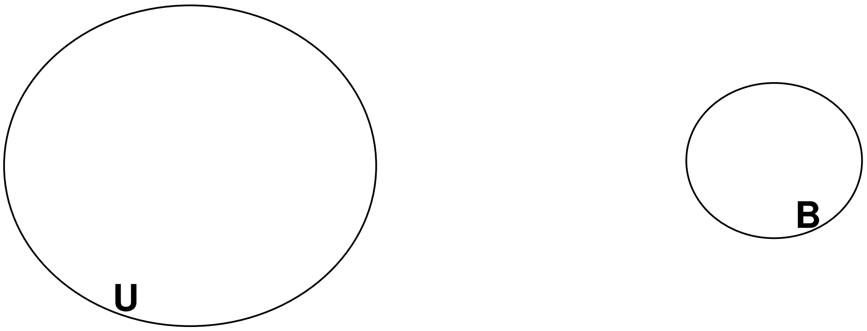


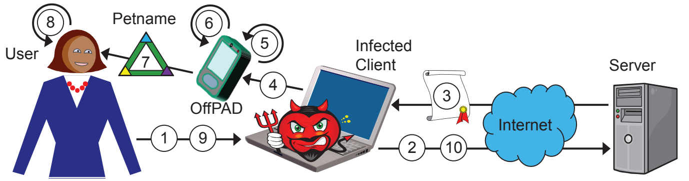
ANP structure of the Ceremony



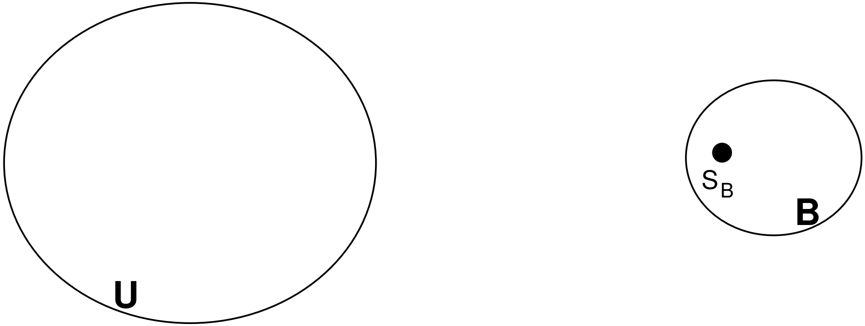


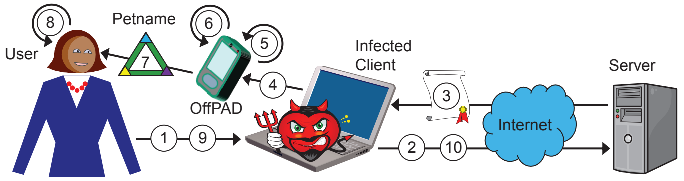
ANP structure of the Ceremony



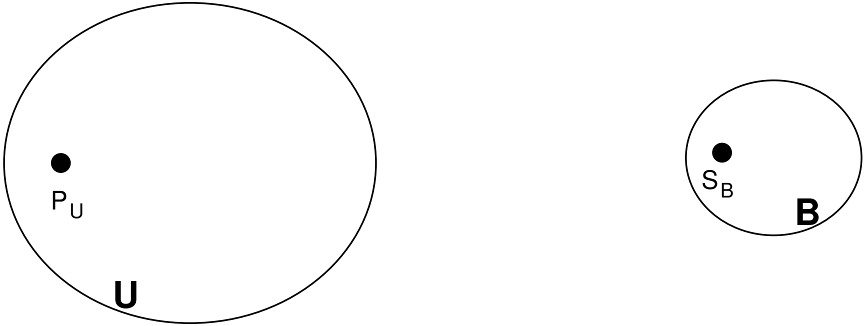


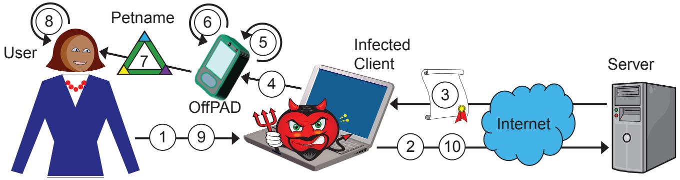
ANP structure of the Ceremony



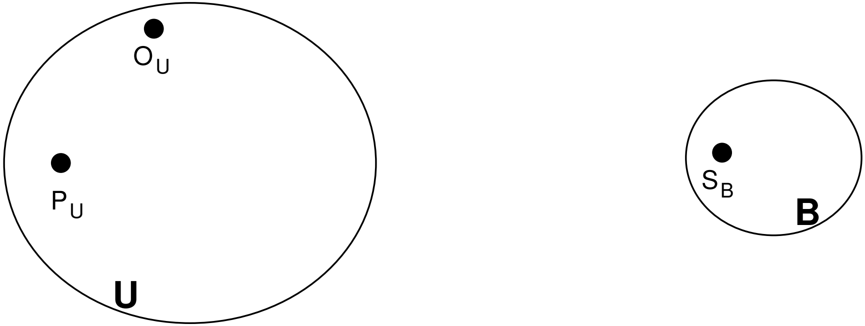


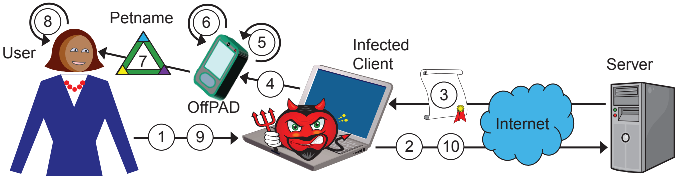
ANP structure of the Ceremony



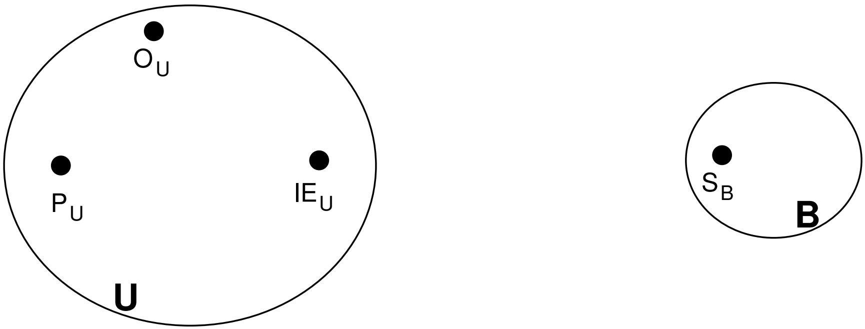


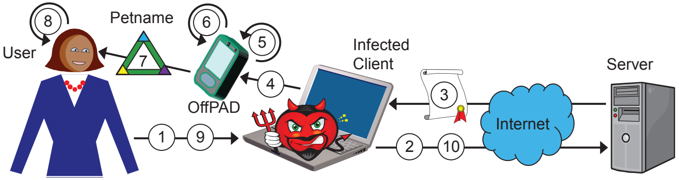
ANP structure of the Ceremony



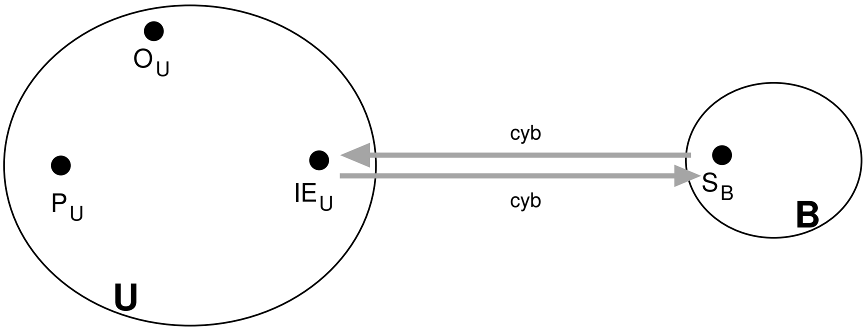


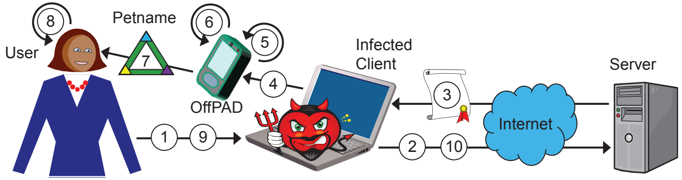
ANP structure of the Ceremony



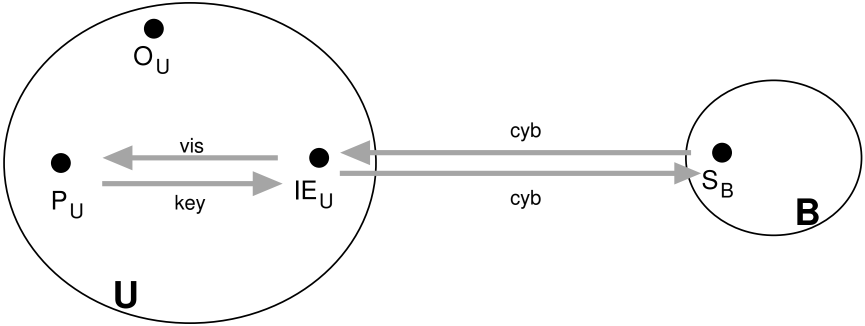


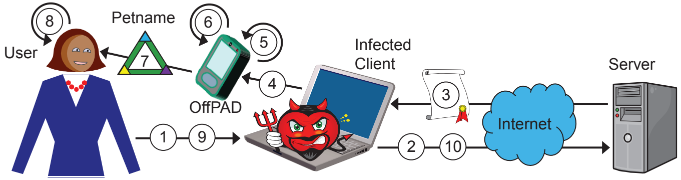
ANP structure of the Ceremony



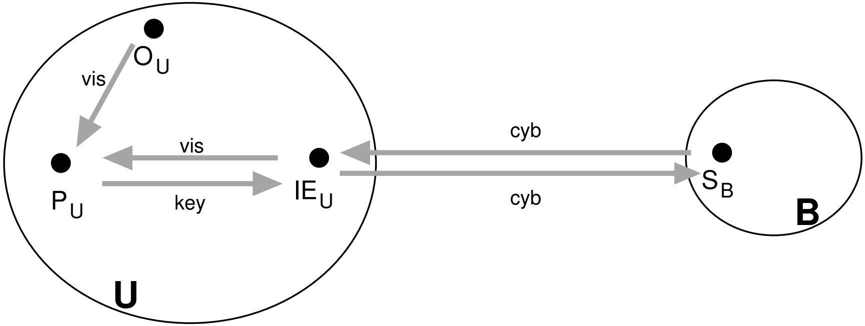


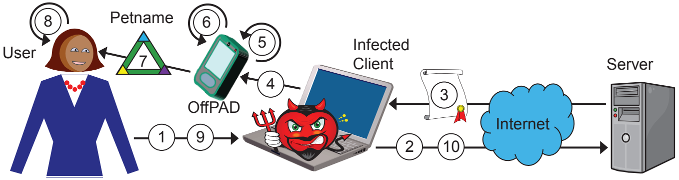
ANP structure of the Ceremony



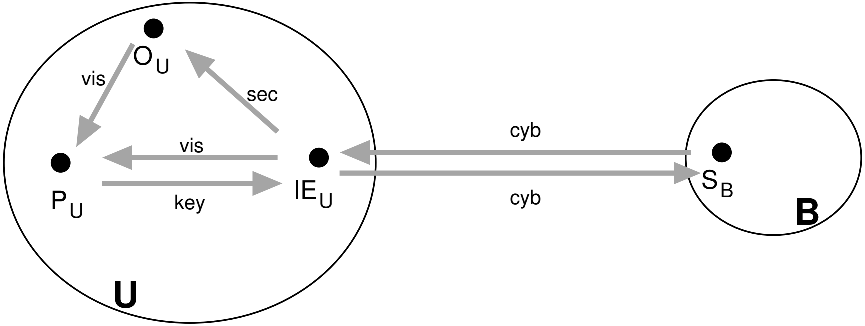


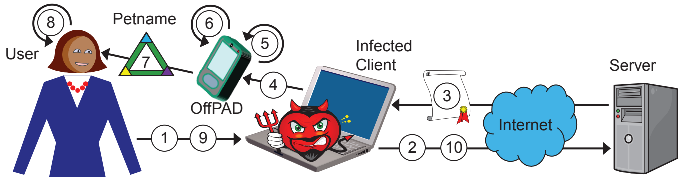
ANP structure of the Ceremony





ANP structure of the Ceremony

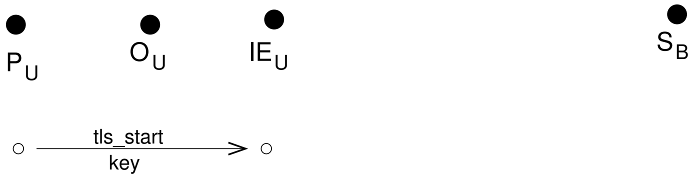




ANP run of the Ceremony

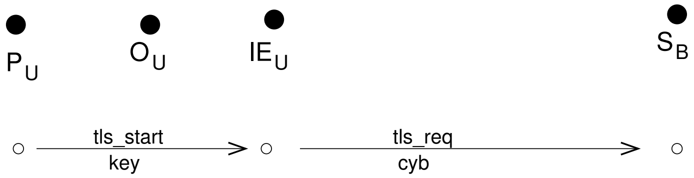
1. User initiates secure TLS connection through client platform
2. Client platform contacts server
3: Server returns server certificate containing public key
4. Server certificate is forwarded to OffPAD
5. Server certificate is validated (syntactic server authentication)

ANP run of the Ceremony



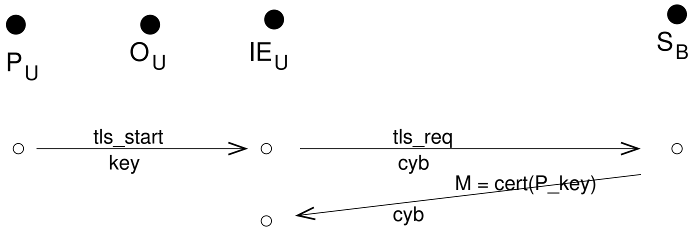
1. User initiates secure TLS connection through client platform
2. Client platform contacts server
3: Server returns server certificate containing public key
4. Server certificate is forwarded to OffPAD
5. Server certificate is validated (syntactic server authentication)

ANP run of the Ceremony



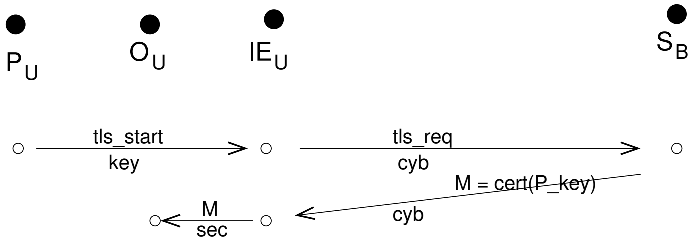
1. User initiates secure TLS connection through client platform
2. Client platform contacts server
3: Server returns server certificate containing public key
4. Server certificate is forwarded to OffPAD
5. Server certificate is validated (syntactic server authentication)

ANP run of the Ceremony



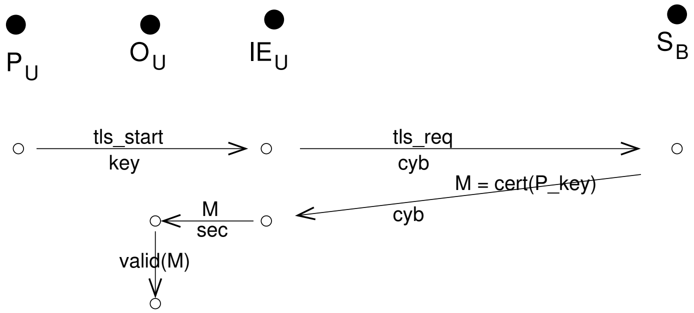
1. User initiates secure TLS connection through client platform
2. Client platform contacts server
3: Server returns server certificate containing public key
4. Server certificate is forwarded to OffPAD
5. Server certificate is validated (syntactic server authentication)

ANP run of the Ceremony



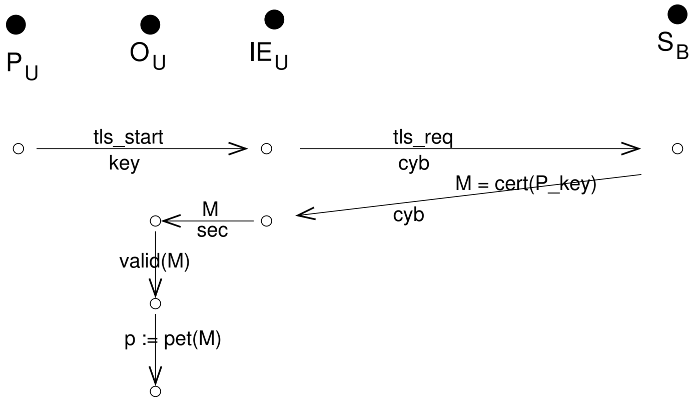
1. User initiates secure TLS connection through client platform
2. Client platform contacts server
3: Server returns server certificate containing public key
4. Server certificate is forwarded to OffPAD
5. Server certificate is validated (syntactic server authentication)

ANP run of the Ceremony



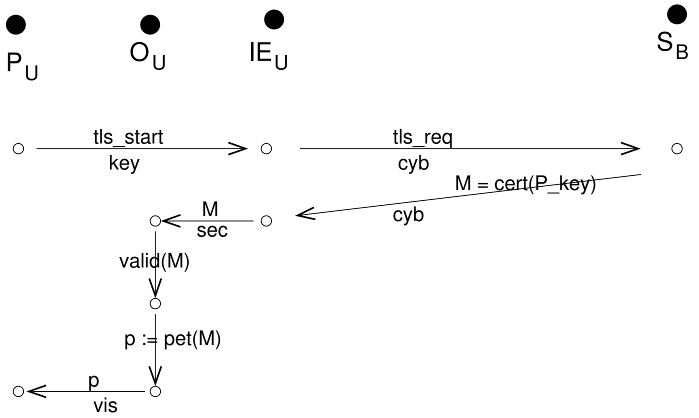
1. User initiates secure TLS connection through client platform
2. Client platform contacts server
3: Server returns server certificate containing public key
4. Server certificate is forwarded to OffPAD
5. Server certificate is validated (syntactic server authentication)

ANP run of the Ceremony



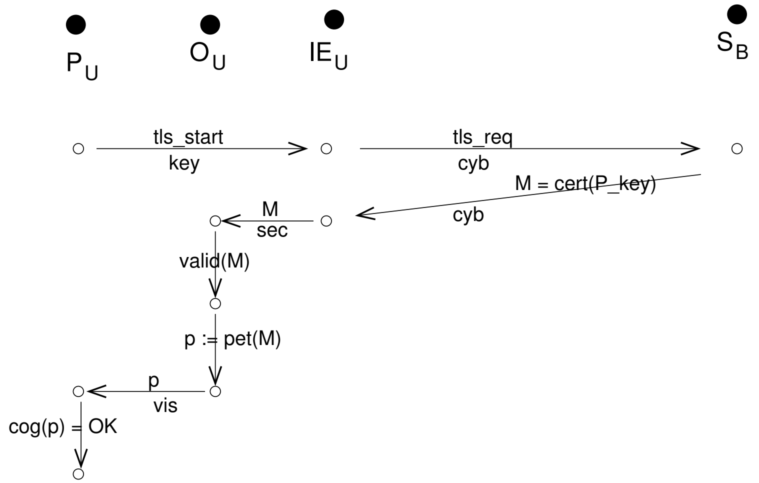
6. Server certificate is mapped to petname
7. Petname is presented to user
8. User performs cognitive server authentication
9. User approves server authentication
10. TLS connection established between client and server

ANP run of the Ceremony



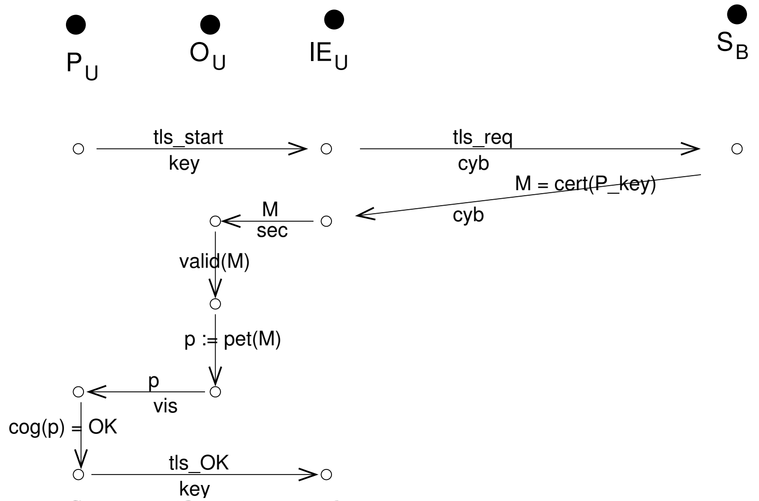
6. Server certificate is mapped to petname
7. Petname is presented to user
8. User performs cognitive server authentication
9. User approves server authentication
10. TLS connection established between client and server

ANP run of the Ceremony



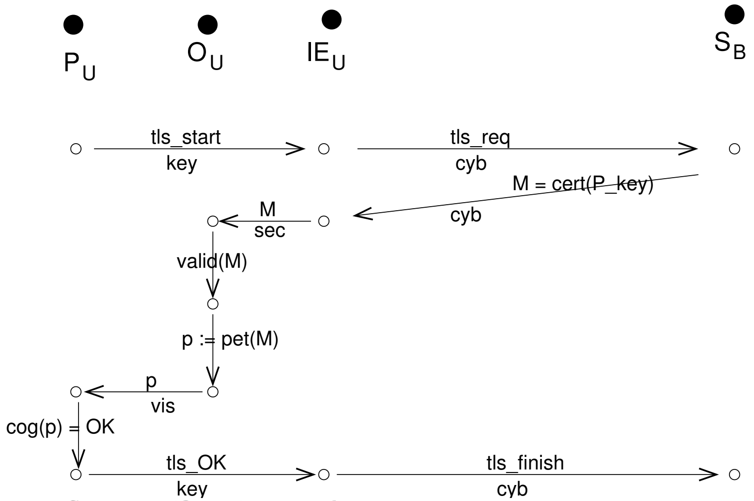
6. Server certificate is mapped to petname
7. Petname is presented to user
8. User performs cognitive server authentication
9. User approves server authentication
10. TLS connection established between client and server

ANP run of the Ceremony



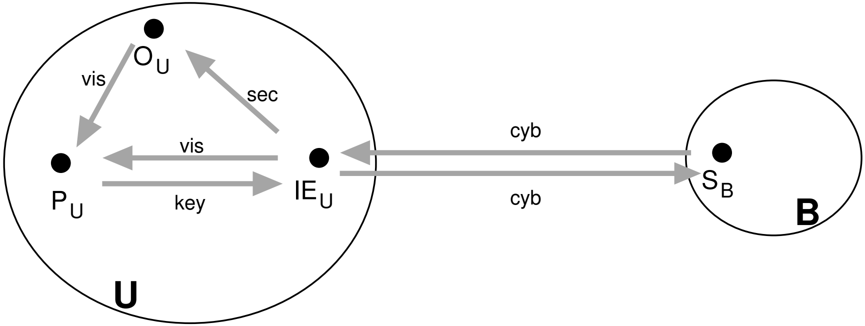
6. Server certificate is mapped to petname
7. Petname is presented to user
8. User performs cognitive server authentication
9. User approves server authentication
10. TLS connection established between client and server

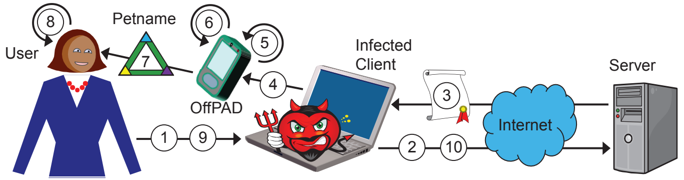
ANP run of the Ceremony



6. Server certificate is mapped to petname
7. Petname is presented to user
8. User performs cognitive server authentication
9. User approves server authentication
10. TLS connection established between client and server

Refining the Ceremony





eurostars™