Trust and Concurrency
A Theory Contribution to Ubiquitous Computing

Vladimiro Sassone

University of Sussex, UK

UbiNet Summer School 2004

Edinburgh 14.09.04

What is this about?

- **Theory? What Theory?**

What is this about?

- **Theory? What Theory?**
  - Not a topic, a research approach, but an investigation method...
- **(Ubiquitous) Computing is Ubiquitous**
  - So, what to present?

What is this about?

- **Theory? What Theory?**
  - Not a topic, a research approach, but an investigation method...
- **(Ubiquitous) Computing is Ubiquitous**
  - So, what to present?
  - Models of Computation
  - Foundation Calculi $((\lambda, \text{CCS}, \pi, \ldots))$
  - Programming (Language) Principles
  - Development Methodologies
  - Security
  - Analysis and Verification
  - ...

What is this about?

- **Theory? What Theory?**
  - Not a topic, a research approach, but an investigation method...
- **(Ubiquitous) Computing is Ubiquitous**
  - So, what to present?
  - Models of Computation
  - Foundation Calculi $((\lambda, \text{CCS}, \pi, \ldots))$
  - Programming (Language) Principles
  - Development Methodologies
  - Security
  - Analysis and Verification
  - Trust
- **Why Trust?**
  - Some specific reasons, no overwhelming one
  - My next research interest

What is this about?

- **Theory? What Theory?**
  - Not a topic, a research approach, but an investigation method...
- **(Ubiquitous) Computing is Ubiquitous**
  - So, what to present?
  - Models of Computation
  - Foundation Calculi $((\lambda, \text{CCS}, \pi, \ldots))$
  - Programming (Language) Principles
  - Development Methodologies
  - Security
  - Analysis and Verification
  - Trust
- **Why Trust?**
  - Some specific reasons, no overwhelming one
  - My next research interest
- **Real Agenda?** **Theory** and **Practice** increasingly need each other...

What is this about?

- **Theory? What Theory?**
  - Not a topic, a research approach, but an investigation method...
- **(Ubiquitous) Computing is Ubiquitous**
  - So, what to present?
  - Models of Computation
  - Foundation Calculi $((\lambda, \text{CCS}, \pi, \ldots))$
  - Programming (Language) Principles
  - Development Methodologies
  - Security
  - Analysis and Verification
  - Trust
- **Why Trust?**
  - Some specific reasons, no overwhelming one
  - My next research interest
- **Real Agenda?** Good **Theory** and Good **Practice** increasingly need each other...
  - Interest you in looking for connections

1. Motivation & Goals
   - Understanding trust based systems
   - Guidelines for their designers and implementers
   - Techniques for reasoning about their properties

2. Formal Models for Trust
   - A simple policy language
   - Trust semantic domains
   - Computing trust values

3. Observing Events
   - Event structures

4. A Calculus of Trust
   - Equational theory

Joint work with Mogens Nielsen, Karl Krukow, Marco Carbone,. . .

Global Ubiquitous Computing

- Billions of autonomous mobile networked entities
  - Mobile users
  - Mobile software agents
  - Mobile networked devices:

Mobile communication devices (phones, pagers, . . .)
Mobile computing devices (laptops, palmtops, . . .)
Commodity products (embedded devices)

Global Ubiquitous Computing

- Billions of autonomous mobile networked entities
  - Mobile users
  - Mobile software agents
  - Mobile networked devices:

    Mobile communication devices (phones, pagers, . . .)
    Mobile computing devices (laptops, palmtops, . . .)
    Commodity products (embedded devices)

- Entities collaborate with each other:
  - Resource sharing: Ad hoc networks, computational grids, . . .
  - Information sharing: Collaborative applications, recommendation systems, . . .

New Security Challenges

- Security scenario for global computing environment
  - Large number of autonomous entities
  - Large number of administrative domains
  - No common trusted computing base
  - No global system trust
  - Virtual anonymity

New Security Challenges

- Security scenario for global computing environment
  - Large number of autonomous entities
  - Large number of administrative domains
  - No common trusted computing base
  - No global system trust
  - Virtual anonymity
- Such requirements exclude the use of current security mechanisms used in large distributed systems

New Security Challenges

- Security scenario for global computing environment
  - Large number of autonomous entities
  - Large number of administrative domains
  - No common trusted computing base
  - No global system trust
  - Virtual anonymity
- Such requirements exclude the use of current security mechanisms used in large distributed systems
- An alternative approach: **Trust based security**

Theory meets Practice

Theory's success stories:

- Verification (relates to type checking/inference)
- Certificates (groups as certified roles)
- ...

All this only works as long as you trust the certified types, or are willing and ble to check and monitor migrating agents yourself (e.g., bytecode verification, PCC, ...).

> ...one cannot "verify" the Internet

The gap between theory and practice matters in practice.

- Trust: In UbiComp, security must work be coupled with trust management.

Which is hard, because of delegation and dynamic policies

Trust and Trust Management

Trust: What is it?

- Think of the usual human-like notion...

Trust and Trust Management

Trust: What is it?

- Think of the usual human-like notion...
- ...but on a *global computing* scale.

Trust and Trust Management

Trust: What is it?

- Think of the usual human-like notion...
- ...but on a *global computing* scale.

Trust Management: Fundamental aspects?

1. Trust is gathered by individuals e.g. from personal experiences;
2. Trust is shared by communities, e.g. to form "reputation systems";

Trust and Trust Management

Trust: What is it?

- Think of the usual human-like notion...
- ...but on a *global computing* scale.

Trust Management: Fundamental aspects?

1. Trust is gathered by individuals e.g. from personal experiences;
2. Trust is shared by communities, e.g. to form "reputation systems";

Which means:

- Principals act according to "policies" upon consulting "trust tables," and "update" these constantly according to the outcome of transactions.

Collaboration Model

- Trust formation
  - Personal experience
  - Recommendation from known (trusted) third parties
  - Reputation (recommendation from many strangers)
  - External events (help build reputation)

Collaboration Model

- Trust formation
  - Personal experience
  - Recommendation from known (trusted) third parties
  - Reputation (recommendation from many strangers)
  - External events (help build reputation)
- Trust evolution
  - Incorporating new trust formation data
  - Expiration of old trust values
  - As a function of time
  - As a reaction to betrayal

Collaboration Model

- Trust formation
  - Personal experience
  - Recommendation from known (trusted) third parties
  - Reputation (recommendation from many strangers)
  - External events (help build reputation)
- Trust evolution
  - Incorporating new trust formation data
  - Expiration of old trust values
  - As a function of time
  - As a reaction to betrayal
- Trust exploitation
  - Feedback based on experience

Some applications

- A peer to peer distributed file system
- A telephone-based micro-payment system
- An agent controlled information portal
- A distributed SPAM filter
- A smart space environment
- Collaborative PDA environment

Some applications

- A peer to peer distributed file system
- A telephone-based micro-payment system
- An agent controlled information portal
- A distributed SPAM filter (Secure Project)
- A smart space environment
- Collaborative PDA environment

Trust-Based Security Decisions

- Security-related decisions:
  - Passive: e.g.: should I allow principal P to access resource r?
  - Active: e.g.: which of principals P, Q, R will provide the best service for me?

Trust-Based Security Decisions

- Security-related decisions:
  - Passive: e.g.: should I allow principal P to access resource r?
  - Active: e.g.: which of principals P, Q, R will provide the best service for me?
- Trust-based decisions:
  - Decisions based on principals' behaviour, reputation, ...
  - Principals collaborate: recommendations, ...
  - Principals are networked, decisions made autonomously.
  - Decisions made based on partial information.

Entity Recognition

Background questions

- How to recognize other entities
  - Digital signature
  - Behavioural patterns, time and place
  - Combinations of above

Entity Recognition

Background questions

- How to recognize other entities
  - Digital signature
  - Behavioural patterns, time and place
  - Combinations of above
- Limit risk of masquerading

Entity Recognition

Background questions

- How to recognize other entities
  - Digital signature
  - Behavioural patterns, time and place
  - Combinations of above
- Limit risk of masquerading
- Limit risk from change of identity
  - Slowly build trust – repeated changes of identity must result in very few privileges

Trust in the Social Sciences

Trust

D.H. McKnight, N.L. Chervany:

The Meaning of Trust

Trust in Cyber-Societies,
Springer LNAI 2246, pp. 27–54, 2001

McKnight-Chervany's Classification

Trust

- Disposition
- Structure/Situation
- Affect/Attitude
- Belief/Expectancy
- Intention
- Behaviour

Trustree

- Competence
- Benevolence
- Integrity
- Predictability
- Openness, carefulness, ...
- People, Institutions, ...

McKnight-Chervany's Classification

Trust

- Disposition
- Structure/Situation
- Affect/Attitude
- Belief/Expectancy
- Intention
- Behaviour

Trustree

- Competence
- Benevolence
- Integrity
- Predictability
- Openness, carefulness, ...
- People, Institutions, ...

Plus some computing specific: time-dependent, rapidly-varying, ...

An Ubiquitous Notion Itself

"Trust is a term with many meanings."

— Oliver Williamson (Economist, Berkeley)

"Trust is itself a term for a clustering of meanings."

— Harrison White (Sociologist, Columbia)

"Researchers' purposes may be better served if they focus on specific components of trust rather than the generalised case."

— Robert Kaplan (Leadership Devel, Harvard)

An Ubiquitous Notion Itself

"Trust is a term with many meanings."

— Oliver Williamson (Economist, Berkeley)

"Trust is itself a term for a clustering of meanings."

— Harrison White (Sociologist, Columbia)

"Researchers' purposes may be better served if they focus on specific components of trust rather than the generalised case."

— Robert Kaplan (Leadership Devel, Harvard)

An Ubiquitous Notion Itself

"Trust is a term with many meanings."

— Oliver Williamson (Economist, Berkeley)

"Trust is itself a term for a clustering of meanings."

— Harrison White (Sociologist, Columbia)

"Researchers' purposes may be better served if they focus on specific components of trust rather than the generalised case."

— Robert Kaplan (Leadership Devel, Harvard)

An Ubiquitous Notion Itself

"Trust is a term with many meanings."

— Oliver Williamson (Economist, Berkeley)

"Trust is itself a term for a clustering of meanings."

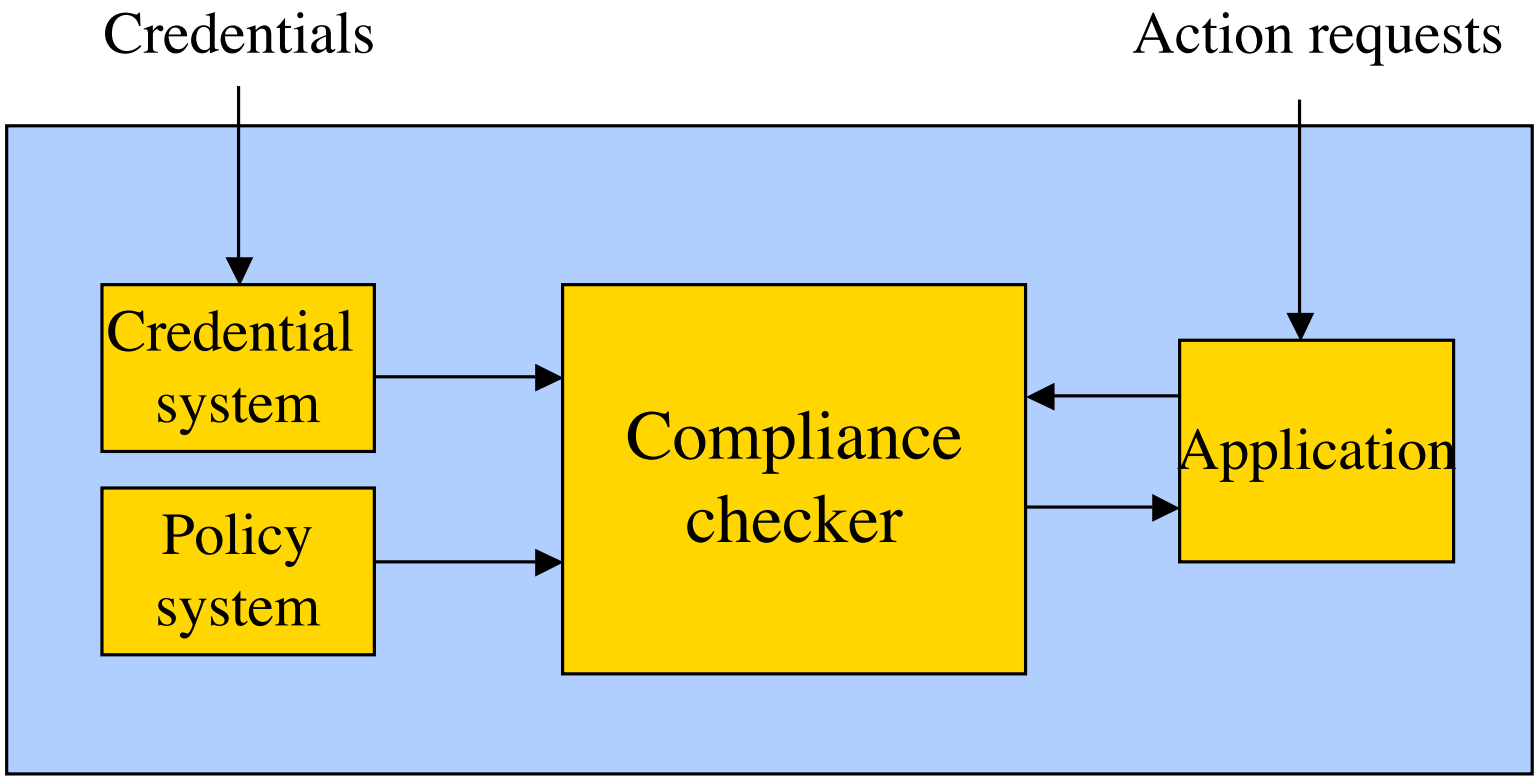
— Harrison White (Sociologist, Columbia)

"Researchers' purposes may be better served if they focus on specific components of trust rather than the generalised case."

— Robert Kaplan (Leadership Devel, Harvard)

Part 1 – Modelling Trust

Trust Management – Blaze et al



Elements of Trust Management

- Language for Actions
- Naming scheme for Principals
- Language for Trust-Policies
- Language for Credentials

- Compliance checker and interface

Towards a Formal Model

Goals: Formal understanding of trust based systems

Stephen Weeks

Understanding Trust Management Systems

IEEE Symposium on Security and Privacy 2001

Week's Model

Scenario with

- A set $\mathcal{P}$ of principals (ranged over by a,b,c)
- A set $\mathcal{T}$ of trust values

Trust information of a system represented by:

$$\texttt{trust-state} : \mathcal{P} \to \mathcal{P} \to \mathcal{T},$$

where $\texttt{trust-state}(a)(b)$ represents a's trust in b.

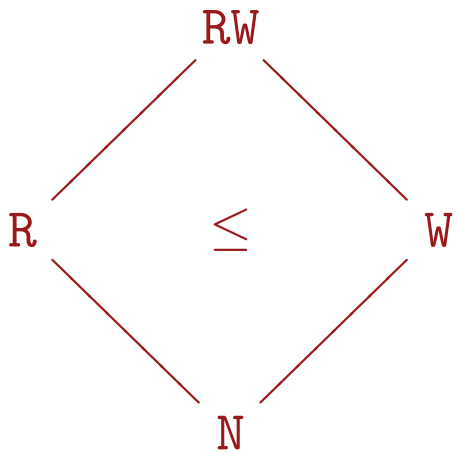
Weeks argues for and assumes that $\mathcal{T}$ is equipped with an partial ordering $\leq$ such that $(\mathcal{T}, \leq)$ is a (complete) lattice.

Aside I: Complete lattices

A Complete Lattice is an ordered set such that each subset of elements has:

- a "least upper bound" (`lub`); and
- a "greatest lower bound" (`glb`).

Let $T = \{\mathtt{N}, \mathtt{R}, \mathtt{W}, \mathtt{RW}\}$.



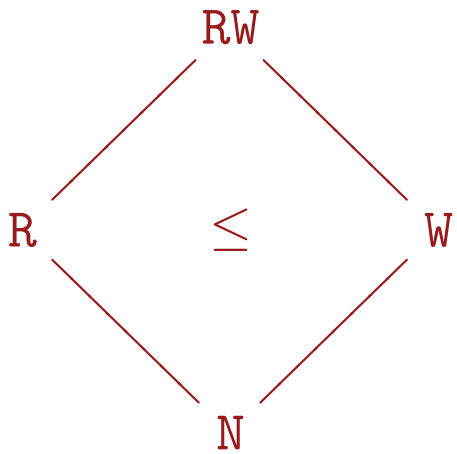
This formalises that given two trust "levels," there always exist a trust "level" which sums them and one which intersects them.

Aside I: Complete lattices

A Complete Lattice is an ordered set such that each subset of elements has:

- a "least upper bound" (`lub`); and
- a "greatest lower bound" (`glb`).

Let $T = \{\mathtt{N}, \mathtt{R}, \mathtt{W}, \mathtt{RW}\}$.



This formalises that given two trust "levels," there always exist a trust "level" which sums them and one which intersects them.

Do you agree?

Modelling Trust

Each principal specifies a policy. This is a local contribution to the global trust, and depends on other principals' policies.

Central notion: **Delegation**

$$\text{“}a\text{'s policy”} = \ldots \text{“ask } b\text{”} \ldots \triangleq \ldots \ulcorner b \urcorner \ldots$$

Modelling Trust

Each principal specifies a policy. This is a local contribution to the global trust, and depends on other principals' policies.

Central notion: **Delegation**

$$\text{“}a\text{'s policy”} = \ldots \text{“ask } b\text{”} \ldots \triangleq \ldots \ulcorner b \urcorner \ldots$$

This gives principals a with policies π_a, where:

$$\pi_a : (\mathcal{P} \to \mathcal{P} \to \mathcal{T}) \to \mathcal{P} \to \mathcal{T}$$

The collection of π_a's induces a global policy bundle:

$$\Pi : (\mathcal{P} \to \mathcal{P} \to \mathcal{T}) \longrightarrow (\mathcal{P} \to \mathcal{P} \to \mathcal{T})$$

In fact, $\Pi = \{\pi_a, \pi_b, \pi_c, \ldots\} \triangleq \lambda x.\pi_x$.

A Few Examples

$\lambda x.x = c \mapsto \mathtt{W}; \ldots$ *abstraction*

$\lambda x.(\ulcorner b \urcorner x \vee \mathtt{R})$ *referencing*

$a:\ \lambda x.(\ulcorner a \urcorner b \wedge \ulcorner b \urcorner x) \vee R$ *discounting*

$a:\ \lambda x.(\ldots \ulcorner b \urcorner x \ldots)$
$b:\ \lambda x.(\ldots \ulcorner a \urcorner x \ldots)$ *cyclic delegation*

A Simple Policy Language

Policies and Expressions:

$$
\begin{array}{llll}
\pi ::= & \ulcorner p \urcorner & \text{(delegation)} & \\
& \lambda x : \mathrm{P}.\tau & \text{(abstraction)} & \\
& \mathrm{op}(\pi_1, \ldots, \pi_n) & \text{(lattice op)} & \\
p ::= & a \in \mathcal{P} & \text{(principal)} & \\
& x : \mathrm{P} & \text{(vars)} &
\end{array}
\qquad
\begin{array}{llll}
\tau ::= & t \in D & \text{(value)} & \\
& \pi(p) & \text{(policy value)} & \\
& e \mapsto \tau;\tau & \text{(choice)} & \\
e ::= & \tau \ \mathrm{cmp} \ \tau & \text{(comparison)} & \\
& p \ \mathrm{eq} \ p & \text{(comparison)} & \\
& e \ \mathrm{bop} \ e & \text{(boolean op)} &
\end{array}
$$

. . . and its Semantics

Correspondence of language expressions and policy functions

$$[\![-]\!]_{\sigma m} : (\tau \to \mathcal{T}) + (p \to \mathcal{P}) + (\pi \to \mathcal{P} \to \mathcal{T}) + (e \to \mathrm{Bool})$$

$$\begin{aligned}
[\![c]\!]_{\sigma m} &= c \\
[\![x]\!]_{\sigma m} &= \sigma(x) \\
[\![\pi(p)]\!]_{\sigma m} &= [\![\pi]\!]_{\sigma m}([\![p]\!]_{\sigma m}) \\
[\![e \mapsto \tau_0;\tau_1]\!]_{\sigma m} &= \text{if } [\![e]\!]_{\sigma m} \text{ then } [\![\tau_0]\!]_{\sigma m} \text{ else } [\![\tau_1]\!]_{\sigma m} \\
[\![\ulcorner p \urcorner]\!]_{\sigma m} &= m([\![p]\!]_{\sigma m}) \\
[\![\lambda x.\tau]\!]_{\sigma m} &= \lambda p : \mathcal{P}.[\![\tau]\!]_{\sigma\{x:=p\}m} \\
[\![\mathrm{op}(\pi_1 \ldots \pi_n)]\!]_{\sigma m} &= \mathrm{op} \circ \langle [\![\pi_1]\!]_{\sigma m} \ldots [\![\pi_n]\!]_{\sigma m} \rangle
\end{aligned}$$

$$\sigma : \mathrm{Vars} \to \mathcal{P} \qquad\qquad m : \mathcal{P} \to \mathcal{P} \to \mathcal{T}$$

The Definition of Global Trust

Assume $\mathcal{T}$ is a complete lattice, given a monotone policy bundle

$$\Pi : (\mathcal{P} \to \mathcal{P} \to \mathcal{T}) \to (\mathcal{P} \to \mathcal{P} \to \mathcal{T})$$

Global Trust is defined as the least fixed point of Π.

$$\mathit{fix}(\Pi) : \mathcal{P} \to \mathcal{P} \to \mathcal{T}.$$

The Definition of Global Trust

Assume $\mathcal{T}$ is a complete lattice, given a monotone policy bundle

$$\Pi : (\mathcal{P} \to \mathcal{P} \to \mathcal{T}) \to (\mathcal{P} \to \mathcal{P} \to \mathcal{T})$$

Global Trust is defined as the least fixed point of Π.

$$\mathit{fix}(\Pi) : \mathcal{P} \to \mathcal{P} \to \mathcal{T}.$$

Fixed point

A fixed point of $F : D \to D$ is $d \in D$ s.t. $F(d) = d$.

If D is a ordered, the least fixed point (`lfp`) is the least such d.

If D is a complete lattice and F is monotonic, the `lfp` always exists.

The Definition of Global Trust

Assume $\mathcal{T}$ is a complete lattice, given a monotone policy bundle

$$\Pi : (\mathcal{P} \to \mathcal{P} \to \mathcal{T}) \to (\mathcal{P} \to \mathcal{P} \to \mathcal{T})$$

Global Trust is defined as the least fixed point of Π.

$$\mathit{fix}(\Pi) : \mathcal{P} \to \mathcal{P} \to \mathcal{T}.$$

Fixed point

A fixed point of $F : D \to D$ is $d \in D$ s.t. $F(d) = d$.

If D is a ordered, the least fixed point (`lfp`) is the least such d.

If D is a complete lattice and F is monotonic, the `lfp` always exists.

"This is the first time they all agree on the trust distribution."

Understanding delegation

Example:

$$
\begin{array}{lll}
a: & p \mapsto \text{trusted}; & \\
& q \mapsto \ulcorner b \urcorner(q); & \\
& z \mapsto \ulcorner p \urcorner(z); &
\end{array}
\qquad
\begin{array}{lll}
b: & p \mapsto \ulcorner a \urcorner(p); & \\
& q \mapsto \text{untrusted}; & \\
& z \mapsto \ulcorner a \urcorner(z); &
\end{array}
$$

Understanding delegation

Example:

$$
\begin{array}{lll}
a: & p \mapsto \text{trusted}; & \\
& q \mapsto \ulcorner b \urcorner(q); & \\
& z \mapsto \ulcorner p \urcorner(z); &
\end{array}
\qquad
\begin{array}{lll}
b: & p \mapsto \ulcorner a \urcorner(p); & \\
& q \mapsto \text{untrusted}; & \\
& z \mapsto \ulcorner a \urcorner(z); &
\end{array}
$$

Delegation, formally: Global trust as a fixpoint. (Weeks)

$$\pi : (\mathcal{P} \to \mathcal{P} \to \mathcal{T}) \to (\mathcal{P} \to \mathcal{T}) \qquad \text{Local Policy}$$

$$\Pi : (\mathcal{P} \to \mathcal{P} \to \mathcal{T}) \to (\mathcal{P} \to \mathcal{P} \to \mathcal{T}) \qquad \text{Collected Policies}$$

Global Trust: $fix(\Pi) : \mathcal{P} \to \mathcal{P} \to \mathcal{T}$. But, is this good enough?

Understanding delegation

Example:

$a:$	$p \mapsto \text{trusted};$	$b:$	$p \mapsto \ulcorner a \urcorner(p);$
	$q \mapsto \ulcorner b \urcorner(q);$		$q \mapsto \text{untrusted};$
	$z \mapsto \ulcorner p \urcorner(z);$		$z \mapsto \ulcorner a \urcorner(z);$

Delegation, formally: Global trust as a fixpoint. (Weeks)

$$\pi : (\mathcal{P} \to \mathcal{P} \to \mathcal{T}) \to (\mathcal{P} \to \mathcal{T}) \qquad \text{Local Policy}$$

$$\Pi : (\mathcal{P} \to \mathcal{P} \to \mathcal{T}) \to (\mathcal{P} \to \mathcal{P} \to \mathcal{T}) \qquad \text{Collected Policies}$$

Global Trust: $fix(\Pi) : \mathcal{P} \to \mathcal{P} \to \mathcal{T}$. But, is this good enough?

p : trusted q : untrusted z : ???

Cannot confuse don't trust with don't know: the value of $\ulcorner p \urcorner(z)$ could become available later.

Need to account for uncertain knowledge of $\ulcorner p \urcorner(z) \in \mathcal{T}$.

Trust Structures

$$(D, \preceq, \sqsubseteq), \quad \text{where } \bigvee \text{ is } \sqsubseteq\text{-continous}$$

where $(D, \preceq)$ is a trust lattice and $(D, \sqsubseteq)$ is an approximation lattice.

Trust Structures

$$(D, \preceq, \sqsubseteq), \quad \text{where } \bigvee \text{ is } \sqsubseteq\text{-continous}$$

where $(D, \preceq)$ is a trust lattice and $(D, \sqsubseteq)$ is an approximation lattice.

Theorem

- $(\mathcal{T}, \preceq, \sqsubseteq)$ yields an adequate model $[\![-]\!]_\sigma : \text{Policies} \to (\mathcal{P} \to \mathcal{P} \to D)$.
- For any collection Π of monotonic policies there is a unique global trust state, given by $\text{gts} \triangleq \mathit{fix}_{\sqsubseteq}(\Pi) : \mathcal{P} \to \mathcal{P} \to D$.
- The fixpoint is computed with respect to $\sqsubseteq$.
- The framework supports the specification of imprecise or uncertain trust values.
- Trust structure can be derived canonically from lattices $(D, \leq)$.

A Constructive Approach to Trust Structures

Let $(D, \leq)$ be a structure of 'trust values' without uncertainty.

$$I(D) = \{[d_0, d_1] \mid d_0, d_1 \in D, d_0 \leq d_1\}$$

Consider now the orderings $\preceq$ and $\sqsubseteq$ on $I(D)$ defined as:

$$[d_0, d_1] \preceq [d'_0, d'_1] \quad \text{iff} \quad d_0 \leq d'_0 \text{ and } d_1 \leq d'_1$$

$$[d_0, d_1] \sqsubseteq [d'_0, d'_1] \quad \text{iff} \quad d_0 \leq d'_0 \text{ and } d'_1 \leq d_1$$

$\preceq$ is the trust ordering used in decision making
$\sqsubseteq$ is the information ordering used in lfp-semantics

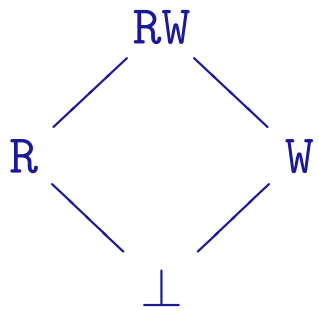
Theorem

For any complete lattice $(D, \leq)$

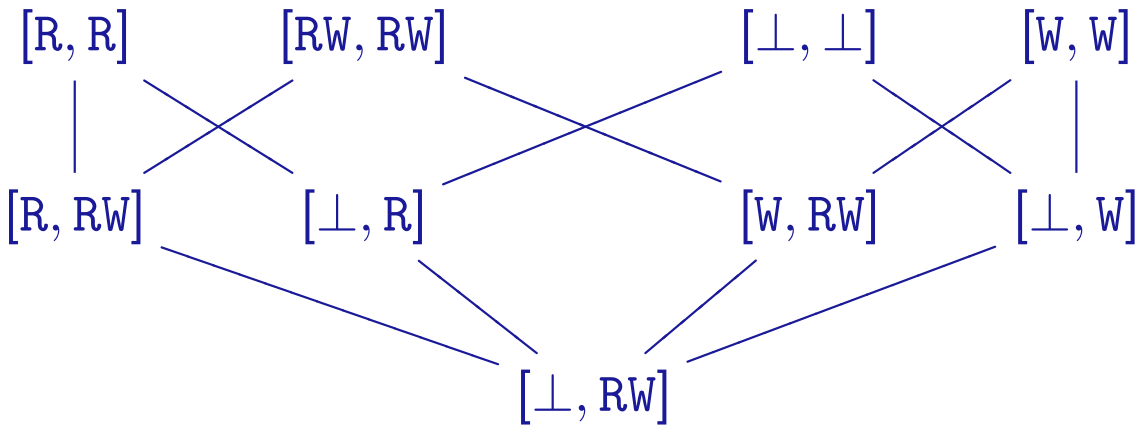
- $(I(D), \preceq)$ is a complete lattice
- $(I(D), \sqsubseteq)$ is a complete lattice

Exemplifying Trust Structures

- Access-rights often form a complete lattice.



- Intervals introduces uncertainty in a canonical way.



Properties of Intervals

Theorem

Given a complete lattice $(D, \leq)$ and a continuous function $f : D_n \to D$, then the pointwise extension F of f is continuous in $(I(D), \preceq)$ and $(I(D), \sqsubseteq)$

- Example: addition and multiplication on the reals
- Example: glb and lub on $(D, \leq)$

Theorem

Given complete lattices D and D', then $I(D \times D)$ is isomorphic to $I(D) \times I(D')$ and $I(A \to D)$ is isomorphic to $A \to I(D)$. with respect to both orderings

Operational Aspects

- Trust structures $TS = (D, \preceq, \sqsubseteq)$ give a framework for denotational semantics for collections of mutually referring trust policies.
- No good if principals are unable to reason about their own trust in others.
- $p \in \mathcal{P}$ wants to compute $fix_{\sqsubseteq}\Pi_\lambda(p) : \mathcal{P} \to D$
  - Problem: function Π_λ is distributed as π_q, for $q \in \mathcal{P}$.
  - Problem: in principle $fix_{\sqsubseteq}(\Pi_\lambda)(p)$ depends on π_q for all $q \in \mathcal{P}$.

Operational Aspects

- Trust structures $TS = (D, \preceq, \sqsubseteq)$ give a framework for denotational semantics for collections of mutually referring trust policies.
- No good if principals are unable to reason about their own trust in others.
- $p \in \mathcal{P}$ wants to compute $fix_{\sqsubseteq}\Pi_\lambda(p) : \mathcal{P} \to D$
  - Problem: function Π_λ is distributed as π_q, for $q \in \mathcal{P}$.
  - Problem: in principle $fix_{\sqsubseteq}(\Pi_\lambda)(p)$ depends on π_q for all $q \in \mathcal{P}$.
  - In practice, perhaps π_p depends on a significantly smaller subset.
  - Dynamically compute dependency, and then run a distributed least-fixed-point algorithm.

Algorithmic Issues

- Efficient distributed algorithms for computing *fix*;
- Policy reduction;
- Approximations often suffice!
- Abstract interpretation.
- Proof carrying requests.

A Distributed LFP Algorithm

Assume we have a trust-referencing graph already computed.

Principal a:

- Compute local trust state m_a (based on no info from other principals), and send it to all b's referencing a
- Whenever a new local trust state is received, compute a new local trust state based on this - if different from previous local trust state, send it to all b's referencing a

Some properties

Lemma

For all local trust states m_a sent by a, $m_a \preceq lfp\Pi(a)$

Assume that $\preceq$ is $\sqsubseteq$-continuous and that Π is $\preceq$-monotone.

Lemma

If for a particular snapshot $\lambda x.m_x$ we have $\lambda x.m_x \preceq \Pi(\lambda x.m_x)$, then $\lambda x.m_x \preceq fix_{\sqsubseteq}(\Pi)$.

If $m \preceq \bot_{\sqsubseteq}$ and $m \preceq \Pi(m)$, then $m \preceq fix_{\sqsubseteq}(\Pi)$.

Proof Carrying Requests

Assume r is sending a request to a which requires `$high$' level trust

$$a : \lambda x.\ulcorner b \urcorner x \times V \ldots$$
$$b : \lambda x.x = r \mapsto high;, \ldots$$

Idea: Requester provides a certain m along with his request, which is compatible with gts, i.e. $m \preceq fix_{\sqsubseteq}(\Pi)$, and such that $high \preceq m(r)$.

That is, r is asked to explain why a should trust her.

Send m to all principals x for which $m(x)$ is different from $\lambda x.\bot_{\preceq}$, and ask a to certify that $m \preceq \pi_x(m)$ (locally!). If that is the case, conclude that $m \preceq \Pi(m)$, and hence that $m \preceq fix_{\sqsubseteq}(\Pi)$.

Part II – Trust Evolution

Making Decisions

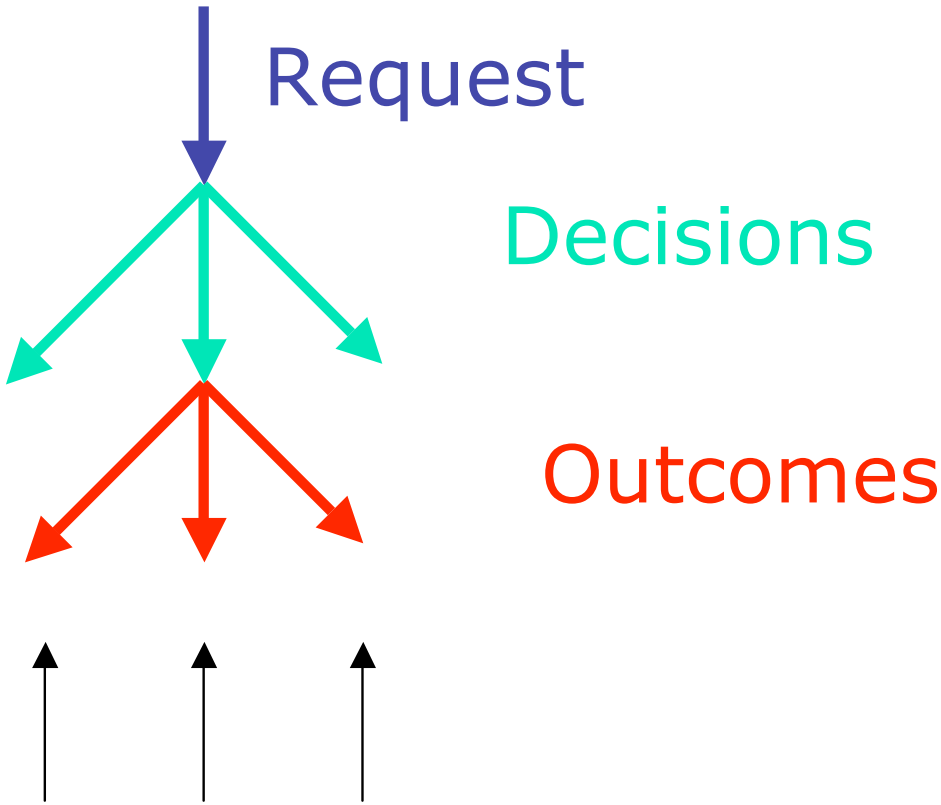
- Model: a trust decision involving entity p has a number of possible *outcomes*, $o_1, o_2, \ldots, o_n$.
- Each outcome o_i has an associated cost or benefit, say $\text{cost}(o_i)$.
- Trust values must convey enough information, that estimation of probabilities of outcomes be possible, e.g.

$$\text{expected-cost} = \sum_{i=1}^{n} \text{cost}(o_i) \cdot \text{likelihood}(o_i)$$

Trust and Risk

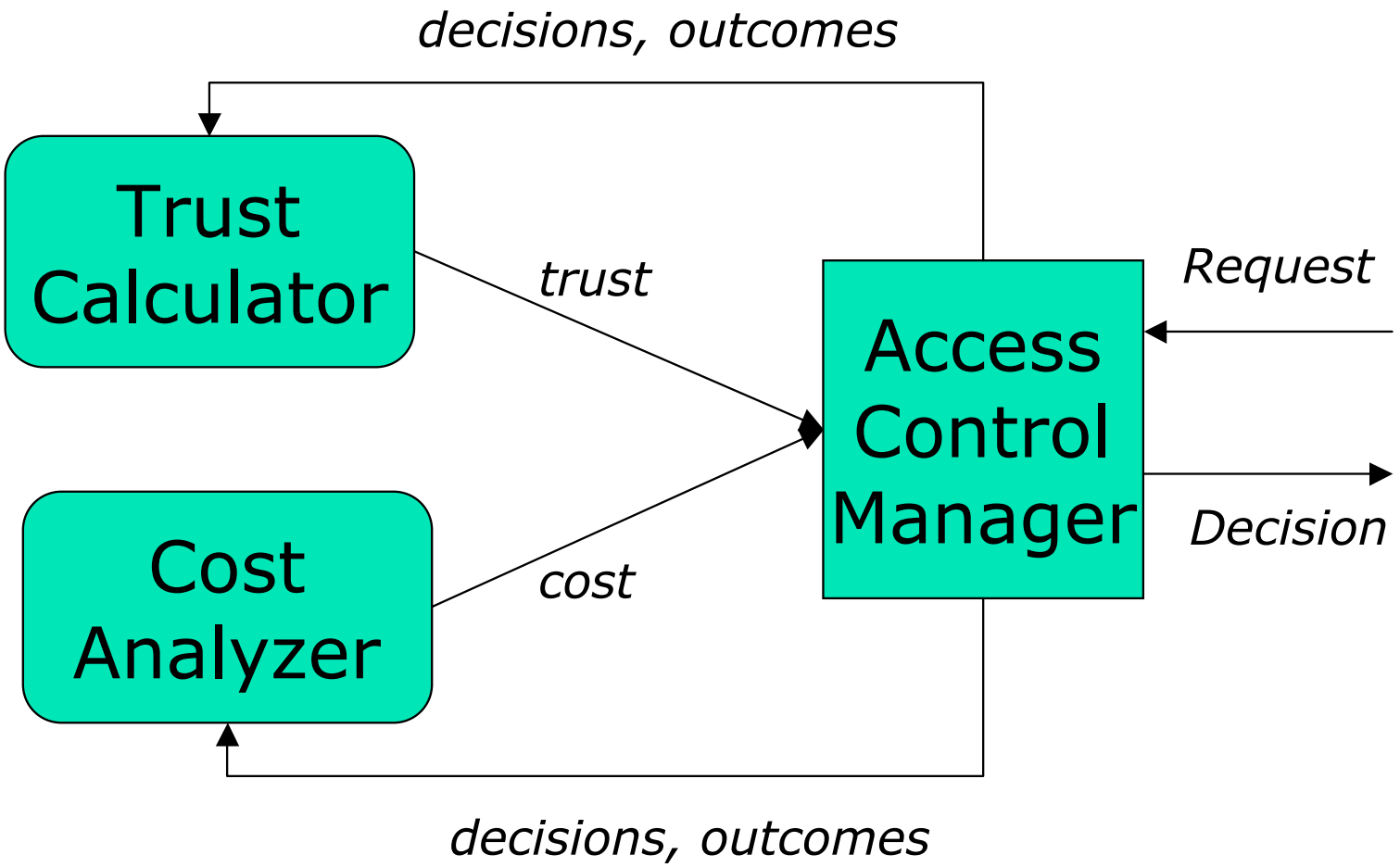
- Requests/actions are mapped to Decisions
- Decisions are mapped to possible Outcomes
- Each outcome has an associated cost / benefit to the principal
- Trust model determines the Likelihood of each outcome
- Decisions based on costs, likelihoods and local security policy

Decisions and Outcomes Flow



Trust based expected costs

A Refined Model



Trust Model

- **Goal:** Find additional structure on $\mathcal{T}$ in such a way that $\mathcal{T}$ can provide information of the form *Outcomes* $\rightarrow$ *Likelihood*.
- Remember? A trust model: mathematical framework that specifies a global trust state: gts : $\mathcal{P} \rightarrow \mathcal{P} \rightarrow \mathcal{T}$.
- But which $\mathcal{T}$? - The trust-structure framework, $TS = (D, \sqsubseteq, \preceq)$?
- Yes, but an arbitrary complete lattice is too abstract:
  - How does one estimate probabilities of outcomes?
  - How does one update trust information based on behaviour?
  - Must formalise: *outcomes*, *behaviour*.

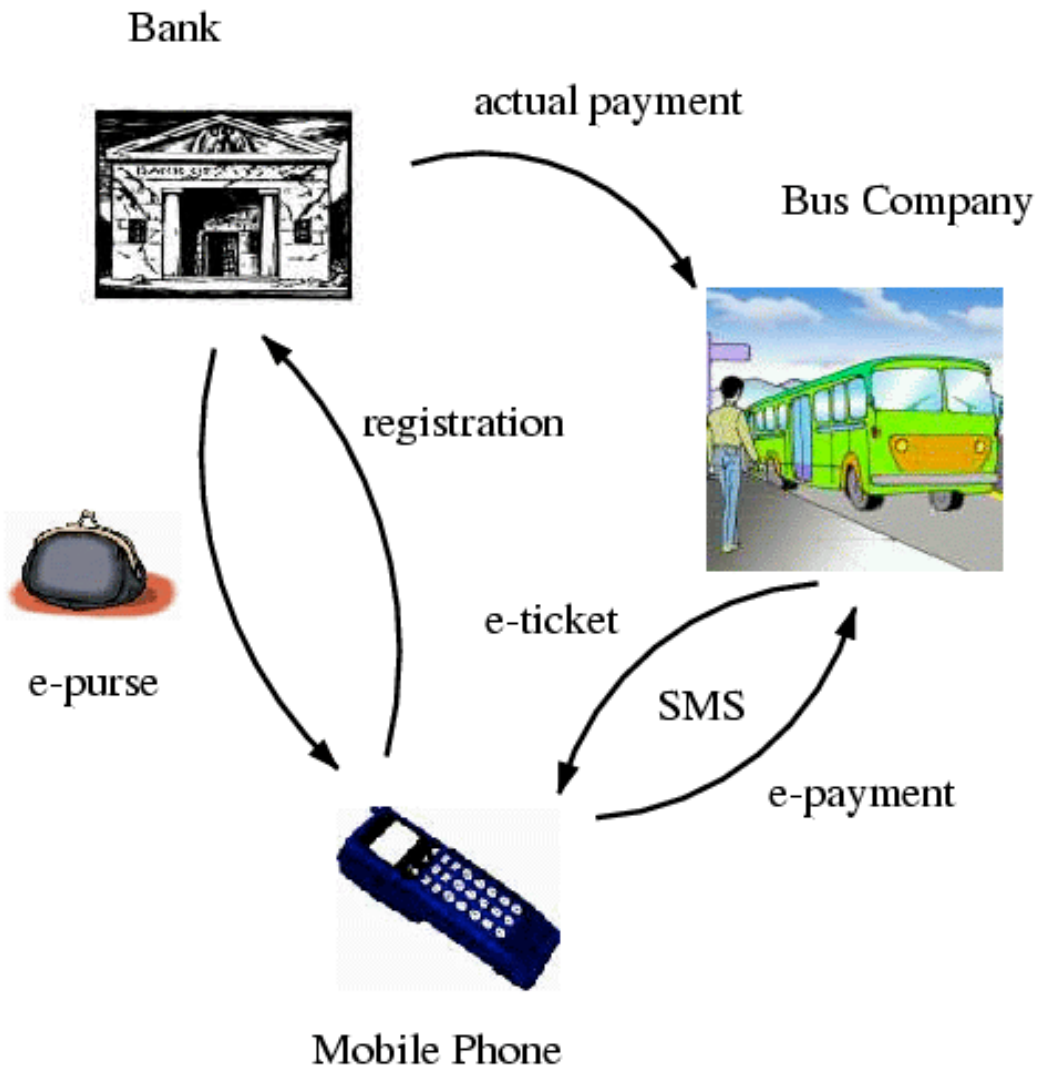
Trust Model

- **Goal:** Find additional structure on $\mathcal{T}$ in such a way that $\mathcal{T}$ can provide information of the form *Outcomes* $\rightarrow$ *Likelihood*.
- Remember? A trust model: mathematical framework that specifies a global trust state: gts : $\mathcal{P} \rightarrow \mathcal{P} \rightarrow \mathcal{T}$.
- But which $\mathcal{T}$? - The trust-structure framework, $TS = (D, \sqsubseteq, \preceq)$?
- Yes, but an arbitrary complete lattice is too abstract:
  - How does one estimate probabilities of outcomes?
  - How does one update trust information based on behaviour?
  - Must formalise: *outcomes*, *behaviour*.
- Require additional structure...
  - $\mathcal{T} = Outcomes \rightarrow EvidenceValues$
  - *Outcomes* and *EvidenceValues* also have structure...

A Running Example: E-Purse

- A GC-like scenario where entities store electronic cash in an electronic `purse'.
- Entities can transfer e-money from one e-purse to another, e.g. to purchase services.
- Entities can request a transfer of `real' money from their bank account to their e-purse.
- Scenario: User p wants to withdraw an amount, m, from its bank-account to its purse.
  - For this decision, what are the possible outcomes?

Example: E-purse – Scenario



Example: E-Purse – Outcomes

From the user's point of view, various events may occur:

- Request may be ***denied***:
  - Insufficient funds on account.
  - Server down.
  - Timeout.
  - ...
- Request may be ***granted***, and m units are transferred:
  - Bank withdraws $n \neq m$ from account.
  - Bank withdraws m from account.
  - Transferred cash is forged.
  - Transferred cash is authentic.
  - ...

Structure of outcomes

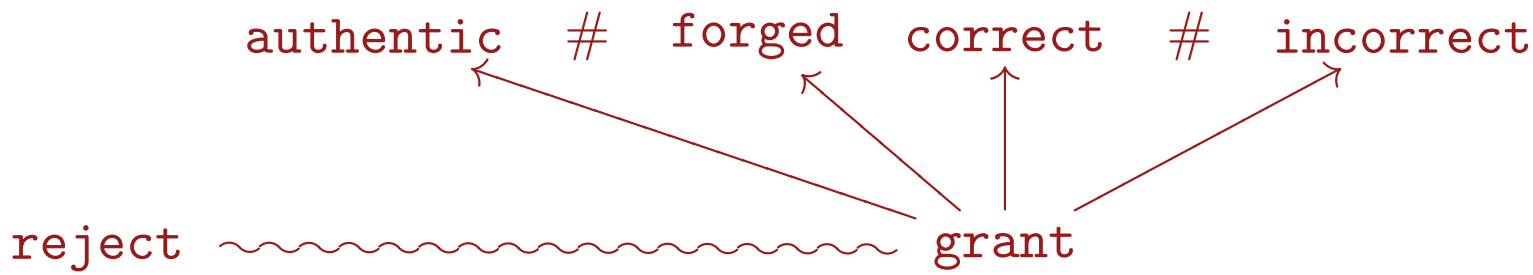
An ***outcome*** can be described by a set of observable events.

These events have structure.

- **Conflict**: both cannot occur.
  - e.g. 'denied' vs 'granted'.
- **Dependence**: a pre-condition for an event to occur.
  - e.g. 'granted' before 'forged'.
- **Independence**: none of the above.
  - e.g. 'forged' and ' correct amount withdrawn'.

Modelling outcomes and behaviour (1/2)

- A very well known model: **Event structures**
  - $ES = (E, \leq, \#)$.
  - E models the set of 'observable events'.
  - $\leq \subseteq E \times E$: dependency relation.
  - $\# \subseteq E \times E$: conflict relation.
- The E-purse example:



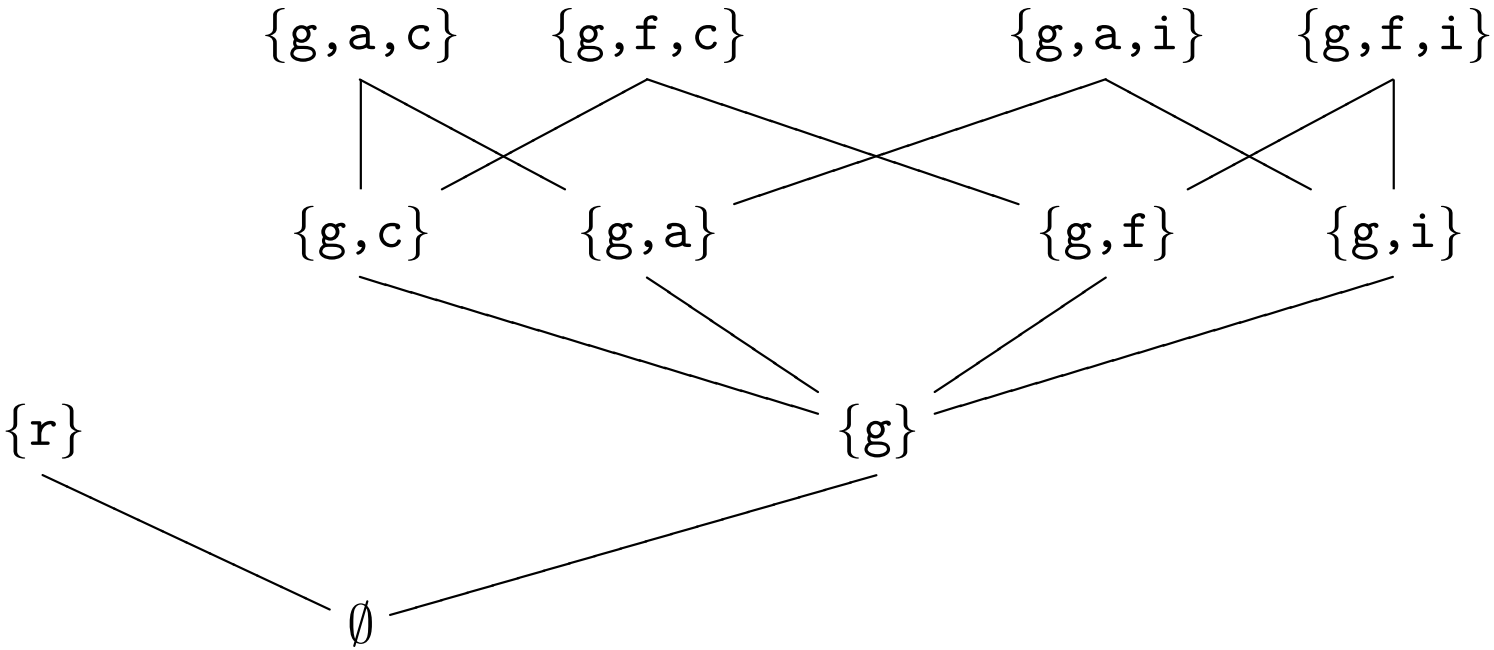
The General Idea

To model each transaction by an event structure $ES = (E, \leq, \#)$

- Each principal maintains an interaction history:
  - A sequence, $H \in \mathrm{Conf}^*_{(} ES)$, where each configuration h_i in H models information from a particular transaction
  - H is extended by either adding an event to one of the h_i's or by adding a new h.

Modelling outcomes and behaviour (2/2)

- Model: **Outcomes** as configurations
- The E-purse example:



- Model: **Behaviour** is a sequence of outcomes

Event Structures as Frames

Event structures as a common frame for interactions representing observations and outcomes:

- **Evidence History:** recording of observations (event structure configurations) based on interactions
- **Evidence Trust:** a derived evidence function on outcomes (event structure configurations)

Choosing trust values

Trust values: Outcomes $\rightarrow$ EvidenceValues.

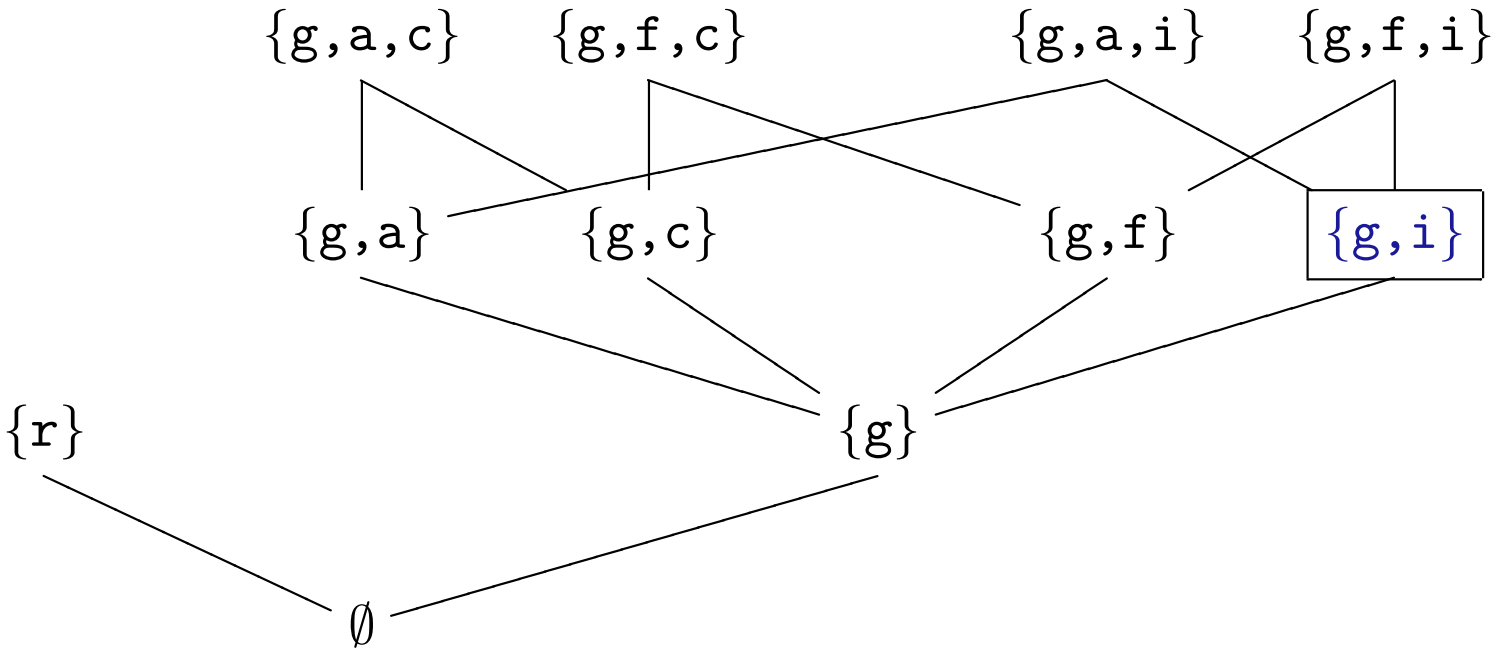
Choosing trust values

Trust values: $\mathsf{Conf}_{ES} \longrightarrow \mathsf{EvidenceValues}$.

Choosing trust values

Trust values: $\mathsf{Conf}_{ES} \to \mathsf{EvidenceValues}$.

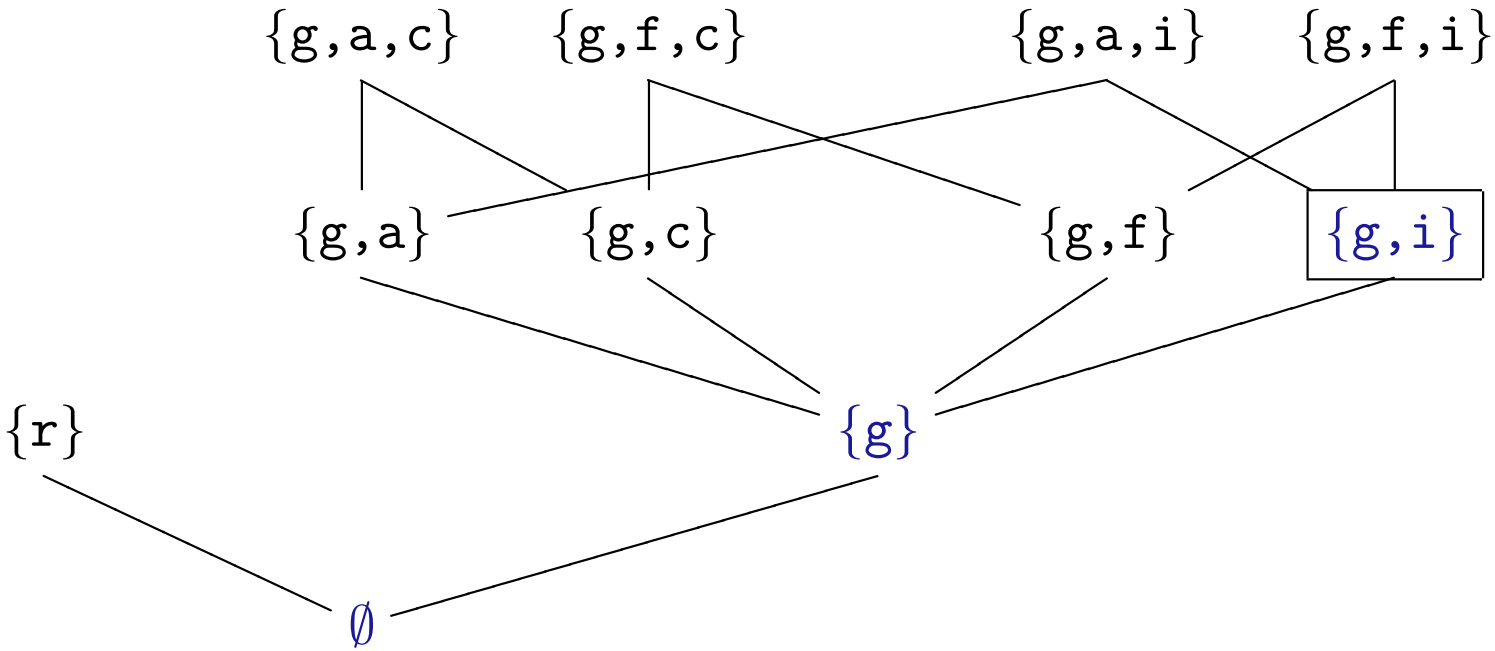
EvidenceValues?



Choosing trust values

Trust values: $\mathsf{Conf}_{ES} \to \mathsf{EvidenceValues}$.

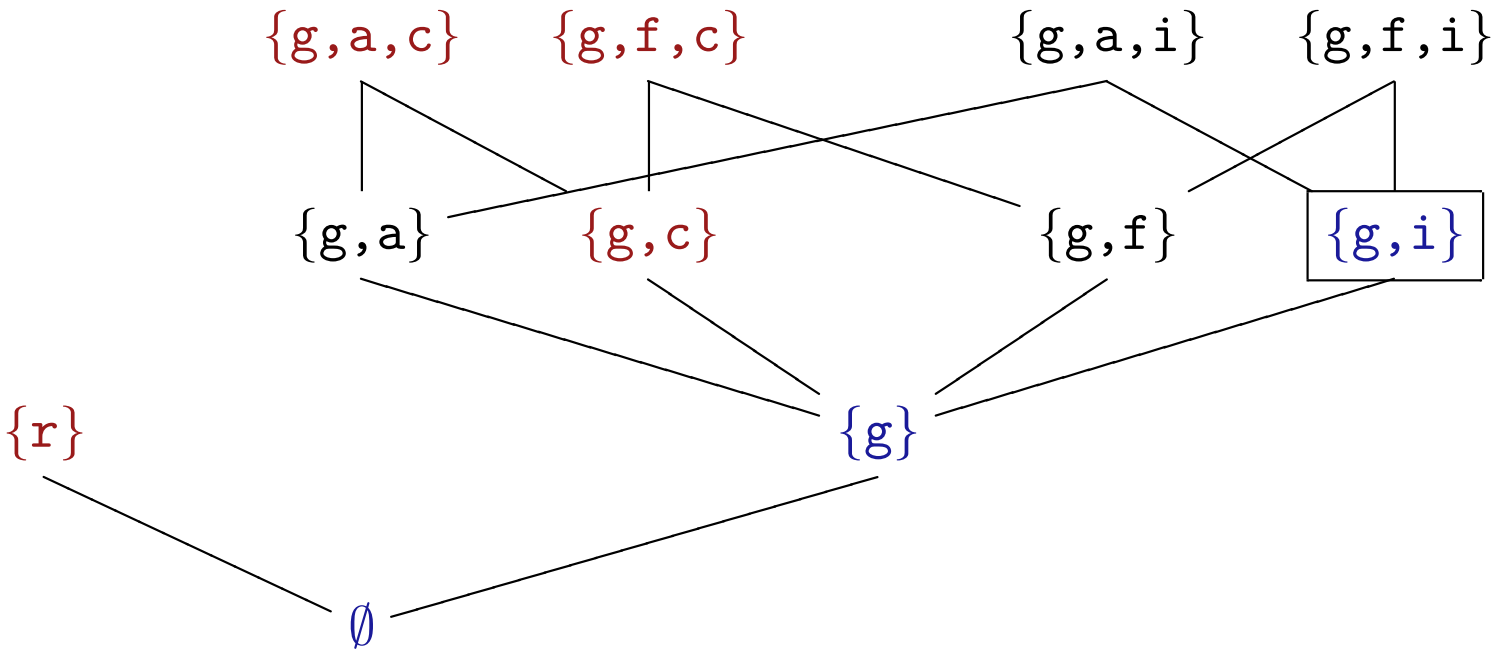
EvidenceValues?



Choosing trust values

Trust values: $\mathsf{Conf}_{ES} \to \mathsf{EvidenceValues}$.

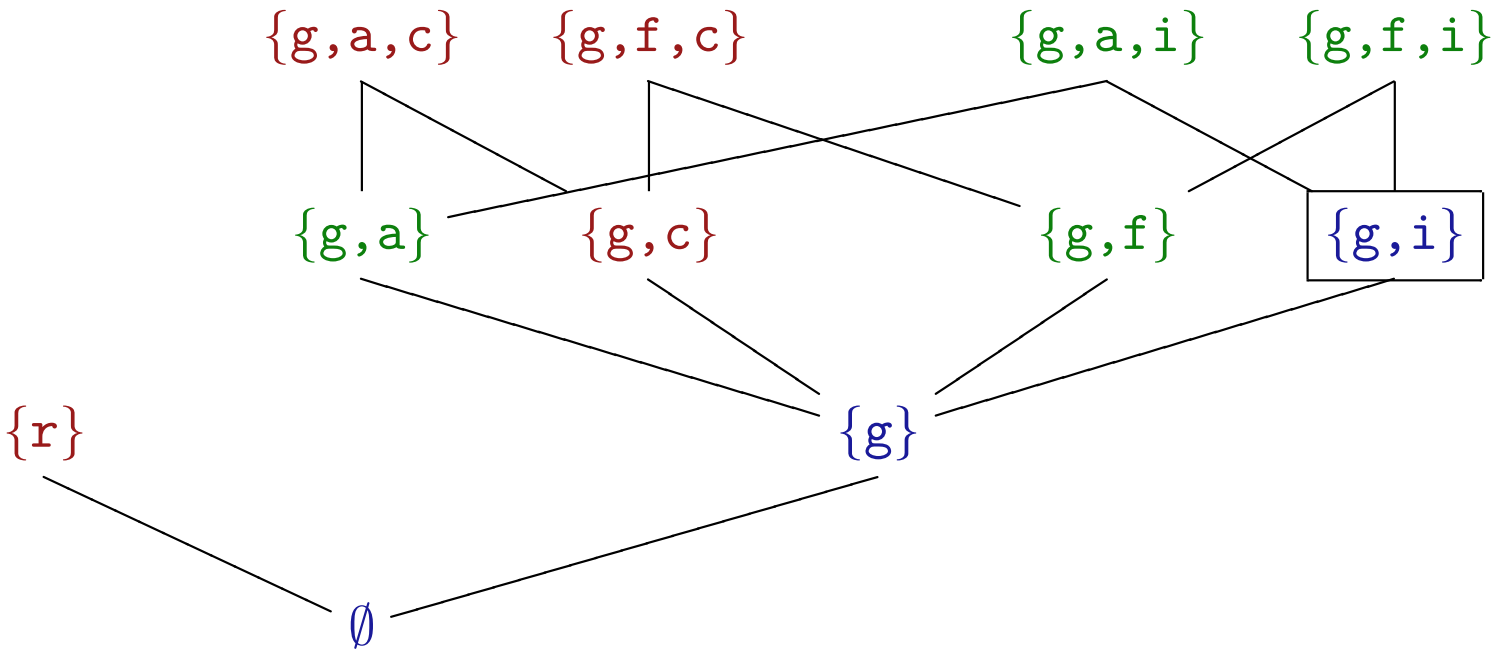
EvidenceValues?



Choosing trust values

Trust values: $\mathsf{Conf}_{ES} \to \mathsf{EvidenceValues}$.

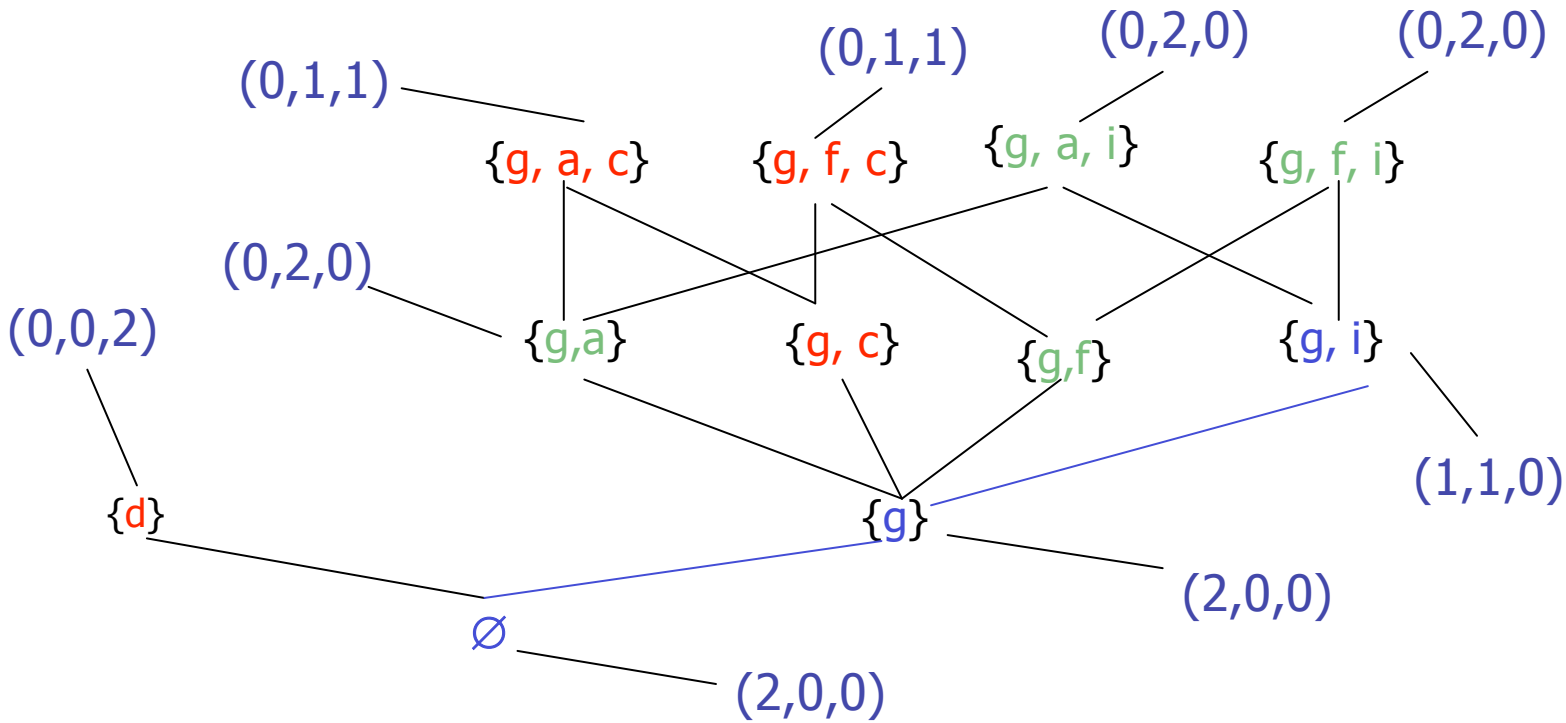
EvidenceValues?



Trust Model: Evidence Values

Assign quantitative measures to outcomes.

Just count positive and negative experiences!



Trust Model: Evidence Values – Formally

For all $x \in \mathsf{Conf}_{ES}$ the effect of x is a function $\mathbf{eff}_x : \mathsf{Conf}_{ES} \to \mathbb{N}^3$:

$$\mathbf{eff}_x(w) = \begin{cases} (1,0,0) & \text{if } w \subseteq x \\ (0,0,1) & \text{if } x \# w \\ (0,1,0) & \text{otherwise} \end{cases}$$

For a history $b = x_1 x_2 \cdots x_n$, define $\mathbf{eval} : \mathsf{Conf}^*_{ES} \to (\mathsf{Conf}_{ES} \to \mathbb{N}^3)$ by

$$\mathbf{eval}(x_1 x_2 \cdots x_n) = \lambda w. \sum_{i=1}^{n} \mathbf{eff}_{x_i}(w)$$

$\mathbf{eval}(b) : \mathsf{Conf}_{ES} \to \mathbb{N}^3$ i.e. $\mathbf{eval}(b)(w) = (s, i, c)$.

Trust Model - Recovering the Orderings

Trust structure on $T_0 = \mathrm{Conf}_{ES} \to \mathbb{N}^3$

- Define $\sqsubseteq$ on $\mathbb{N}^3$ by

$$(s, i, c) \sqsubseteq (s', i', c') \Leftarrow : (s \leq s') \wedge (c \leq c') \wedge (s + i + c \leq s' + i' + c')$$

- $(T_0, \sqsubseteq)$ is a complete lattice (up to top element, $\top_{\sqsubseteq}$)
- Define $\preceq$ on $\mathbb{N}^3$ by

$$(s, i, c) \preceq (s', i', c') \Leftarrow : (s \leq s') \wedge (c \geq c') \wedge (s + i + c \leq s' + i' + c')$$

- $(T_0, \preceq)$ is a (binary) lattice.

Trust Model – Summary

- **The Model of Trust:**
  - Decision as event structure $ES = (E, \leq, \#)$.
  - (Partial) Outcomes as configurations Conf_{ES}.
  - Behaviour as sequences of outcomes.
- Evidence values as $\mathbf{eval}(b) : \mathrm{Conf}_{ES} \to \mathbb{N}^3$ derived for each outcome from $b \in \mathrm{Conf}^*_{ES}$.
- Trust values as $\mathrm{Conf}_{ES} \to \mathbb{N}^3$.
- Trust structure on such values derived (almost).

Jøsang's Belief Logic

Belief Logic based on "approximated" truth values.
Let D be the unit interval [0, 1] of the real numbers.

$$I(D) = \{[r_0, r_1] | 0 \leq r_0 \leq r_1 \leq 1\}$$

Jøsang's Belief Logic

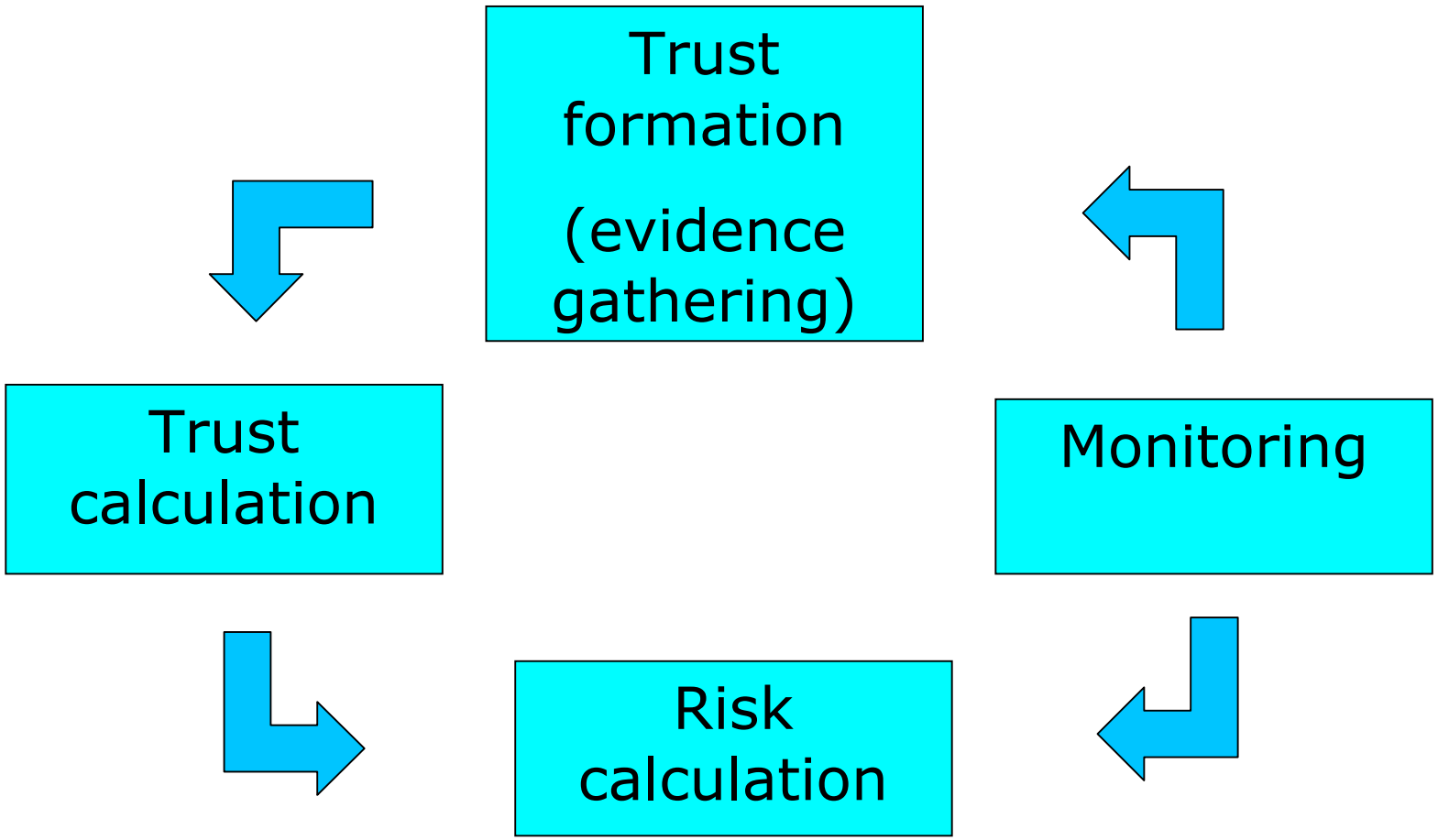
Belief Logic based on "approximated" truth values.
Let D be the unit interval [0, 1] of the real numbers.

$$I(D) = \{[r_0, r_1] | 0 \leq r_0 \leq r_1 \leq 1\}$$

Clearly related to our evidence values.

We can easily "normalise" $\mathbb{N}^3$ evidence to [0, 1] on obtain a probability distribution on maximal histories.

Trust Management – Summary



Part III – Trust Analysis

A Calculus of Trust

Systems:

$$a\{\ P\ \}_{\alpha} \mid N$$

It consists of:

- The Principal's name
- The Principal's program
- The Principal's policy
- The rest of the network

A Calculus of Trust

Systems:

$$a\{\ P\ \}_{\alpha} \mid N$$

It consists of:

- The Principal's name
- The Principal's program
- The Principal's policy
- The rest of the network

- $(b \cdot c)y\,.\,P$: Receive y from b along c, and record the observation in policy α.

A Calculus of Trust

Systems:

$$a\{\ P\ \}_{\alpha} \mid N$$

It consists of:

- The Principal's name
- The Principal's program
- The Principal's policy
- The rest of the network

- $(b \cdot c)y\,.\,P$: Receive y from b along c, and record the observation in policy α.
- $\phi :: b \cdot c\langle n \rangle$: if a can prove ϕ according to α, it will grant n to b along c. E.g.

$$(x \cdot \texttt{print})y\,.\,Access(x, ColorPrinter) :: colPr \cdot \texttt{print}\langle y \rangle$$

The Interaction Rule

Interaction

$$\frac{\beta \vdash \phi \quad \alpha' = \alpha\, \mathsf{upd}(\mathsf{b} \cdot \mathsf{c} \triangleright \tilde{\mathsf{m}}) \quad \mathsf{b} : \tilde{\mathsf{m}}\ \texttt{match}\ \mathsf{p} : \tilde{\mathsf{x}} = \sigma}{a\{\ (b \cdot c)\tilde{x}\,.\,P\ \}_{\alpha} \mid b\{\ \phi :: a \cdot c\langle \tilde{m} \rangle\,.\,Q\ \}_{\beta} \searrow a\{\ P\sigma\ \}_{\alpha'} \mid b\{\ Q\ \}_{\beta}}$$

The logic

$\mathsf{Val} = \mathsf{P} + \mathsf{N}$: all principal and method/channel names allowed.

$\overline{\mathsf{Val}} = P \times \mathsf{Val}^+$: observations $(p, ch, mess)$.

Definition

Fix any signature Σ augmented with:

- constants $\overline{\mathsf{Val}}$;
- $upd : s \times \overline{\mathsf{Val}} \to s$ (s distinguished sort, represents memory).

Definition

A message structure S, Op is a term algebra for the Σ above. Let $\mathcal{R}$ be a set of predicate symbols.

Let π be a set of Horn clauses $L \leftarrow L_1, \ldots L_k$ over such S and $\mathcal{R}$.

Principal's policies α is of the form $(\pi, \#)$, for $\# \in S$.

The calculus

Definition

$$
\begin{array}{rlr}
N, M ::= & \epsilon & \text{(empty)} \\
& \mid N \mid N & \text{(net-par)} \\
& \mid a\{\, P \,\}_{\alpha} & \text{(principal)} \\
& \mid\mid (\nu n)N & \text{(new-net)}
\end{array}
\qquad
\begin{array}{rlr}
P, Q ::= & \mathbf{0} & \text{(null)} \\
& \mid Z & \text{(sub)} \\
& \mid P \mid P & \text{(par)} \\
& \mid (\nu n)P & \text{(new)} \\
& \mid\ !P & \text{(bang)}
\end{array}
$$

$$
\begin{array}{rlr}
Z ::= & (p \cdot u)\tilde{v}\,.\,P & \text{(input)} \\
& \mid \phi :: p \cdot u\langle \tilde{v} \rangle\,.\,P & \text{(output)} \\
& \mid Z + Z & \text{(sum)}
\end{array}
\qquad
\begin{array}{rlr}
\phi ::= & L(\tilde{l}) \quad L \in \mathcal{P} & \text{(null)}
\end{array}
$$

Example: A print server

Basic predicate $Access(x, y)$, for x a principal and $y \in \{Color, BW\}$.

$$\text{Site policy } \pi : \{\ x \cdot - \rhd \texttt{junk} < 3 \rightarrow Access(x, Color),$$
$$x \cdot - \rhd \texttt{junk} < 6 \rightarrow Access(x, BW)\}$$

where $x \cdot - \rhd \texttt{junk}$ counts the occurrences of `junk` messages.

Example: A print server

Basic predicate $Access(x, y)$, for x a principal and $y \in \{Color, BW\}$.

$$\text{Site policy } \pi : \{\ x \cdot - \rhd \texttt{junk} < 3 \rightarrow Access(x, Color),$$
$$x \cdot - \rhd \texttt{junk} < 6 \rightarrow Access(x, BW)\}$$

where $x \cdot - \rhd \texttt{junk}$ counts the occurrences of `junk` messages.

Let a, the print server, and b be principals with resp. protocols:

$$P = !(x \cdot \texttt{printCol})y\,.\,Access(x, Color) :: printer \cdot \texttt{printCol}\langle y\rangle \mid$$
$$!(x \cdot \texttt{printBW})y\,.\,Access(x, BW) :: printer \cdot \texttt{printBW}\langle y\rangle$$

Example: A print server

Basic predicate $Access(x, y)$, for x a principal and $y \in \{Color, BW\}$.

$$\text{Site policy } \pi : \{\ x \cdot - \rhd \texttt{junk} < 3 \rightarrow Access(x, Color),$$
$$x \cdot - \rhd \texttt{junk} < 6 \rightarrow Access(x, BW)\}$$

where $x \cdot - \rhd \texttt{junk}$ counts the occurrences of `junk` messages.

Let a, the print server, and b be principals with resp. protocols:

$$P = !(x \cdot \texttt{printCol})y\,.\,Access(x, Color) :: printer \cdot \texttt{printCol}\langle y\rangle \mid$$
$$!(x \cdot \texttt{printBW})y\,.\,Access(x, BW) :: printer \cdot \texttt{printBW}\langle y\rangle$$

$$Q = a \cdot \texttt{printCol}\langle \texttt{junk}\rangle\,.\,a \cdot \texttt{printBW}\langle \texttt{junk}\rangle\,.\,a \cdot \texttt{printCol}\langle \texttt{junk}\rangle$$
$$\mid a \cdot \texttt{printCol}\langle \texttt{doc}\rangle$$

Consider $N = a\{\ P\ \}_{(\pi,\emptyset)} \mid b\{\ Q\ \}_{\alpha}$.

Example: A bank recommendation system

Interpret messages as recommendations.

Assume message structure is the list of the last k recommendations for each user. Let's consider the protocol

$$P = !(x \cdot \mathtt{mg})y \,.\, \mathtt{Grant}(x, y) :: x \cdot \mathtt{mg}\langle\rangle \,.\, (x \cdot \mathtt{pay})y \mid$$
$$!(\mathtt{ITAbank} \cdot \mathtt{rec})x, y$$

Policy for principal *UKBank*:

$$\pi = \{\mathtt{ITAbank} \cdot \mathtt{rec} \rhd (x, \mathtt{Bad}) + x \cdot \mathtt{pay} \rhd \mathtt{no} = 0 \rightarrow \mathtt{Grant}(x, y)\}$$

which checks if the sum of messages from `ITAbank` of type $(x, \mathtt{Bad})$ and from x of type `no` is zero.

Mortgage allowed whenever there is not bad observed or bad recommended behaviour.

System Analysis

Based on a nice cluster of behavioural equivalences I don't have time to tell you about.

- Barbed equivalence $\sim$ on networks in the usual sense
- Various interesting derived equivalences:
  - **Principal equivalence:**
    P and Q equivalent iff for all principal contexts $C_p[P] \sim C_p[Q]$
  - **Message structure equivalence:**
    m and n equivalent iff for all message contexts $C_\mu[m] \sim C_\mu[n]$.
  - **Trust Policy equivalence:**
    α and β equivalent iff for all policy contexts $C_\pi[\alpha] \sim C_\pi[\beta]$.
  - **Network equivalence:**
    N and M equivalent iff for all network contexts $C_\nu[N] \sim C_\nu[M]$.

Some Publications

- M. Carbone, M. Nielsen, V. Sassone.
  A Calculus for Trust Management, FSTTCS 2004 to appear.
- M. Nielsen, K. Krukow.
  On the Formal Modeling of Trust in Reputation-Based Systems, SLNCS 3113
- M. Nielsen, K. Krukow.
  Towards a Formal Notion of Trust, PPDP, 2003
- M. Carbone, M. Nielsen and V. Sassone.
  A Formal Model for Trust in Dynamic Networks, SEFM, 2003
- SECURE Project's members:
  Using trust for Secure Collaboration in Uncertain Environments, IEEE Pervasive Computing, 2003

Conclusion

We haven't even started yet !!

Need:

- more validation - application scenarios
- develop specification and reasoning techniques!
- develop static/dynamic policy enforcement systems
- integrate with tools
- develop models of autonomy
- ...