

# Adjustable Autonomy for Cross-Domain Entitlement Decisions

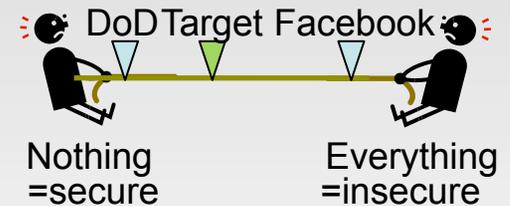
**Jacob Beal**, Jonathan Webb, Michael Atighetchi

[jbeal@bbn.com](mailto:jbeal@bbn.com), [jwebb@bbn.com](mailto:jwebb@bbn.com), [matighet@bbn.com](mailto:matighet@bbn.com)

**3<sup>rd</sup> AISEC Workshop, October, 2010**

# Problem: Cross-Domain Security

- “Need to share” vs. “need to protect”
  - Business partnership / competition
  - Medical records vs. service provision, research
  - Government inter-agency / coalition cooperation

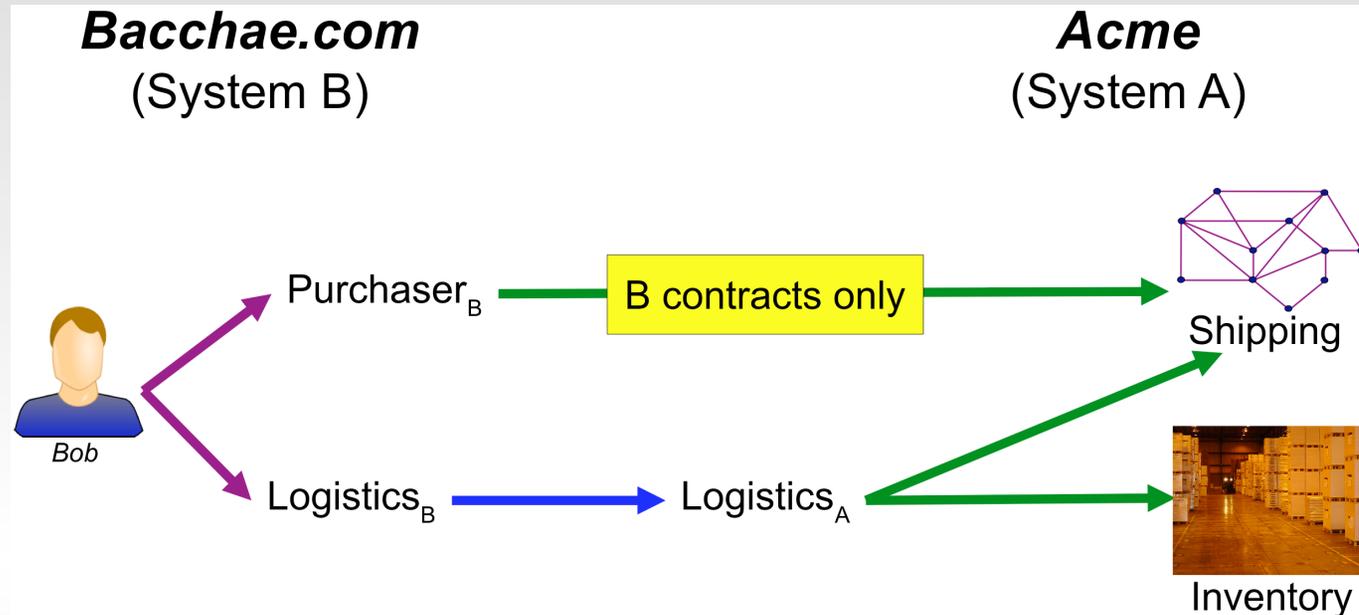


- Major challenges:
  - Increasing scale, interconnectedness
  - Many administrators, lack of mutual trust
  - More decisions by unsophisticated / hurried users
- Existing solutions do not scale / do not apply

*How can we make systems safe and humanly usable?*

*Adjustable Autonomy: machine aid... but humans in control.*

# Example: Privilege Escalation Case



1. Acme allows Bacchae purchasers filtered shipping data
2. Acme logistics staff have free access to inventory data
3. Acme gives Bacchae logistics staff access to inventory data by treating them like Acme logistics staff
4. Acme logistics staff have free access to shipping data

## ■ Interesting properties:

- All policy comes from A; all credentials and attributes from B
- Interaction of intra-domain, cross-domain mappings & references
- Failure comes from interaction of individually correct policies

# Approach: Adjustable Autonomy

*(or maybe “Domain Specific Language”)*

- Persuade human to give autonomy-enabling information:
  - Make it easy to express intent
  - Don't make them say unnecessary / subtle things
  - Specify what decisions can be made w/o a human
- Request human intervention when problems occur:
  - Success requires no false negatives, low false positives

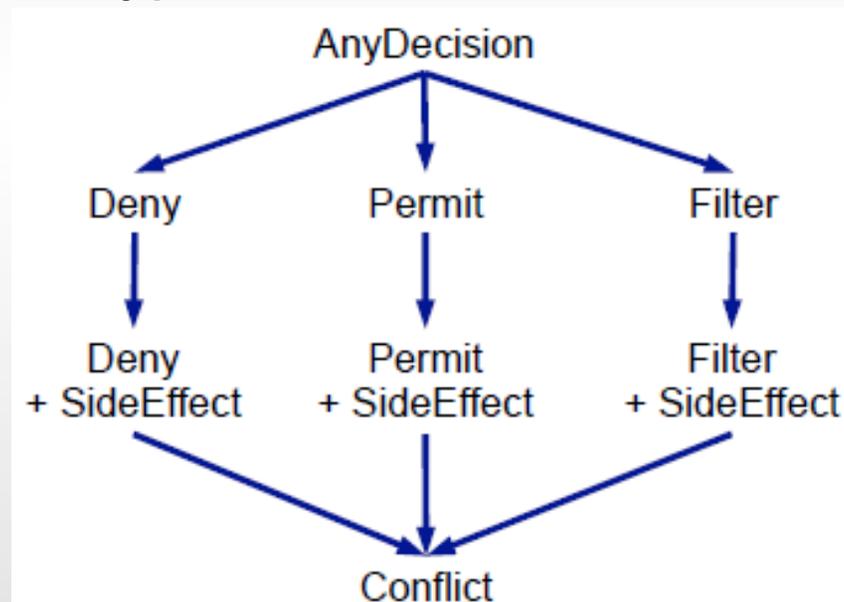
*The lesson of Graffiti: a little constraint on expression can give a whole lot of analytic power*

# Intent in Policy Combination (1/2)

- Partial order on policies:
  - Explicit: declared by administrator
    - e.g. “B purchasers permitted shipping data for B contracts only” declared to override “A logistics permitted unfiltered shipping data.”
  - Implicit: strictly more specific subject has precedence
    - e.g. “B purchasers permitted shipping data for B contracts only” implicitly overrides “Purchasers permitted unfiltered shipping data.”
- Of a set of applicable policies, consider only maxima

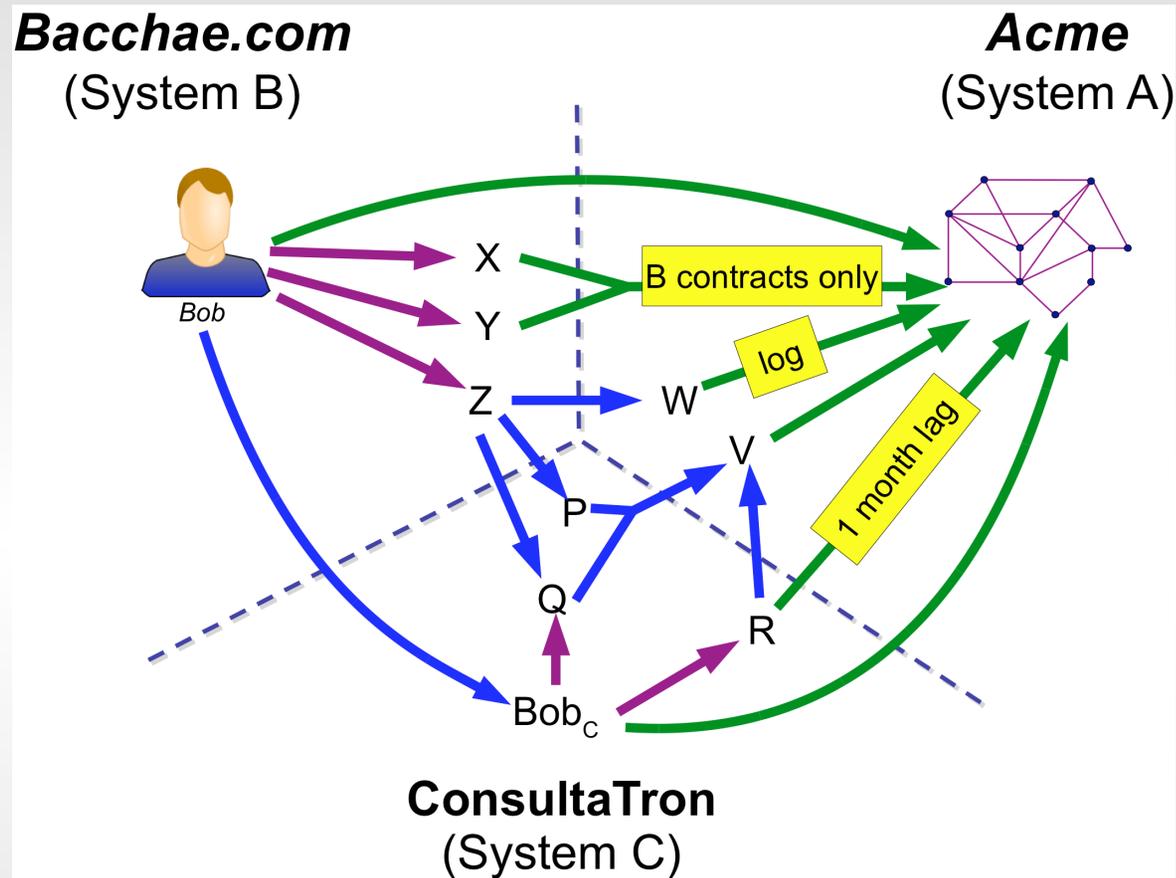
# Intent in Policy Combination (2/2)

- Three qualitative intents: Deny, Permit, Filter
  - Within these intents, combine automatically
    - e.g. “Lag” and “Filter” becomes “Lag+Filter”
  - Combining different intents = Conflict
- Formalized as type lattice:



- Filter, SideEffect are user-specified sublattices

# Cross-Domain Mappings



- Domain interactions may give many paths to access
- Must search space of mappings (cacheable)

# Decision Resolution Algorithm

- Search policies + mappings to find applicable policies
- Use implicit and explicit precedence to find maxima
  - Discard non-maximal policies
- Compute GCS (greatest common subtype) of decisions
  - Result: Compound or Conflict

*Analysis = static resolution against generic clients:  
Is it possible for there to be a problem client?*

*Lattice formulation → cheap policy validation*

# Ongoing Implementation...

- CDEL Policy modelling:

domain Acme:

```
flags: Logistics, External
```

```
import from Bacchae: Logistics->{Logistics, External}
```

```
filter ShippingData for Bacchae:Purchaser
```

```
permit ShippingData for Logistics
```

```
permit Inventory for Logistics
```

Domain Bacchae:

```
client: Bob is Purchaser, Logistics
```

```
(compute-decision Bob Acme:ShippingData)
```

```
-> Conflict
```

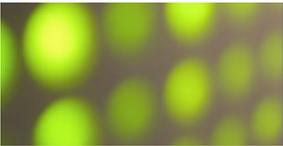
*Initial working implementation in LISP*

*XACML generation = path to backward compatibility*



# Contributions

- Partial automation by intent-capturing representation
  - Partial order of policies
  - Implicit policy ordering
  - Decision combination lattices
- Low-cost decision algorithm, analysis
- Initial implementation (CDEL)
  
- *Open Problems:*
  - *Distributed analysis & information hiding*
  - *Delegation to service agents (non-monotonicity)*
  - *Lattice acceleration of generalized security analysis*
  - *Pragmatics of implementation and deployment*



# Additional Material



# Terminology

- **Identity:** The essence of an entity. One's identity is often described by one's characteristics, among which may be any number of identifiers and attributes.
- **Identifier:** A data object (for example, a string) mapped to a system entity that uniquely refers to the system entity.
- **Attribute:** A distinct characteristic of an object.
- **Credential:** Data that is transferred to establish a claimed principal identity.
- **Policy:** A set of rules and practices that specify or regulate how a system or organization provides security services to protect resources.
- **Decision:** The result of evaluating a policy.

*(from SAML, XACML specifications)*