Trust Management

Scott D. Stoller

Stony Brook University

Outline

- Introduction to Trust Management
- Rule-Based Trust Management

Essential Features: Attributes and Relations

- Policy can use application-specific attributes and relations.
- Example: Nurses in the workgroup treating a patient can access the patient's medical record.
  - Attributes: isNurse(employee)
  - Relations: treatedBy(patient,workgroup).
- Encoding such policies as identity-based policies is
  - impractical: potential users are not known to resource owners in advance
  - dangerous: attributes can change
- In traditional RBAC, users are added to roles based on identity, not attributes.

Essential Features: Attributes and Relations

- Attributes and relations can be defined in terms of other attributes and relations.
  - Example: Nurses in the workgroup treating a patient can access the patient's medical record. A nurse is in the workgroup if a manager assigned the nurse to it.
- This allows interactions that are essential in decentralized systems.
- Standard RBAC does not support this. Each role is defined independently (aside from inheritance).
  - Example: RBAC does not support policies like role1.members = role2.members ∩ role3.members

Essential Features: Delegation

- Policy administration is completely decentralized at the top level. No globally-trusted administrators. No root of trust.
- Policies interact through delegation of authority: rely on others for attribute values, access control decisions, etc.
- Example: Hospitals, doctor's offices, insurance companies, and government agencies share information (medical, financial, and personnel records) and have limited trust.
- Example: Student discount at research conference. Conference trusts Dept of Education (U.S., etc.) about universities. It trusts each university about enrollment.

Delegation of Permissions

- Delegation of permissions: a principal with a permission can grant it to other principals (under some conditions).
  - Example: President of bank delegates permission to approve loans to VP while the President is on vacation.
  - Example: A gives B access to file. B gives access to C.
- Delegation of authority is more general.
  - Trust for information as well as authorization decisions.
  - Principals can delegate permissions they don’t have.
    - Example: Office of Grants Management can grant faculty permission to approve expenditures from an account. OGM cannot approve expenditures itself.

Essential Features of Trust Management

- Each policy statement is associated with a principal, called its source or issuer.
- Each principal's policy specifies which sources it trusts for which kinds of statements, thereby delegating some authority to those sources.
- Policies may refer to domain-specific attributes of and relationships between principals, resources, and other objects.
- Example: Acme Hospital says “doc can access pat's medical record if AMA says doc is a licensed doctor and pat says doc is treating him." (patient consent)

Outline

- Intro to Trust Management
- Rule-Based Trust Management

Simple Rule-Based Trust Management Language

- Essentially Datalog. Start simple. Extend later.
- Atom: issuer.relation(arguments)
- Argument: constant, variable, or constant(arguments)
  - Relation names and variables start with lowercase.
  - Constants start with uppercase.
  - Restrict the use of arguments so constants have bounded depth. In other words, allow tuples, not lists.
- Rule: atom :- atom1, atom2, ...
  - If atom1 and atom2 and … hold, then atom holds.
- Fact: a rule with no hypotheses.
- Policy: a collection of facts and rules.

Simple Trust Management Language: Example

- By convention, issuer.allow(principal, operation(resource)) means issuer authorizes principal to perform operation on resource. This notation is similar to [Becker+ 2004].
- The default issuer of an atom in a rule is the owner of the policy database containing the rule.
- Acme Hospital says “doc can access pat's medical record if AMA says doc is a licensed doctor and pat says doc is treating him."

```
AcmeHospital.allow(doc, Read(EPR(pat)) :-
        AMA.doctor(doc),
        pat.consentToTreatment(doc).
```

Simple Trust Management Language: Example

- SUNY says its employees can read the campus directory.
- SUNY.allow(e, Read(Directory)) :- SUNY.employee(e)
- SUNY says X is a SUNY employee if a SUNY campus says X is a campus employee.

  SUNY.employee(e) :- SUNY.campus(c), c.employee(e)
- In this example, the conclusion of a rule is used as a premise of another rule.
- Variables that appear in premises and not in the conclusion are, in effect, existentially quantified.

Compliance Checking: Goal-Directed Alg.

```
Boolean derivable(goal)
  for each rule r and substitution σ s.t. σ(r.conclusion)=goal
    for each premise p of r
      if derivable(σ(p)) continue; // try next premise
      else break; // this rule failed; try next rule
    return true  // we proved the goal using this rule
  // no rule succeeded
  return false
```

Proof of Compliance

- The goal-directed algorithm can easily be extended to provide a proof that the goal is derivable.
- A proof is a tree formed from instantiated rules, with facts from the policy at the leaves, and with the goal at the root.

$$\dfrac{\dfrac{\text{a12} \qquad \dfrac{\text{a121} \quad \text{a122}}{\text{a12}}}{\text{a1}} \qquad \dfrac{\text{a21}}{\text{a2}}}{\text{goal}}$$

Goal-Directed Algorithm: Tabling

- The simple goal-directed algorithm on previous slide may:
  - re-derive the same goal many times
    - Example: a12 and a21 could be the same.
  - diverge on recursive policies
- Goal-directed evaluation with tabling:
  - Cache all derived goals.
  - Look in the cache for an existing goal that unifies with the new goal before attempting to derive the new goal.

Outline

- Introduction and Motivation
- Design Issues and Features
    - Re-Delegation
    - Proof Search
    - Credential Gathering
    - Policy Changes
    - Trust Negotiation
    - Constraints
    - External Data
    - Separation of Duty
    - Separation of Privilege
    - Roles
    - Global and Local Names
- Trust Management Frameworks
- Sample Application Domains

Re-Delegation

- If A delegates a permission to B, can B re-delegate it to C?
- Example: Conference's policy for reviewing papers
- A PC member can submit a review of a paper.
- allow(pcmem, Submit(Review(p))) :- PCmember(pcmem)
- A PC member can designate a subreviewer.
- allow(subrev, Submit(Review(p))) :- PCmember(pcmem), pcmem.subreviewer(subrev)
- Can subreviewer S1 delegate to sub-sub-reviewer S2? No.
- S1.subreviewer(S2) doesn't work, because Conf.PCmember(S1) doesn't hold.
  - S2 could write S1's review, though.

Re-Delegation

- Re-delegation is allowed if relations are defined recursively.
- Conf: allow(rev, Submit(Review(p))) :- PCmember(rev)
- If rev can submit review, rev can designate subreviewer.
- allow(subrev, Submit(Review(p))) :-

      allow(rev, Submit(Review(p))),
      rev.allow(subrev, Submit(Review(p)))
- This allows delegation chains of arbitrary length.
- To allow delegation chains up to a specified length, use a subreviewer relation parameterized by the allowed delegation depth.

In compliance checking, who searches for proof?

- Resource owner, i.e., the policy enforcement mechanism
  - Example: medical records database server
- Requester (needs resource owner's policy) [Bauer+05]
  - Appropriate for embedded devices
  - Example: Lock with Bluetooth on Mike's office door.
  - The lock’s policy is: allow(e, Open()) :-
    Mike.allow(e, Open(OfficeDoor))
  - e's cell phone needs to present a proof of Mike.allow(e,Open(OfficeDoor)).
  - The cell phone can communicate with Mike and his delegatees. The lock can't.

Credential Gathering

- Credential: signed certificate containing a policy statement, usually a fact (in some systems, a fact or rule).
- To import a credential [iss.r(args)] signed by K: if K is iss’s key, and the signature is valid, then add iss.r(args) to the policy; otherwise, the credential is invalid.
- Compliance checking requires credentials for all subgoals with remote issuers.
- Example:

```
AcmeHospital.allow(doc, Read(EPR(pat)) :-
        AMA.doctor(doc),
        pat.consentToTreatment(doc).
```

Where To Get Credentials?

- From issuer
  - Example: request AMA.doctor(doc) from AMA
- From requester
  - Example: request AMA.doctor(doc) from doc
  - doc may have it or may request it from AMA.
- From a location specified in the policy (instead of hard-coding the decision in the evaluation algorithm).
  - Details on the next slide.

Policy-Directed Credential Gathering

- Each premise is labeled with a location (e.g., an Internet address) where the credential should be obtained. Location can be a variable. Default location is issuer.
- Notation: location◊issuer.relation(args).
- Example: request AMA.doctor credential from doctor. AcmeHospital.allow(doc, Read(EPR(pat))) :-

  doc◊AMA.doctor(doc), pat.consentToTreatment(doc).
- Example: request AMA.doctor credential from AMA.
- AcmeHospital.allow(doc, Read(EPR(pat))) :-

  AMA◊AMA.doctor(doc), pat.consentToTreatment(doc).
- Can include both of these rules in the policy.

Policy Changes

- Derived facts may be cached locally and may be sent in credentials and cached at other sites, for efficiency and fault-tolerance.
  - Example: Hospital caches AMA.doctor credential.
- These facts may become invalid due to:
  - Deletion of facts or rules.
  - Addition of facts or rules, if the policy language is non-monotonic (adding a fact or rule can invalidate facts), i.e., can express negation or equivalent.
- This is a standard problem with caching in distributed systems. Standard solutions, such as expiration dates or revocation lists, can be used.

Trust Negotiation: Gradual Release of Sensitive Credentials

- Credentials may contain sensitive information.
- Example [Bhargava+ 2004, Winsborough+ 2004]: On-Line University (OLU) gives a discount to veterans. Joe reveals his veteran status only to IRS-certified non-profits.
- Joe → OLU: register(CS101)
- OLU → Joe: requestCredential: VA.veteran(Joe)
- Joe → OLU: requestCredential: IRS.nonProfit(OLU)
- OLU → Joe: IRS.nonProfit(OLU)
- Joe → OLU: VA.veteran(Joe) // OLU: give discount
- OLU → Joe: requestPayment: $1000
- Joe → OLU: Citigroup.creditCard(Joe,1234-5678-9012)

Trust Negotiation: Privacy Policy for Credentials

- Suppose we associate privacy policies with credentials. Example [Winsborough+ 2004]:
- Joe reveals his low-income credential only to non-profits.
- Joe → E-Realty: request house listings
- E-Realty → Joe: requestCredential: IRS.lowIncome(Joe)
- Joe → E-Realty: requestCred.: IRS.nonProfit(E-Realty)
- E-Realty is not non-profit. Rich is not low-income.
- Rich → E-Realty: request house listings
- E-Realty → Rich: requestCred.: IRS.lowIncome(Rich)
- Rich → E-Realty: nothing
- E-Realty infers (no proof) Joe is low-income, Rich isn’t.

Trust Negotiation: Privacy Policy for Attributes

- Associate a privacy policy with each attribute, regardless of whether user has that attribute or a credential for it.
- Joe → E-Realty: request house listings
- E-REalty → Joe: requestCredential: IRS.lowIncome(Joe)
- Joe → E-Realty: requestCred.: IRS.nonProfit(E-Realty)
- Rich is not low-income but has the same privacy policy.
- Rich → E-Realty: request house listings
- E-Realty → Rich: requestCred.: IRS.lowIncome(Rich)
- Rich → E-Realty: requestCred.: IRS.nonProfit(E-Realty)
- E-Realty learns nothing.

Where to Get Privacy Policies for Attributes?

- Where does Rich get the privacy policy for IRS.lowIncome? Perhaps Rich never heard of IRS.lowIncome before E-Realty asked about it.
- Issuers should provide standard privacy policies for attributes, at well-known locations [Winsborough+ 2004].
- Example: When Rich sees request for IRS.lowIncome credential, he contacts IRS policy server and obtains and uses the IRS-recommended privacy policy for attribute IRS.lowIncome.
- If Rich doesn't bother to do this, then other people probably won't do it for other attributes, which Rich might want to keep private.

Trust Negotiation Strategies

- Trust negotiation policy determines when a credential may be released.
- Trust negotiation strategy determines which releasable credentials are released.
- Eager Strategy: at each step, send all releasable credentials.
- Targeted Strategy: at each step, send releasable credentials that help achieve the current goal.

Trust Negotiation Strategies: Example

- Eager Strategy (fewer rounds of communication):
- Joe → NP-Realty: request house listings
- NP-Realty → Joe: requestCred.: IRS.lowIncome(Joe); IRS.nonProfit(NP-Realty), BBB.member(NP-Realty), ...
- Joe → NP-Realty: IRS.lowIncome(Joe)
- Targeted Strategy (fewer credentials sent):
- Joe → NP-Realty: request house listings
- NP-Realty → Joe: requestCred.: IRS.lowIncome(Joe)
- Joe → NP-Realty: requestCred.: IRS.nonProfit(NP-Realty)
- NP-Realty → Joe: IRS.nonProfit(NP-Realty)
- Joe → NP-Realty: IRS.lowIncome(Joe)

Constraints

- Constraint: a premise that uses an externally defined relation on a data type. Common examples include:
- Numerical inequalities
  - Example: allow(empl,Read(file)) :-
    securityLevel(empl,m), securityLevel(file,n), $m \geq n$.
- Prefix-of relation on sequences (e.g., pathnames)
  - Example: allow(stu, Read(file)) :-
    Registrar.enrolled(stu, CSE306),
    /CSE306/project/ prefix-of file.
- These relations can't be defined in Datalog.

External Data

- Policy may depend on external data.
- Example: personnel database: employees, their department and rank
- Example: EHR database: author of each entry
- Storing this info in policy database would be inefficient.
- How does the policy access it?
    - Request credentials from the DBMS. This is inefficient and unnecessary, assuming DBMS is local and trusted.
    - Use a connector that makes the DBMS look like part of the policy database. Neat, because policy language and DBMS are both relational.

External Data: Connector to DBMS

- Each table corresponds to a relation.
- Each record corresponds to a fact.
- Connector generates SQL queries to retrieve relevant data.
- Example: allow(e, read(Budget(dept)) :-
  deptSeniorPers(sp, dept), sp.allow(e, read(Budget(dept))).
- sp is unbound when deptSeniorPers is evaluated.
- If deptSeniorPers is external, the generated SQL query finds and returns all senior personnel of the department.
- If results from DBMS are cached, they must be invalidated if a DBMS update changes the relevant data.

External Functions

- Manipulate data
  - Example: selectors for compound data structures
- Provide environment and context information
  - Example: allow(stu, Read(file)) :- Registrar.enrolled(stu,CSE306),
    /CSE306/project/ prefix-of file,
    currentTime() > 09:00.1feb2006.
- Provide simple interface to external data (file, DBMS, …).
  - Arguments must be ground (constants) at call time.
  - Example: Note: author is an external function.
    allow(e, Update(rec)) :- employee(e), e=author(rec).

Object-Based Separation of Duty

- Separation of duty limits the set of permissions of a single user. This helps prevent fraud, which requires collusion.
- Example: A single employee may perform at most 1 of the 3 steps involved in a purchase: issue purchase order, verify receipt of goods, issue payment.
- Object-based separation of duty allows an employee to perform at most 1 of these operations for a single purchase.
- Example: allow(e, IssuePayment(trans)) :-

  acctgClerk(e),

  e ≠ getPurchClerk(trans), e ≠ getRcvClerk(trans)
- getPurchClerk(trans): clerk who issued the PO for trans

Separation of Privilege

- Separation of privilege: an action is permitted only if a specified number of authorized users request it.
- issuer.allow2(principal1,principal2,operation(resource)) means issuer authorizes principal1 and principal2 jointly (together) to perform operation on resource.
- Example: allow2(clerk, mgr, IssuePayment(amount)) :-

  AcctgClerk(clerk), AcctgManager(mgr),

  clerk ≠ mgr, amount < 1,000,000.

  allow(clerk, IssuePayment(amount)) :-
  AcctgClerk(clerk), amount < 10,000.
- Alternative: Decompose the action into multiple actions.
- Example: clerk: InitiatePayment, mgr: ApprovePayment.

Roles

- Parameterized role: r(args). Abbreviate r() as r.
- Example: Manager(department), Guardian(patient)
- "p is a member of r(args)" can be represented as
  - r(p, args)   roles as relations
  - member(p, r(args)) roles as values
- Permission-role relation is defined by rules like:
  - permit(p, oper(resource)) :- r(p,args), ...
  - permit(p, oper(resource)) :- member(p,r(args)), ...
- Roles as values allows variables that range over roles, but this can be simulated with roles as relations.

Role Hierarchy

- Role hierarchy can be expressed by rules for inheritance of membership.
- Example: Manager $\geq$ Employee is expressed by

  member(e, Employee) :- member(e, Manager).
- Role hierarchy could be expressed instead by rules for inheritance of permissions if we made the permission-role relation PR(action,role) explicit.

Global and Local Names

- Local names: names of attributes and relations include the name of a principal, called its source or issuer.
  - Example: AMA.doctor(Dan), BMA.doctor(Dan)
  - Statements about src.r may be issued only by src.
- Global names: shared namespace for attributes & relations.
  - Less structured, but more flexible.
  - Example: AcmeHosp.member(Dan, Doctor(AMA))
- RT [Li+ 2003]: local. Cassandra [Becker+ 2004]: global.
- An ontology can provide common meaning for names.
- Local names for principals, e.g., [SBU President]. Used in SPKI/SDSI. Can be simulated using parameterized roles.

Outline

- Introduction and Motivation
- Design Issues and Features
- Trust Management Frameworks
  - List of several proposed frameworks on next slide.
  - We'll discuss a few representative frameworks.
- Sample Application Domains

Some Trust Management Frameworks

- Authentication in Distributed Systems, Taos [Burrows, Abadi, Lampson, Wobber, 1992-1994]
- PolicyMaker, Keynote [Blaze, Feigenbaum, et al., 1996-9]
- SPKI/SDSI [Rivest, Lampson, et al., 1997-1999]
  - Simple PKI / Simple Distributed Security Infrastructure
- Delegation Logic, RT [Li et al., 2000-present]
- SD3 [Jim 2001]
- Binder [DeTreville 2002]
- TrustBuilder [Seamons, Winslett, et al., 2002-present]
- PeerTrust [Nejdl, Olmedilla, et al., 2003-present]
- Cassandra [Becker and Sewell, 2004-2005]

PolicyMaker [Blaze, Feigenbaum, et al.]

- PolicyMaker is a blackboard-based trust management architecture. The blackboard contains
  - requests (goals): action/statement to be authorized
  - acceptance records: issuer allows action/statement
- Policy: functions that read requests and acceptance records from the blackboard and write acceptance records.
  - Use any safe functional programming lang. (SafeAWK)
- PolicyMaker is flexible but offers minimal functionality.
  - Application gathers credentials, verifies signatures, etc.
  - AWK interpreter (or …) evaluates policy functions

SPKI/SDSI [Rivest, Lampson, et al.] Name Certificates

- Local names for principals. The meaning of local names is given by name certificates that relate local names to each other and to global identifiers (public keys).
- Format: [K, name, subject, validitySpec] signed by K
- Meaning: local name K name refers to subject
- subject may be a name or a public key
- Example: [K-SBU-CS, Chair, K-Ari, exp. june 2007]
- A local name mapped to multiple keys is a group name.
- Example: [K-SUNY, Student, K-Joe, exp. may 2006]
  [K-SUNY, Student, K-Mary, exp. may 2006]
- validitySpec: expiration date, CRL location, ...

SPKI/SDSI: Authorization Certificate

- Format: [K, subject, deleg, tag, validitySpec] signed by K
  - Not using official syntax.
- Meaning: issuer K gives permission tag to subject.
  - deleg indicates whether subject can delegate the permission (in addition to using it himself).
- subject may be a name (defined by other certificates), a public key, a threshold structure, or an object hash (ignore)
  - A threshold structure [{K1,K2,…}, n] means any n of the listed keys can together authorize the delegated action (separation of privilege).

SPKI/SDSI: Authorization Certif. Examples

- [K-SBU, [K-SBU Faculty], false, readDir, exp. 2010]
- Example with delegation:
- [K-SBU, [K-SBU Faculty], true, (getRoster *), exp. 2010]
- Name Certificate: [K-SBU, Faculty, Scott, exp. 2009]
- [K-Scott, K-Tom, false, (getRoster CSE101), exp. 2007]
  - Tom is the TA. He can't delegate this permission.
- Authorization certificates define one relation: allows (delegates).
- Role-based policies can be expressed using groups

Limitations of SPKI/SDSI

- Delegation and authorization and not distinguished: a principal must have a permission in order to delegate it.
  - Example [De Treville 2002]: DMV must be a licensed driver in order to be authorized to license drivers.
- Only unary relations on principals, expressed as groups, are supported.
  - Can't express policies like: "Nurses in the workgroup treating a patient can access the patient's medical record“, which uses relation treatingWorkgroup(pat,grp)
- No variables (parameters) in tags. No conjunction of groups/attributes. No trust negotiation.

Binder [DeTreville 2002]

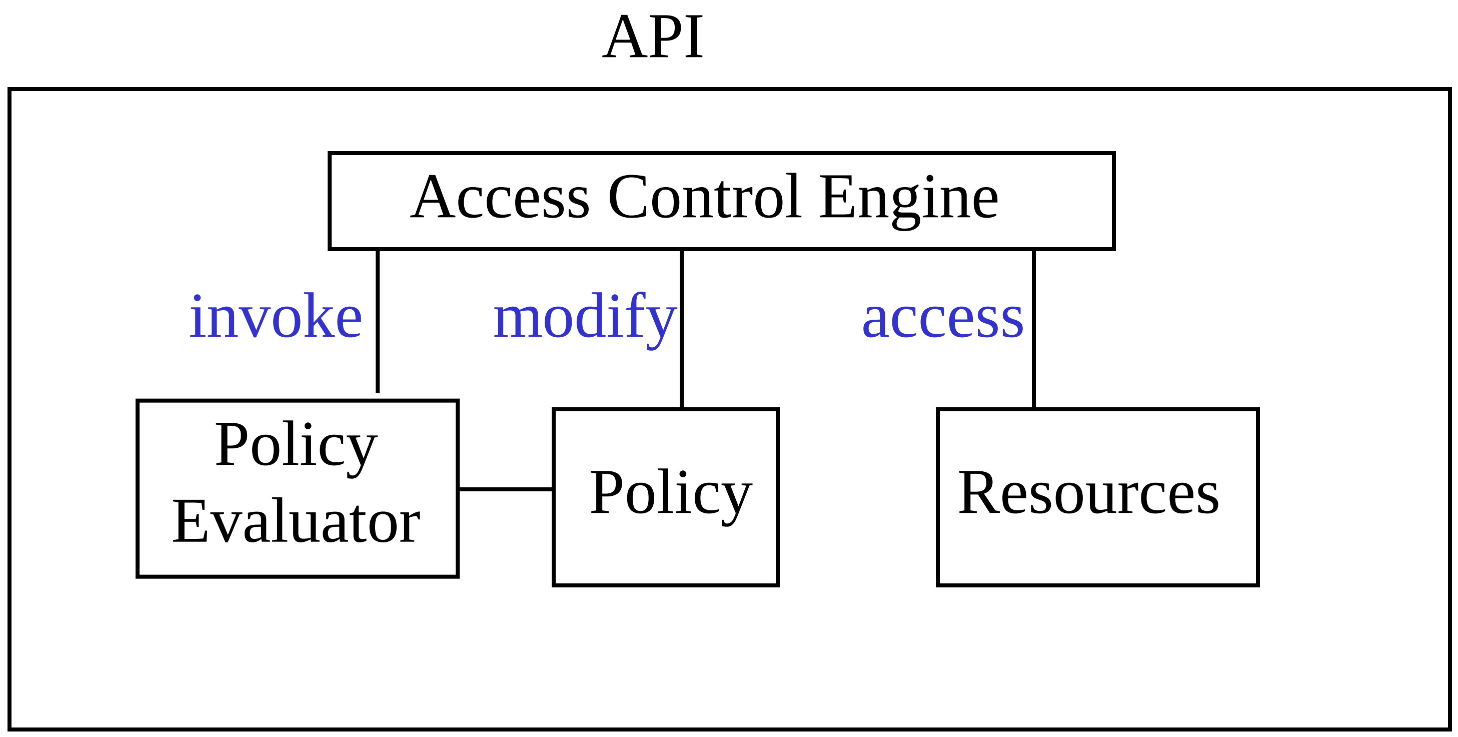
- Our simple policy language (without constraints) is very similar to Binder's.
- Authorization relation:

  issuer says can(principal,operation,resource)
- Nicely written paper; recommended as an introduction to rule-based trust management
- Allows communication of rules (discussed later), although the details are unspecified
- Does not consider trust negotiation.

Cassandra [Becker and Sewell, 2004-2005]

- Lang: Datalog + constraints + aggregation (non-monotonic)
- Role-based
- Parameterized roles, parameterized actions (permissions)
- Global names (simulate local names using issuer and other parameters)
- Goal-directed evaluation with memoization (tabling)
- Policy-controlled credential gathering
- Role activation (but not sessions; details below)
- Trust negotiation
- External functions

Cassandra Architecture



- Policy: local policy + cached credentials
- Alternative architecture: return authorization decisions to the application

Cassandra Predicates

- Syntax for Predicates: location◊issuer.pred(args)
- Special Predicates (special significance in the API):
- permits(entity,action): entity is authorized to perform action
- canActivate(entity, role): entity can activate (is a member of) role
- hasActivated(entity,role): entity has activated role
  - Role activations are recorded as facts in the policy.
  - No explicit notion of sessions; implicitly, there is one session per Cassandra instance.

Facts as Role Activations

- Many kinds of facts are expressed as role activations. An entity says fact(args) by activating the "role" fact(args).
  - This moderately simplifies the API.
- Example: A patient consents to treatment by Dr. Dan by activating the role ConsentToTreatment(Dan).
- Example: The manager of department dept appoints an employee e in dept by activating AppointEmployee(e,dept)
- canActivate(mgr, AppointEmployee(e, dept)) :- hasActivated(mgr, Manager(dept))
- canActivate(e, Employee(dept)) :-

  hasActivated(mgr, AppointEmployee(e, dept))

Cassandra Predicates: canDeactivate

- canDeactivate(entity,entity1,role): entity is authorized to deactivate entity1's activation of role
- Example:

```
canDeactivate(pat,pat,ConsentToTreatment(doc)) :- true.
canDeactivate(grdn,pat,ConsentToTreatment(doc)) :-
      hasActivated(grdn,Guardian(pat)).
```

- Cassandra does not consider administrative policy, so there is no notion of who is authorized to remove an entity from a role (by changing the policy).

Cassandra Predicates: isDeactivated

- isDeactivated(entity,role): entity's activation of role is being deactivated. Used to trigger other role deactivations.
- Example: A user being removed from the Employee role should also be removed from the Manager role.
- isDeactivated(e, Manager()) :-

  isDeactivated(e, Employee()).
- Example: A doctor being removed from "on duty at hospital" should also be removed from "attending doctor".

  isDeactivated(doc, AttendingDoctor(pat)) :-
  isDeactivated(doc, OnDuty()).
- Could add premise hasActivated(doc,AttendingDoctor(pat))

Cassandra Predicates: canRequestCredential

- This predicate expresses trust negotiation policy.
- canRequestCredential(entity, issuer.r(args)): entity is authorized to request credentials that match issuer.r(args).
  - Abbreviate as canRequestCred
- Example: OLU allows a registered student to request a credential showing this.
- stu is registered in semester sem ↔ stu has activated Student(sem).
- canRequestCred(stu,OLU.hasActivated(stu,Student(sem))) :- hasActivated(stu,Student(sem)).
  - Response: "here's the certif" or "unauthorized request"

Cassandra Predicates: canRequestCredential

- Continuing the example, consider an alternative rule:
- canRequestCred(stu,OLU.hasActivated(stu,Student(sem))) :- true.
  - Same effect as previous policy, except response to requests by non-student asking about own status is "You are not registered as a student."
- OLU allows a student's parent to request that credential.
- canRequestCred(par,OLU.hasActivated(stu,Student(sem))) :- parentOf(par,stu).

Cassandra Predicates: canRequestCredential

- A student can delegate authority to get his Student credential to anyone (e.g., potential employer).
- OLU's policy: canRequestCred(e,OLU.hasActivated(stu,Student(sem))) :- stu.canRequestOLUreg(e).
- Joe Cool's policy: JoeCool.canRequestOLUreg(Google).
- An entity can have a canRequestCredential policy for credentials (attributes) it does not have.
- Example: Every user can have the policy
- canRequestCredential(c, IRS.lowIncome(y)) :- IRS.nonProfit(c)

Cassandra API: doAction, activate

S: site where the operation is invoked.

P: S's policy.

P |- fact: fact is derivable from P.

e: the entity invoking the operation.

```
e:doAction(a)
  if P |- S.permits(e,a)
    execute action a.

e:activate(r)
  if P |- S.canActivate(e,r) and not P |- S.hasActivated(e,r)
    add S.hasActivated(e,r) to P
```

Cassandra API: deActivate

```
e:deactivate(e1,r)
 if P |- S.hasActivated(e1,r)  and  P |- S.canDeactivate(e,e1,r)
   add S.isDeactivated(e1,r) to P
   D = { [e2,r2] | P |- S.isDeactivated(e2,r2) }
   // note: D contains (e1,r)
   for [e2,r2] in D
     remove S.hasActivated(e2,r2) from P (if present)
   remove S.isDeactivated(e1,r) from P
```

To handle rules like isDeactivated(…) :- ¬hasActivated(…), another loop is needed.

deActivate: Example

Initial policy P:

isDeactivated(e, Manager()) :- isDeactivated(e, Employee())

hasActivated(Mike, Employee())

hasActivated(Mike, Manager())

canDeactivate(Charles,Mike,Employee())

Charles:deActivate(Mike,Employee())

Add isDeactivated(Mike, Employee()) to P.

Then P |- isDeactivated(Mike, Manager()).

So D = { [Mike,Employee()], [Mike,Manager()] }

Cassandra API: requestCredential

- may be invoked directly or via a remote premise.
- iss.r(args) must be ground. Ignore constraint creds here.

```
e:requestCredential(iss.r(args))
 if P |- S.canRequestCredential(e, iss.r(args))
   if iss = S
     if P |- S.r(args)  return S.r(args) signed by S
     else return “not S.r(args)”
   else // iss ≠ S.  Forward cached credential, if any.
     if P contains iss.r(args)  return iss.r(args) signed by iss
 else return “unauthorized request”
```

Cassandra: Constraints and Aggregation

- Constraints over integers, sets, other domains.
- Aggregation operators:
- Group: collect the facts that match a pattern S.r(args) into a set
- Count: count the number of facts that match a pattern S.r(args)
- Variables in S.r(args) not used elsewhere in query may have any value
- S is the local entity; otherwise, answer may be incomplete.

Non-Monotonic Policies

- Aggregation + constraints allow non-monotonic policies
- Example: Dynamic Separation of Duty: no user may have the Doctor and Patient roles active concurrently.

```
canActivate(doc, Doctor()) :-
        AMA.Doctor(doc),
        COUNT(hasActivated(doc, Patient())) = 0
```

- This is my own syntax. Cassandra's syntax is more general but harder to read.
- Non-monotonic because adding hasActivated(Dan,Patient()) changes canActivate(Dan,Doctor()) from true to false.

Example: Chinese Wall

- To avoid conflict of interest, a consultant can work on at most one project involving each industry sector. (This simple policy does not consider information flow.)
- Example: can't work on projects for Intel and AMD, both in semiconductor sector.
- industrySector(proj,sec): proj involves industry sector sec.
- Example: industrySector(IntelReengg, Semiconductor).
- employeeSector(emp,sec): employee emp is working on a project in sector sec.
- employeeSector(emp,sec) :-
  hasActivated(mgr,AppointEmployee(emp,proj)),
  industrySector(proj,sec)

Example: Chinese Wall

- canActivate(mgr,AppointEmployee(emp,proj)) :-
  Manager(mgr,proj), industrySector(proj,sec),
  COUNT(employeeSector(emp,sec)) = 0.

Outline

- Introduction and Motivation
- Design Issues and Features
- Trust Management Frameworks
- Sample Application Domains
  - Electronic Health Records (EHR)
  - Other application domains

Exercise: Express This Policy

1. OLU: A student can read his or her own record.
2. OLU: Someone claiming a student as a dependent on his tax return, according to IRS, can read the student's record.
3. IRS: Information about dependents is provided to universities registered with the Education Dept (ED).
4. ED: Info about registered universities is avail to everyone.
5. OLU:A faculty can assign a grade for a student enrolled in a class he is teaching.

- Actions: ReadRec(stu), AssignGrade(class,stu)
- Relations: taxDependent(dependent, filer), isUniv(univ), teaching(faculty, class), enrolled(student, class), canActivate(e, role), canReqCred(e, cred), permits(e, act)

A Solution

1. OLU: permits(stu, ReadRec(stu)) :- true.
2. OLU: permits(p, ReadRec(stu)) :-
   IRS◊IRS.taxDependent(stu, p)
3. IRS: canRequestCredential(u, taxDependent(x, y)) :-
   ED◊ED.isUniv(u).
4. ED: canRequestCredential(x, isUniv(u)) :- true.
5. OLU: permits(fac, AssignGrade(class, stu)) :-
   teaching(fac,class), enrolled(stu,class).

Electronic Health Records (EHR)

- Promising application for RBAC and trust management
- People and organizations with limited trust must share sensitive information: patients, doctors, nurses, hospitals, billing companies, insurance companies, government agencies (e.g., Medicaid, FDA), professional societies (e.g., AMA), medical researchers, etc.
- More interactive information sharing will increase the need for trust management.
- Case study [Becker 2005]: Output Based Specification for Integrated Care Record Service, version 2, 2003. Developed by the National Health Service (NHS) of the United Kingdom.

EHR Policy

- Spine: a nationwide EHR system
  - one electronic health record (EHR) per patient
  - multiple items per record
- Registration Authority (RA): issues credentials for clinicians, with name, affiliation, specialty, etc.
  - typically for one organization, but may be regional
- Local health organizations: hospitals, doctors' offices, etc.
  - one electronic patient record (EPR) per patient, with full data
- Patient Demographic System (PDS)
  - One nationwide PDS

Registration Authority Policy

- Main Role: RA-manager
- A manager registers a clinician by activating NHS-Clinician-cert(...).
- canActivate(mgr,
            NHS-clinician-cert(org, cli, spcty, start, end)) :-
  hasActivated(mgr, RA-manager()),
  hasActivated(y, NHS-health-org-cert(org, start2, end2)),
  start in [start2, end2],
  end in [start2, end2],
  start < end

Spine Policy: Main Roles and Main Actions

- Clinician
  - request consent to treatment, read and update EHR
  - emergency access to EHR
  - refer a patient to another clinician
  - approve requests to seal items
  - appoint agents for patient (if patient is unable to)
  - conceal items (from patient or other clinicians)
- Administrator
  - register new patients, clinicians, and administrators
  - unregister old ones

Spine Policy: Main Roles and Main Actions

- Patient
  - one-off (i.e., one-time) consent to policy
  - consent to treatment
  - appoint agents
  - read items in EHR
  - request that items in EHR be concealed
- Agent: someone who can act on a patient’s behalf
- Third party: a third party whose consent is needed for an action. Example: Joe needs his father’s consent to access item in Joe's EHR describing his father's cardiac disease.

Spine Policy: Activate Spine-clinican Role

- A NHS-certified clinician can activate Spine-clinician role.
- canActivate(cli, Spine-clinician(ra, org, spcty)) :-

  ra.hasActivated(x, NHS-clinician-cert(org, cli, spcty, start, end)), // a similar rule has location ra for this premise

  canActivate(ra, Registration-authority()),

  no-main-role-active(cli),

  Current-time() in [start, end]
- canActivate(ra, Registration-authority()) :-

  NHS.hasActivated(x, NHS-registration-authority(ra, start, end)),

  Current-time() in [start, end]

Spine Policy: Author Can Read Item

- The author of an EHR item can always read it, provided the patient has given one-off consent, even if the patient has sealed the item.
- permits(cli, Read-spine-record-item(pat, id)) :-

  hasActivated(cli, Spine-clinician(ra, org, spcty)),

  hasActivated(x, One-off-consent(pat)),

  Get-spine-record-org(pat, id) = org,

  Get-spine-record-author(pat, id) = cli

Spine Policy: Treating Clinician Can Read Item

- A treating clinician can read item if patient has given one-off consent, item is not sealed by patient, and the item's subjects are permitted for the clinician's specialty.
- permits(cli, Read-spine-record-item(pat, id)) :-
  hasActivated(cli, Spine-clinician(ra, org, spcty)),
  hasActivated(x, One-off-consent(pat)),
  canActivate(cli, Treating-clinician(pat, org, spcty)),
  count-concealed-by-spine-patient(n, a, b),
  n = 0, a = (pat, id), b = (org, cli, spcty),
  Get-spine-record-subjects(pat, id) ⊆ Permitted-subjects(spcty)

Spine Policy: Activate Treating-clinician

- cli can activate Treating-clinician(pat, org, spcty) if
  - pat consented to treatment by cli, or
  - pat consented to treatment by workgroup containing cli, or
  - a clinician treating pat referred pat to cli, or
  - there is an emergency situation (audited later)

Patient Demographic System: Main Roles

- PDS-manager
  - Register and unregister people
- Patient
- Agent
- Professional-user
  - Clinicians, Caldicott guardians, etc.

Patient Demographic System: Activate Agent

- An agent can activate the Agent(pat) role if the agent is registered as a patient at the PDS, and the Spine confirms that he is an agent for pat.
- canActivate(ag, Agent(pat)) :-

  hasActivated(x, Register-patient(ag)),

  no-main-role-active(ag),

  Spine◊Spine.canActivate(ag, Agent(pat))

Local Health Organization: Staff Roles

- Clinician(spcty)
  - as in Spine
- Caldicott-guardian()
  - patient advocate and ombudsman. can give consent on behalf of a patient and, in exceptional cases, override a patient's decisions.
- HR-manager()
  - Register and unregister patients and staff
- Receptionist()
  - Register patients

Local Health Organization: Non-Staff Roles

- Patient
- Agent
- Ext-treating-clinician
    - external clinician who needs access to patient's local EPR
- Third-party

Local Health Organization Policy: Activate Ext-treating-clinician

- An clinician can activate Ext-treating-clinician if the referring clinician has activated the Consent-to-referral role and the clinician is certified by an RA certified by NHS.
- canActivate(cli, Ext-treating-clinician(pat, ra, org, spcty)) :-

  hasActivated(ref, Consent-to-referral(pat, ra, org, cli, spcty)),

  no-main-role-active(cli),

  ra◊ra.hasActivated(y, NHS-clinician-cert(org, cli, spcty, start, end)),

  canActivate(ra, Registration-authority())

Local Health Organization Policy: Deactivate Ext-treating-clinician

- Ext-treating-clinician is deactivated if patient (or patient's agent) cancels the referral by deactivating the referring clinician's Consent-to-referral.
- other-referral-consents(…) holds if, e.g., an agent of the patient has given consent for the referral.

isDeactivated(cli, Ext-treating-clinician(pat, ra, org, spcty)) :-
isDeactivated(x, Consent-to-referral(pat, ra, org, cli2 , spcty)),
other-referral-consents(0, x , pat, ra, org, cli , spcty)

Other Potential Application Domains

- Military: cooperation with
  - other armed services (Army, Navy, Air Force, Marines)
  - government agencies
  - highly trusted coalition partners
  - less trusted coalition partners
- Collaborative Engineering Design [Bhargava+, 2004]
  - multi-vendor bids for large engineering contracts
  - collaborators need to share designs and design knowledge, status of technologies, etc.

Other Potential Application Domains

- Supply Chain Management [Bhargava+, 2004]
  - For tight integration, a company must give its suppliers, customers, and its customers' customers (to increase their confidence in its ability to deliver) some access to its order entry, order status, sales forecast, and production planning systems.

Recap: Essential Features of Trust Management

- Each policy statement is associated with a principal, called its source or issuer.
- Each principal's policy specifies which sources it trusts for which kinds of statements, thereby delegating some authority to those sources.
- Policies may refer to domain-specific attributes of and relationships between principals, resources, and other objects.
- Example: AcmeHospital.allow(doc, Read(EPR(pat)) :-

```
      AMA.doctor(doc),
      pat.consentToTreatment(doc).
```