University of Twente
The Netherlands

A procrastinators' guide to compliance control

Why enforce today what you can check tomorrow

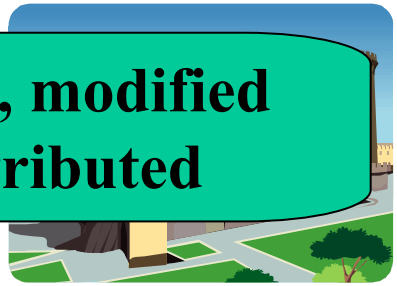
"I'll gladly proof tomorrow that I may eat this hamburger today"

Overview

- Current Situation
- Why A-Posteriori Compliance Control (APCC)
- Key Aspects
  - ✓ Trust
  - ✓ Obervability
- Generic APCC system
  - ✓ First step
  - ✓ Policy checkability.
  - ✓ Extentions

2007: the middle ages of compliance control(*)

- Confidential data
  - ✓ Medical records, RFID data, …
- Policy enforcement
  - ✓ Data should not be **disclosed** to unauthorized users
- How? Nowadays: DRM, Access Control
  - ✓ Preventative
  - ✓ No control outside the walls
  - ✓ One security domain; no …
- In case of more domains
  - ✓ Lawyers & Auditors



Also: used, modified and distributed

Detect and deter

Multi domain;
- different authorities
- different policies
- different policy enforcement systems

(*) Shamelessly copied from slides by Sandro

Why APCC

- Because you like procrastinating…
- Flexibility
  - ✓ Detect and deter
  - ✓ New settings
  - ✓ Policies; access if
    - Delete within a day
    - Do not work on competing projects
- Allows policy violation
  - ✓ Emergencies, unforeseen circumstances
  - ✓ Justified afterwards
  - ✓ Non-technical check

A-priori and APCC

- Combine APCC and a-priori checks
  - ✓ Trade off risk ↔ flexibility
  - ✓ Break the glass policy
    - Common in medical setting
    - Default a-priori
    - Emergency: break glass, switch to APCC
  - ✓ Partially validate
    - User is certified doctor
    - Detailed access rights checking postponed

ALFA

- Audit logic for a-posteriori compliance control
  - ✓ Logical policy language
  - ✓ Storable compliance proofs
  - ✓ Logging and auditing framework
- Key aspects
  - ✓ Action *may* be logged, *could* be checked
  - ✓ Misbehaviour possible
- What about ***trust***
  - ✓ How much does it deter misuse
  - ✓ Likelihood of getting caught
  - ✓ Ability to cause regret…

Trust, Accountability, Regret

- Trust Management (TM)
  - ✓ Shortly recall
  - ✓ Link to APCC
  - ✓ Role accountability and regret
- Main TM classes
  - ✓ Rule based TM
  - ✓ Reputation based TM

Rule Based Trust Management

- Example systems
  - ✓ Role based trust management (RT)
  - ✓ SDKI/SPKI
  - ✓ …
- Example scenario
  - ✓ “Student at accredited university gets discount”

    Shop.Discount ← AccBody.Univ.Student

    AccBody.Univ ← UT

    UT.student ← Alice

Rule Based Trust Management

- Distributed, Open
  - ✓ Each participant is authority, issues credentials
  - ✓ Participants can join, leave
- Delegation
  - ✓ entrust credentials of others
- Binary
  - ✓ User either fully trusted or not trusted
- Static trust level
  - ✓ No change based on actions of the user

Rule Based Trust Management

- They work because
  - ✓ No notion of risk so no policy violation
  - ✓ Users get defined rights; Alice will get the discount if she is entitled there is no notion of misusing the policy.
- They fail because
  - ✓ The policy may be wrong or not able to capture the intended meaning.
- Research issues:
  - ✓ Credential chain discovery
  - ✓ Trust Negotiation

Reputation, Recommendation Systems

- Example systems
  - ✓ E-bay transaction feedback system
  - ✓ Eigentrust
- Example scenario
  - ✓ “Users with good recommendations can buy a book”
  - ✓ Joint ordering action to get bulk discount
  - ✓ More participants means more savings
  - ✓ They do have to show up when the book arrives
  - ✓ Allow friends to join and/or recommend others to join
    - Alice joins, Bob does not join but does recommend Charlie.

Reputation Based Trust Management

- Main properties
  - ✓ Distributed, Open
    - Each participant is an authority
    - Issues its own recommendations/feedback.
  - ✓ Delegation
    - Place trust in the recommendations of others.
  - ✓ Multilevel and dynamic trust level
    - level of trust
    - actions influences the level of trust

Reputation Based Trust Management

- They work because:
  - ✓ Estimate likelihood of successful transaction
  - ✓ Give negative feedback if needed
- They fail because:
  - ✓ Past results give no guarantee for the future
- Research issues:
  - ✓ Trust metric definitions
  - ✓ Efficient and secure collection and exchange of trust related data.

Rule vs. Reputation based systems

- Analogies:
  - ✓ distributed systems;
    - information from different sources
    - combined to reach a decision
  - ✓ open
    - anyone can join or leave the system, issue credentials
    - value of credentials decided by others
- Differences:
  - ✓ Trust value domain
    - Yes/No vs. level of Trust
  - ✓ Role of Risk
  - ✓ ***Static vs. Dynamic***

Rule vs. Reputation based systems

- Static vs. Dynamic
  - ✓ Rule based
    - Alice being student not dependent on buying books
  - ✓ Reputation
    - Subjective probability favourable behaviour
    - Needs to reflect the actions
  - ✓ If Charlie does not collect book
    - His reputation will suffer
    - as will Bob's for recommending Charlie
  - ✓ ***However…will Charlie really care about this ?***

Regret, Punishment and accountability

- Will charlie *regret* not showing up?
  - ✓ only if the lost reputation was valuable to him...
- E.g. *trust* e-bay seller with high reputation because:
  - ✓ Past behaviour was good
  - ✓ Will want to keep high reputation
    - can cause regret.
- Can the trustee be held accountable, i.e.
  - ✓ Can misdoings be detected
  - ✓ How much regret
  - ✓ Cost to achieve
    - legal costs, cost to own reputation, etc.

Back to APCC

- Fundamental properties of APCC
  - ✓ Trust
  - ✓ Observability
- Trust in `the observer'
  - ✓ Authority doing the checking
  - ✓ Rule based system appropriate
  - ✓ Rules stating reputations may be interesting.
- Trust in users
  - ✓ Regret mechanism
    - evaluate and implement misuse deterrence

A basic APCC system

Actions $a \in A$, States $\sigma \in \Sigma$, Transition system $\rightarrow \subseteq \Sigma \times A \times \Sigma$, Start state $\sigma_0 \in \Sigma$.

A system trace/run: $tr = \sigma_0 \xrightarrow{a_1} \sigma_1 \ldots \xrightarrow{a_n} \sigma_n$.

Observables $(o \in) O$. Observable traces: OTr as traces but with O replacing A. Observation function $obs : A \rightarrow O$.

Observation lifted to traces by:
$obs(tr) = \sigma_0 \xrightarrow{obs(a_1)} \sigma_1 \ldots \xrightarrow{obs(a_n)} \sigma_n$.

(Models: states are visible but nature of actions may be hidden.)

Policies and auditing

Policies $\phi \in \mathcal{P} = A \times \Sigma \to \{ok, \ldots\}$.
Policy specifies whether action is allowed in a situation (optionally how bad if not ok).

Infringement $infr(\phi, n, tr)$ of policy ϕ at n in
$tr = \sigma_0 \overset{w}{\twoheadrightarrow} \sigma_{n-1} \overset{a_n}{\to} \sigma_n \overset{w'}{\twoheadrightarrow} \sigma_m$ when $\phi(a_n, \sigma_{n-1}) \neq ok$.

Audit $\mathcal{A} : OTr \to \{ok, \ldots\}$.
Audit specifies whether an observed sequence is compliant (optionally how bad if not ok).

Audit $\mathcal{A}$ marks trace tr iff $\mathcal{A}(obs(tr)) \neq ok$

$\mathcal{A}$ is ϕ-correct: $\mathcal{A}$ marks trace $tr \Rightarrow \exists n : infr(\phi, n, tr)$.
$\mathcal{A}$ is ϕ-complete: $\mathcal{A}$ marks trace $tr \Leftarrow \exists n : infr(\phi, n, tr)$.
$\mathcal{A}$ fully audits ϕ: $\mathcal{A}$ is ϕ-correct and complete.

ϕ is fully APPC checkable if $\exists \mathcal{A} : \mathcal{A}$ fully audits ϕ.

Possible extensions and future work

- Generalize system
  - ✓Different notions of observer
- Map existing APCC approaches
- Probabilistic model
  - ✓Likelihood detection misbehaviour
  - ✓Risk assessment (e.g. before instating policy)
- Trust feedback
  - ✓Reputation based on audit results

Conclusion…

- Work in progress
  (Comments & Ideas welcome)

or:

I will gladly give you a conclusion tomorrow…