MuAC

Access Control Language for Mutual Benefits

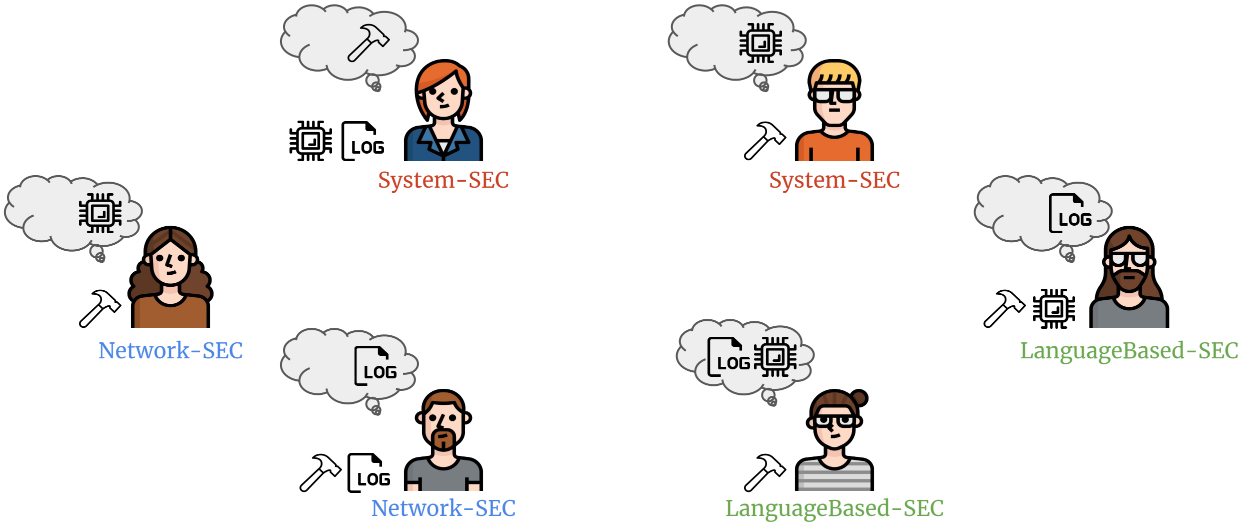
ITASEC 2020

Lorenzo Ceragioli (Università di Pisa)
Pierpaolo Degano (Università di Pisa)
Letterio Galletta (Scuola IMT Alti Studi Lucca)

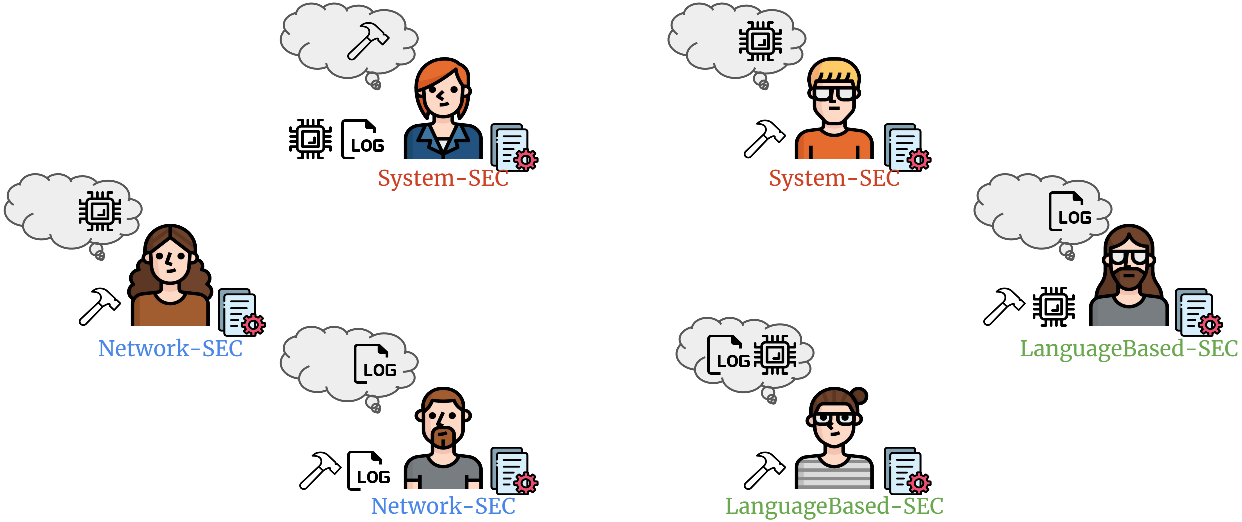
Access Control - Based on …

- Some requester quality (attribute, trust, roles)
- Some relationship between owner and requester
- Something that the owner will have in return?

Context: collaboration… with an eye to mutuality



Context: collaboration… with an eye to mutuality



Policy - What to ask in return

You can ask something

- for you or for someone else
- from the requester or from someone else

Policy - What to ask in return

You can ask something

- for you or someone else
- from the requester or someone else

▣ - if one of your colleagues shares ▣ with me

LOG 🔨 - if you share LOG or 🔨 with a colleague of mine

🔨 - with every colleague of mine

MuAC Language

U : Me, Subject, user variables u, u’ …

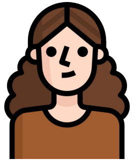
R : Resource, resource variables r, r’ …

p : atomic predicates p, q, p’, q’ …

$$\Phi \ni \phi ::= p(U) \mid p(R) \mid Allows(U, R, U) \mid \phi, \phi$$

$$\sigma : U \longrightarrow 2^{\Phi}$$

Direct Exchange Policies

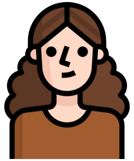


Network-SEC

tool(Resource), Allows(Me, r, Subject), computational-power(r)

She is asking for a direct exchange of computation-power for tools

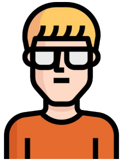
Direct Exchange Policies



Network-SEC

tool(Resource), Allows(Me, r, Subject), computational-power(r)

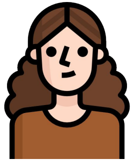
She is asking for a direct exchange of computation-power for tools



Wants to use Alice's tools

System-SEC

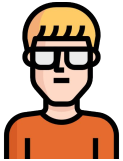
Direct Exchange Policies



Network-SEC

tool(Resource), Allows(Me, r, Subject), computational-power(r)

She is asking for a direct exchange of computation-power for tools



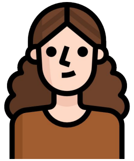
Wants to use Alice's tools

System-SEC

Network-SEC(Subject), computational-power(Resource)

He allows Network-SEC members access computation-power

Direct Exchange Policies

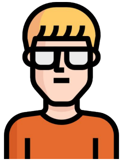


OK!

Network-SEC

tool(Resource), Allows(Me, r, Subject), computational-power(r)

She is asking for a direct exchange of computation-power for tools



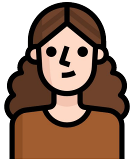
Wants to use Alice's tools

System-SEC

Network-SEC(Subject), computational-power(Resource)

He allows Network-SEC members access computation-power

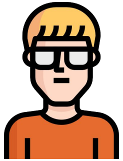
Direct Exchange Policies



Network-SEC

tool(Resource), Allows(Me, r, Subject), computational-power(r)

She is asking for a direct exchange of computation-power for tools



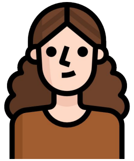
Wants to use Alice's tools

System-SEC

Network-SEC(Subject), computational-power(Resource), Allows(Me, r, Subject), tool(r)

He allows Network-SEC members to access computation-power if they allow him access tools

Direct Exchange Policies

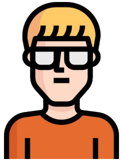


OK!

Network-SEC

tool(Resource), Allows(Me, r, Subject), computational-power(r)

She is asking for a direct exchange of computation-power for tools



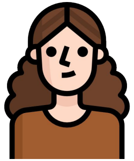
Wants to use Alice’s tools

System-SEC

Network-SEC(Subject), computational-power(Resource), Allows(Me, r, Subject), tool(r)

He allows Network-SEC members to access computation-power if they allow him access tools

Group-related Policies

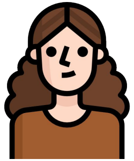


Network-SEC

computational-power(Resource), System-SEC(u), Allows(Me, r, u), log(r), System-SEC(Subject)

She asks someone in System-SEC group to give her logs for her computation-power

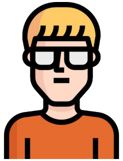
Group-related Policies



Network-SEC

computational-power(Resource), System-SEC(u),
Allows(Me, r, u), log(r), System-SEC(Subject)

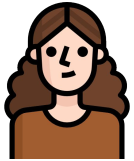
She asks someone in System-SEC group to give her logs for her computation-power



Wants to use Alice's computational power

System-SEC

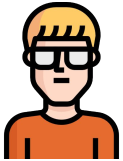
Group-related Policies



Network-SEC

computational-power(Resource), System-SEC(u),
Allows(Me, r, u), log(r), System-SEC(Subject)

She asks someone in System-SEC group to give her logs for her computation-power



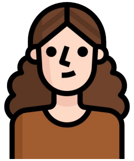
Wants to use Alice's computational power

System-SEC

log(Resource), Network-SEC(u'), System-SEC(u),
Allows(u', r, u), tool(r), Network-SEC(Subject)

He asks for someone of Network-SEC group to give tools to someone in his group for his logs

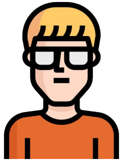
Group-related Policies



Network-SEC

computational-power(Resource), System-SEC(u),
Allows(Me, r, u), log(r), System-SEC(Subject)

She asks someone in System-SEC group to give her logs for her computation-power

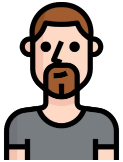


Wants to use Alice’s computational power

System-SEC

log(Resource), Network-SEC(u’), System-SEC(u),
Allows(u’, r, u), tool(r), Network-SEC(Subject)

He asks for someone of Network-SEC group to give tools to someone in his group for his logs

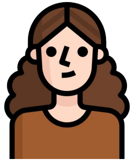


Network-SEC

tool(Resource), System-SEC(u),
Allows(Me, r, u), log(r), System-SEC(Subject)

He asks someone in System-SEC group to give him logs for his tools

Group-related Policies

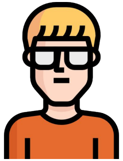


OK!

Network-SEC

computational-power(Resource), System-SEC(u),
Allows(Me, r, u), log(r), System-SEC(Subject)

She asks someone in System-SEC group to give her logs for her computation-power

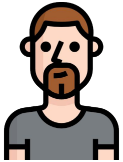


Wants to use Alice's computational power

System-SEC

log(Resource), Network-SEC(u'), System-SEC(u),
Allows(u', r, u), tool(r), Network-SEC(Subject)

He asks for someone of Network-SEC group to give tools to someone in his group for his logs

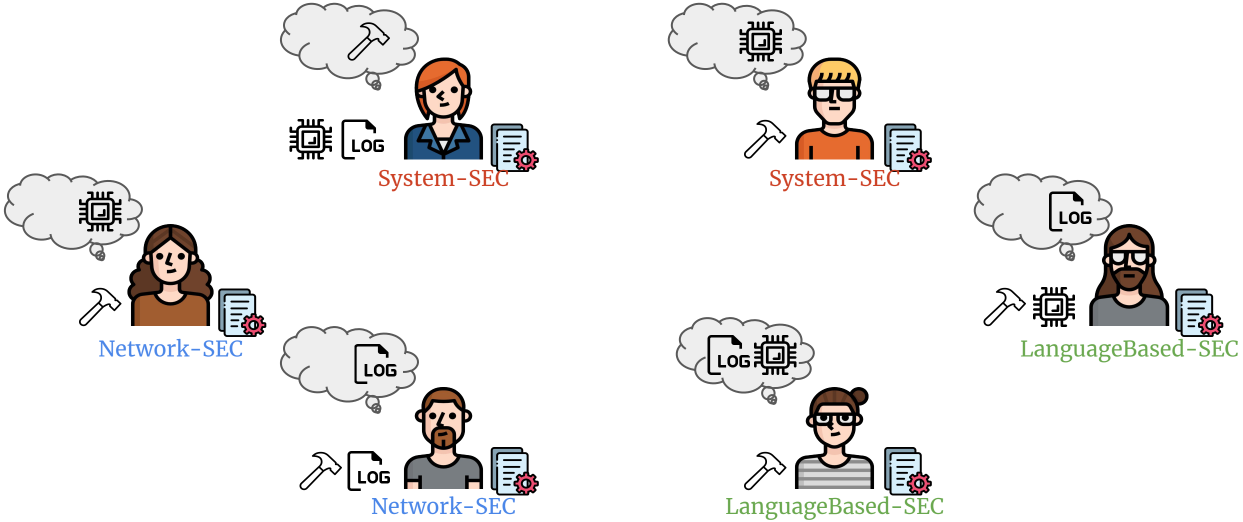


Network-SEC

tool(Resource), System-SEC(u),
Allows(Me, r, u), log(r), System-SEC(Subject)

He asks someone in System-SEC group to give him logs for his tools

Context - *Every user defines his policy in isolation*



To evaluate a request

- check owner policy
- check recursively other policies that affect the result (Subject, u, u' …)

We rely on

Propositional Contract Logic

Propositional Contract Logic (PCL)

"A Calculus of Contracting Processes" by Bartoletti & Zunino - LICS 2010

Intuitionistic propositional logic with ***Contractual Implication***

$p \twoheadrightarrow q$: a promise that "q will be satisfied if also p is"

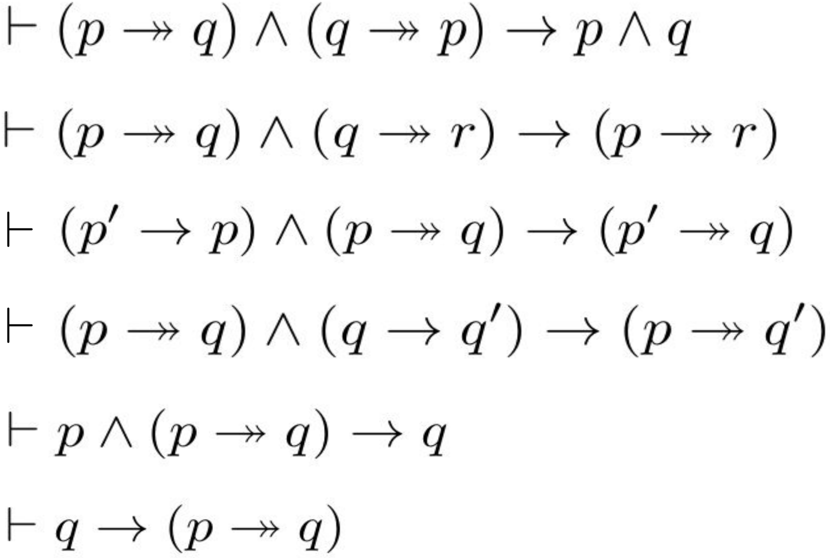
$$\vdash (p \twoheadrightarrow q) \wedge (q \twoheadrightarrow p) \rightarrow p \wedge q$$

Decidable (deduction is PSPACE complete)

The theorem prover with acceptable performance for common examples

Propositional Contract Logic (PCL)

“A Calculus of Contracting Processes” by Bartoletti & Zunino - Symposium on Logic in Computer Science, 2010



$$\vdash (p \twoheadrightarrow q) \land (q \twoheadrightarrow p) \to p \land q$$

$$\vdash (p \twoheadrightarrow q) \land (q \twoheadrightarrow r) \to (p \twoheadrightarrow r)$$

$$\vdash (p' \to p) \land (p \twoheadrightarrow q) \to (p' \twoheadrightarrow q)$$

$$\vdash (p \twoheadrightarrow q) \land (q \to q') \to (p \twoheadrightarrow q')$$

$$\vdash p \land (p \twoheadrightarrow q) \to q$$

$$\vdash q \to (p \twoheadrightarrow q)$$

MuAC Language Semantics

Rules ϕ interpreted as sets of promises

$$\text{Allows}(\text{Alice}, \text{log1.txt}, \text{Bob}), \ldots\ \text{Allows}(\text{Bob}, \text{tool1.sh}, \text{Carl}) \twoheadrightarrow \text{Allow}(\overset{\text{Subject}}{\overset{\downarrow}{\text{Bob}}}, \underset{\text{Resource}}{\underset{\uparrow}{\text{log2.txt}}}, \overset{\text{Me}}{\overset{\downarrow}{\text{Alice}}})$$

From configuration σ to PCL theory Γ

Access request **asks(Bob, log2.txt)** allowed iff

$$\Gamma \vdash \text{Allows}(\text{Bob}, \text{log2.txt}, \text{Alice})$$

where Alice is the owner of log2.txt

Future Work: still a lot to do!

Efficient algorithm for access control decision

- we only have a proof-of-concept algorithm
- there are implicit quantifications in rules (but not in PCL)
- maybe we can use DataLog
- distributed implementation

Future Work: still a lot to do!

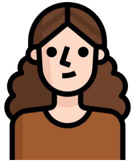
Trust and usage control - dealing with malicious users

- trust is assumed between all users
- time is not considered
- Eve may grab what she wants and run (free-rider)
    - Declare to share all she have for nothing
    - Make a copy of what she wants as soon as possible
    - Leave the system before someone can actually access her resources

Future Work: still a lot to do!

Language extension

- deny rules
  - conflicts resolution
- not-Allows as condition
  - ***Conflict-of-Interest*** policies
  - ***Embargo*** policies



Network-SEC

logs(Resource), not-Allows(u, r, Subject), LanguageBased-SEC(u)

To access her logs, she asks the requester to share nothing with LanguageBased-SEC members