Secure Cloud-Assisted Smart Cars: Dynamic Groups and ABAC

Maanak Gupta, James Benson, Farhan Patwa and Ravi Sandhu

L13-2

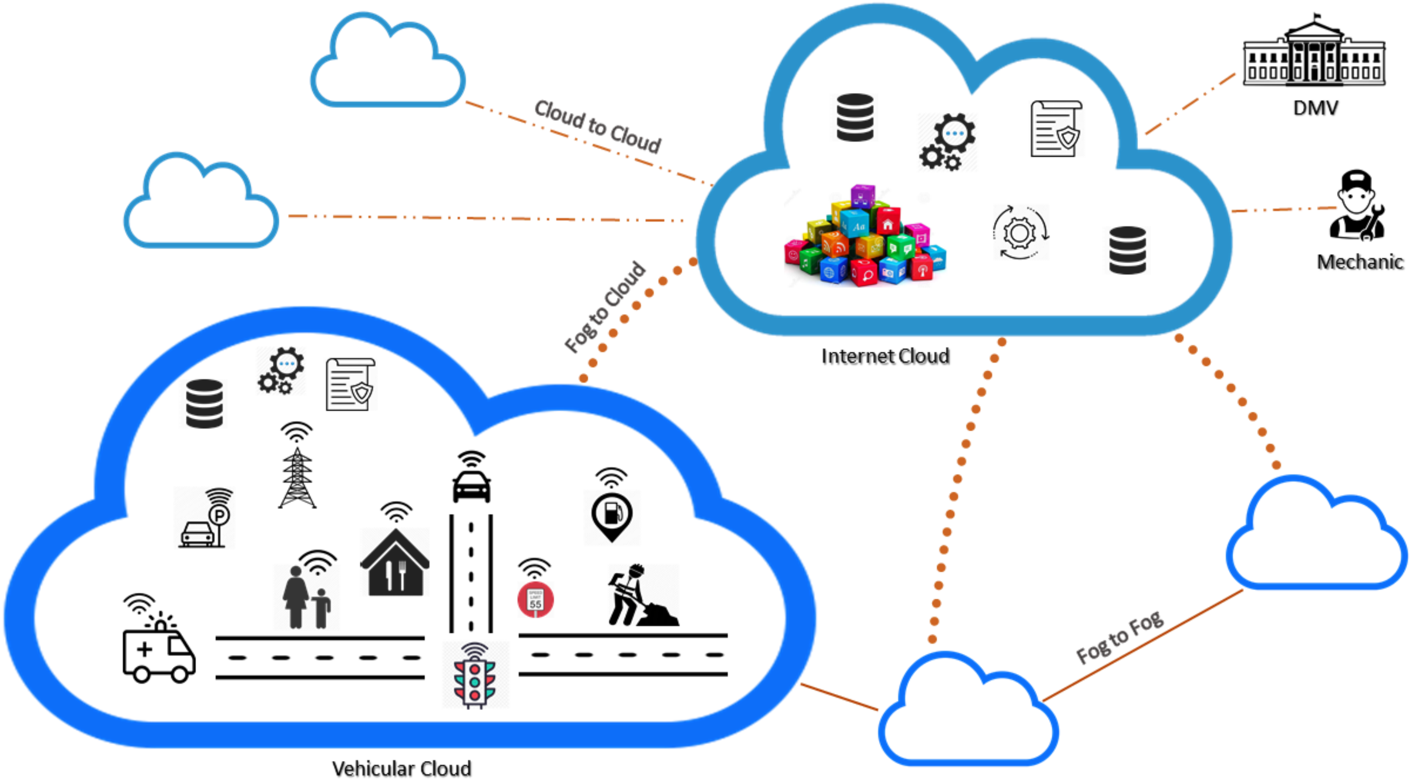
CS6393

Spring 2020

© Maanak Gupta

UTSA Computer Science

Connected Cars Ecosystem



Scope of Contribution

- Contribution
  - Propose formalized ABAC model for cloud assisted applications.
  - Dynamic groups and user preferences.
  - Implementation of the model in AWS.

- Scope
  - Single Central Cloud
  - No direct access and physical tampering
  - Communication Channel is encrypted.
  - Data in Cloud is secure
  - In-vehicle security not considered

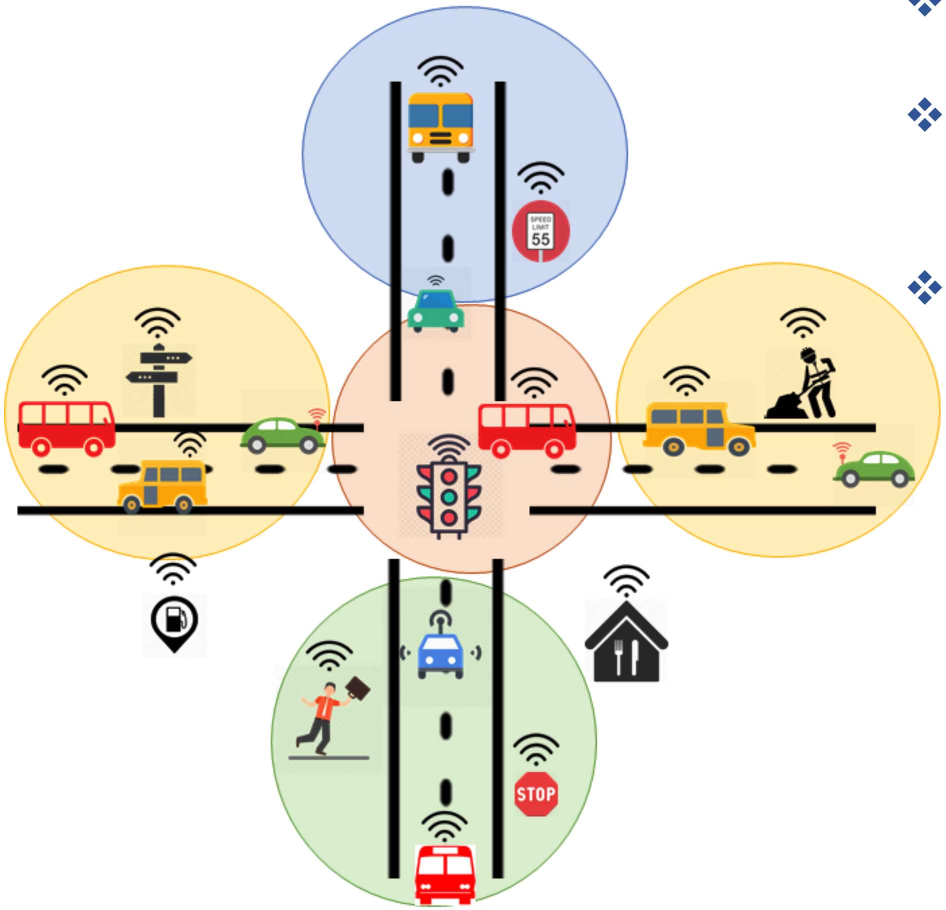
Attribute Based Access Control

- ABAC: Decision based on the attributes of entities
- Attributes are name value pairs: age (Alice) → 29
- Core entities in ABAC include:
  - Users
  - Objects
  - Environment or Context
  - Operations

  Attributes
- Authorization Policies: determine rights just in time
  - retrieve attributes of relevant entities in request
- Enhance flexibility and fine grained access control

© Maanak Gupta

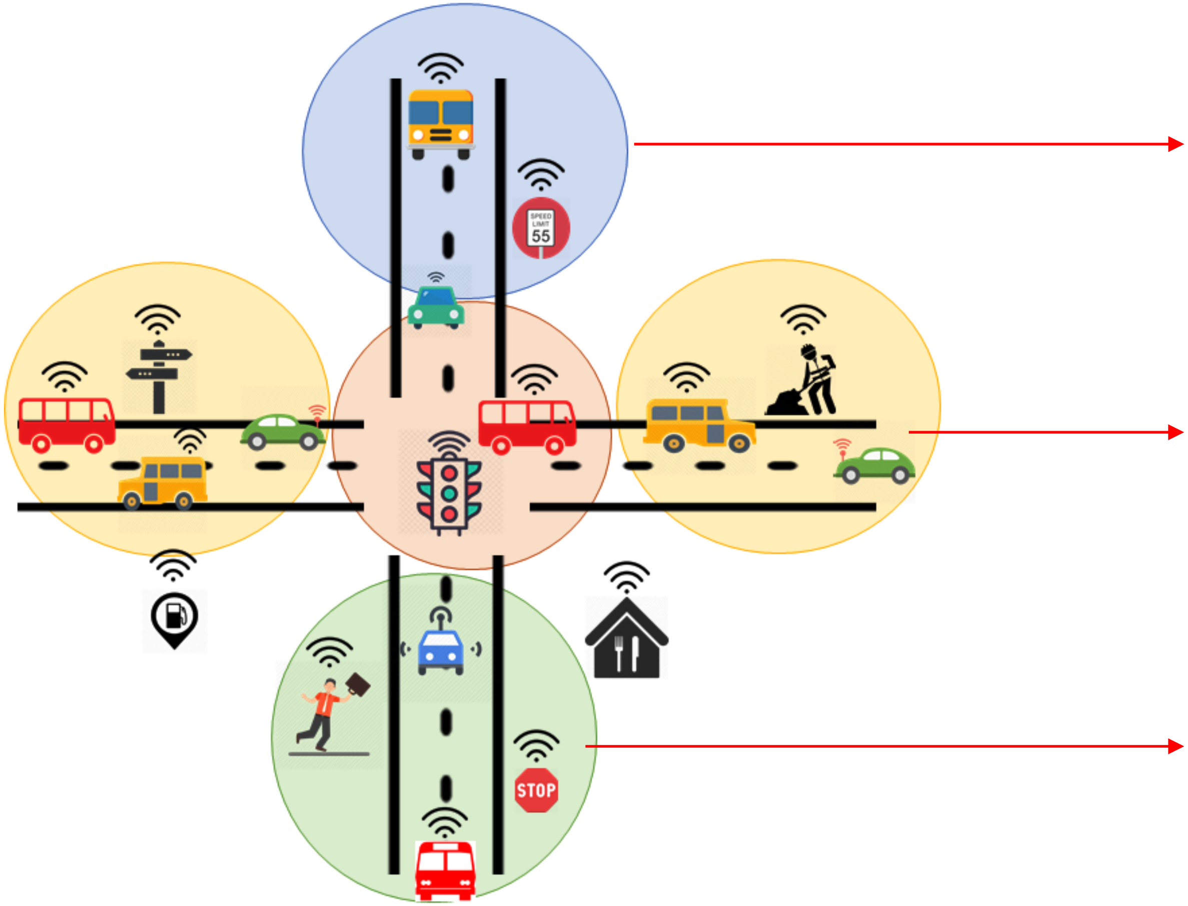
Location Groups



- Categorizing wide locations into smaller groups.
- Vehicles dynamically become member based on current GPS, vehicle-type or individual user preferences.
- Ensure relevance of alerts and notifications

© Maanak Gupta

Attributes and Alerts



Speed Limit: 50 mph
Deer Threat: ON
Ice on Road: NO

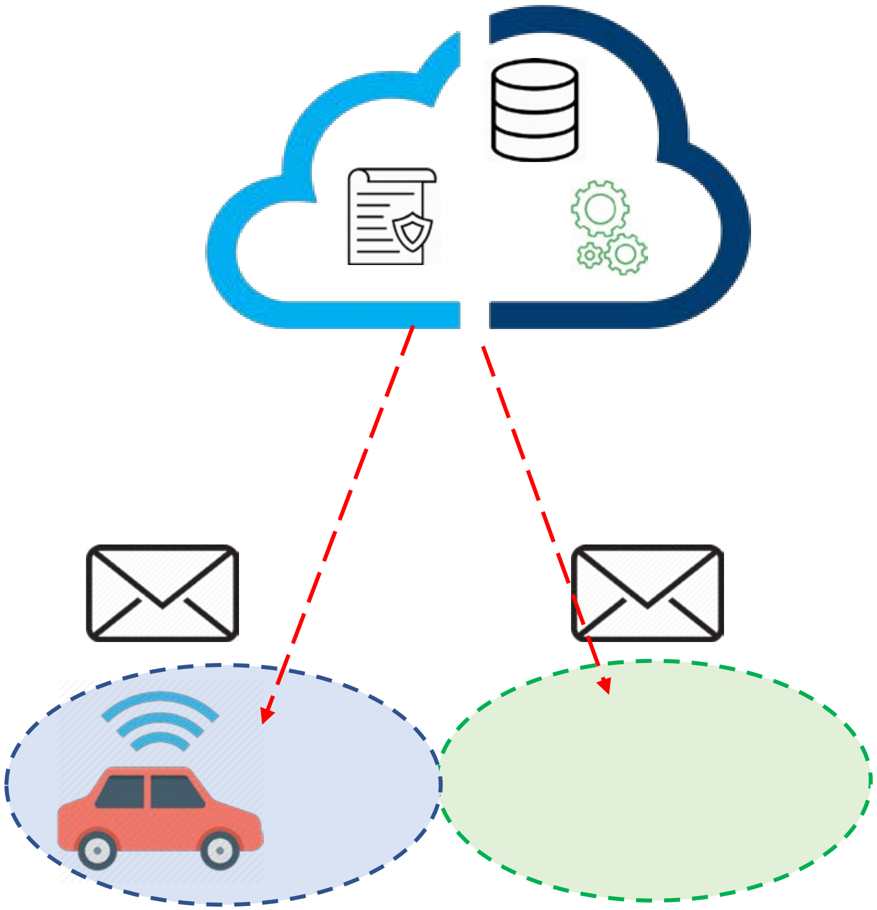
Speed Limit: 30 mph
Flood Warning: ON
Road Work: ON

Speed Limit: 20 mph
School Zone: ON
Amber Alert: ABC123

Vehicles move and are assigned to different groups and inherit their attributes/alerts.

© Maanak Gupta

Using Location Groups



Administrative Questions:

- How the attributes or alerts of groups are updated?
- How are moving entities assigned to groups?
- How groups hierarchy is created?

Operational Questions:

- How attributes and groups are used to provide security?
- How user privacy preferences are considered?

{"state": {"reported": {"Latitude": "29.4769353",
"Longitude":"-98.5018237"}}}

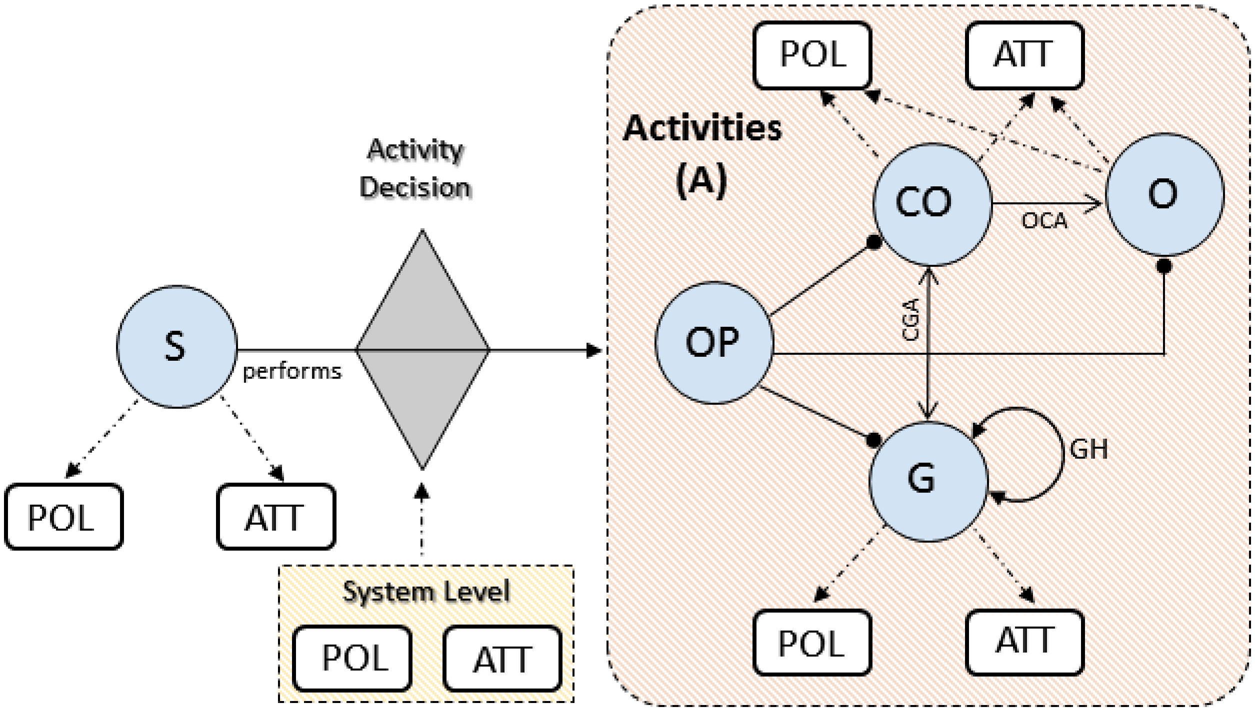
Speed Limit: 50 mph
Deer Threat: ON
Ice on Road: NO

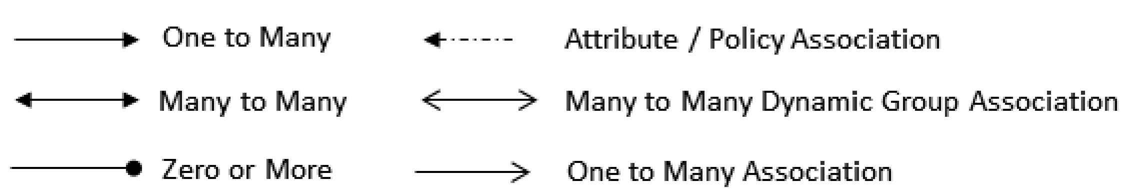
Speed Limit: 30 mph
Flood Warning: ON
Road Work: ON

Reported MQTT message

© Maanak Gupta

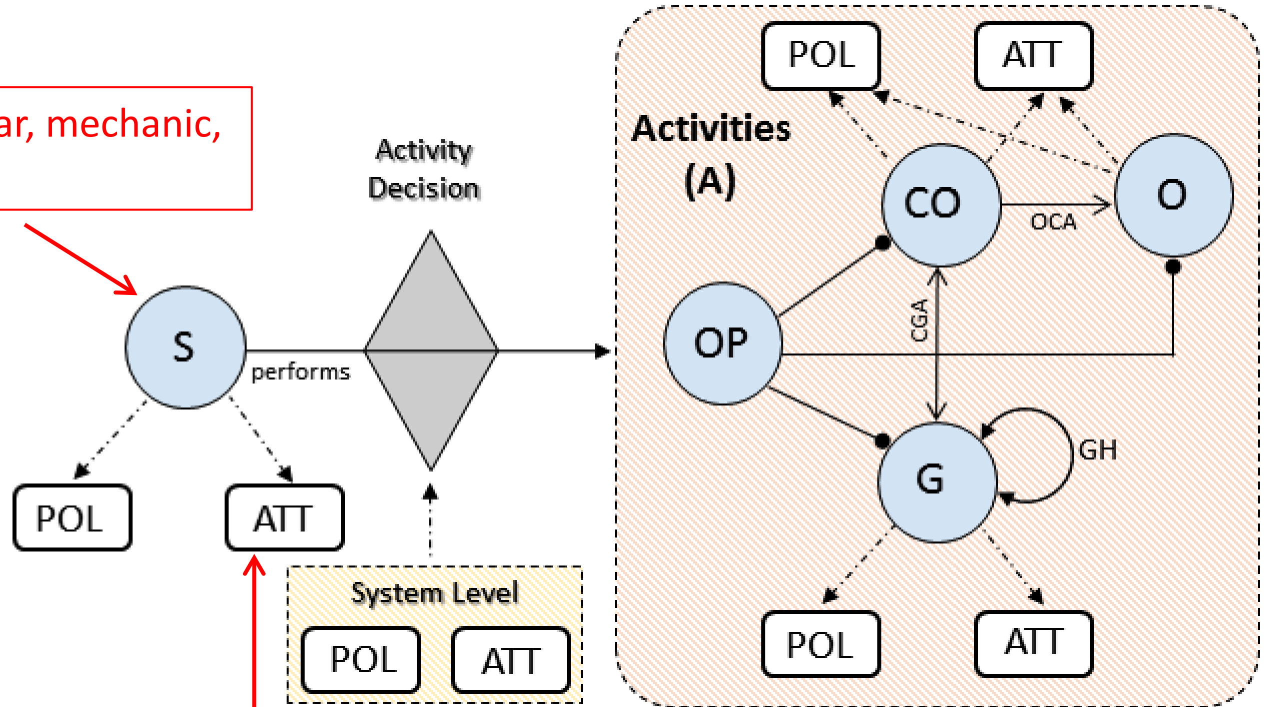
CV-ABAC$_G$ Model





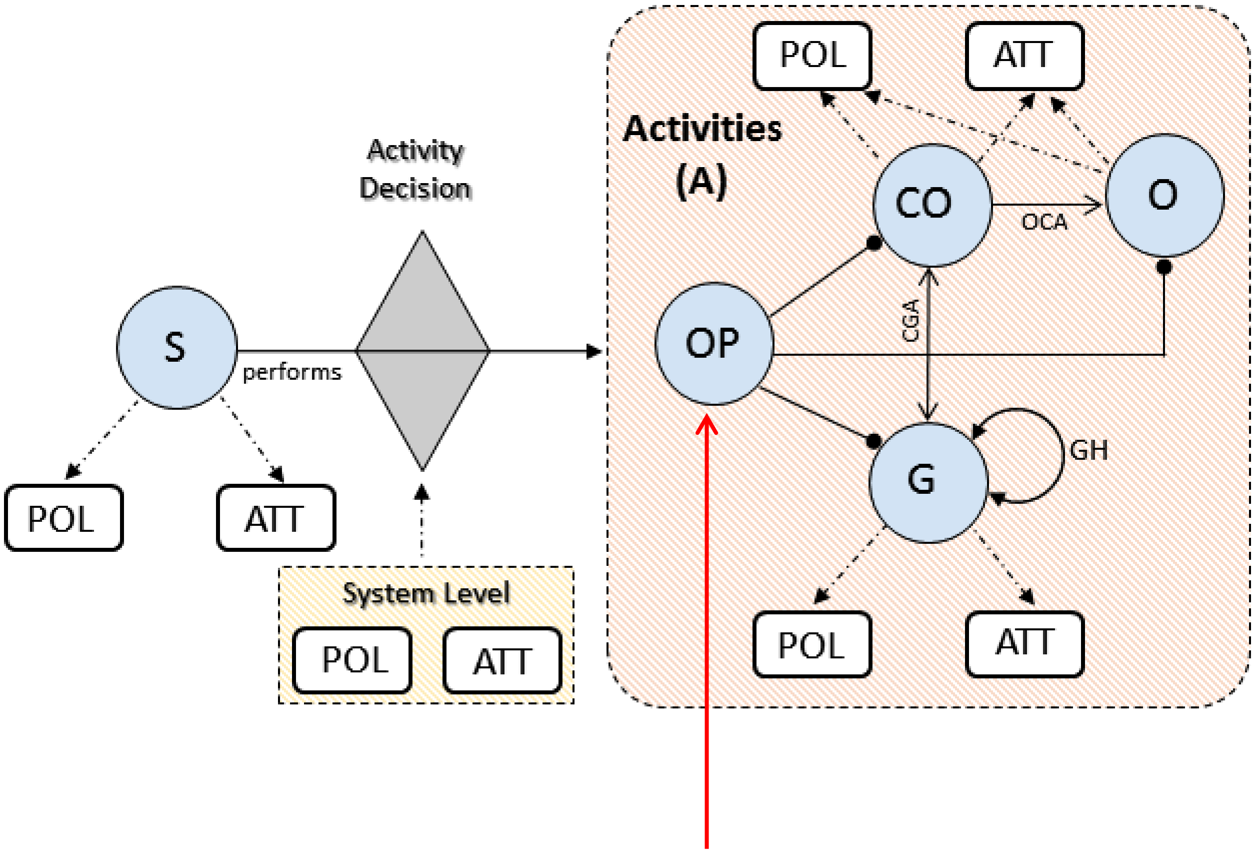
Model Components

user, sensor, car, mechanic, restaurant



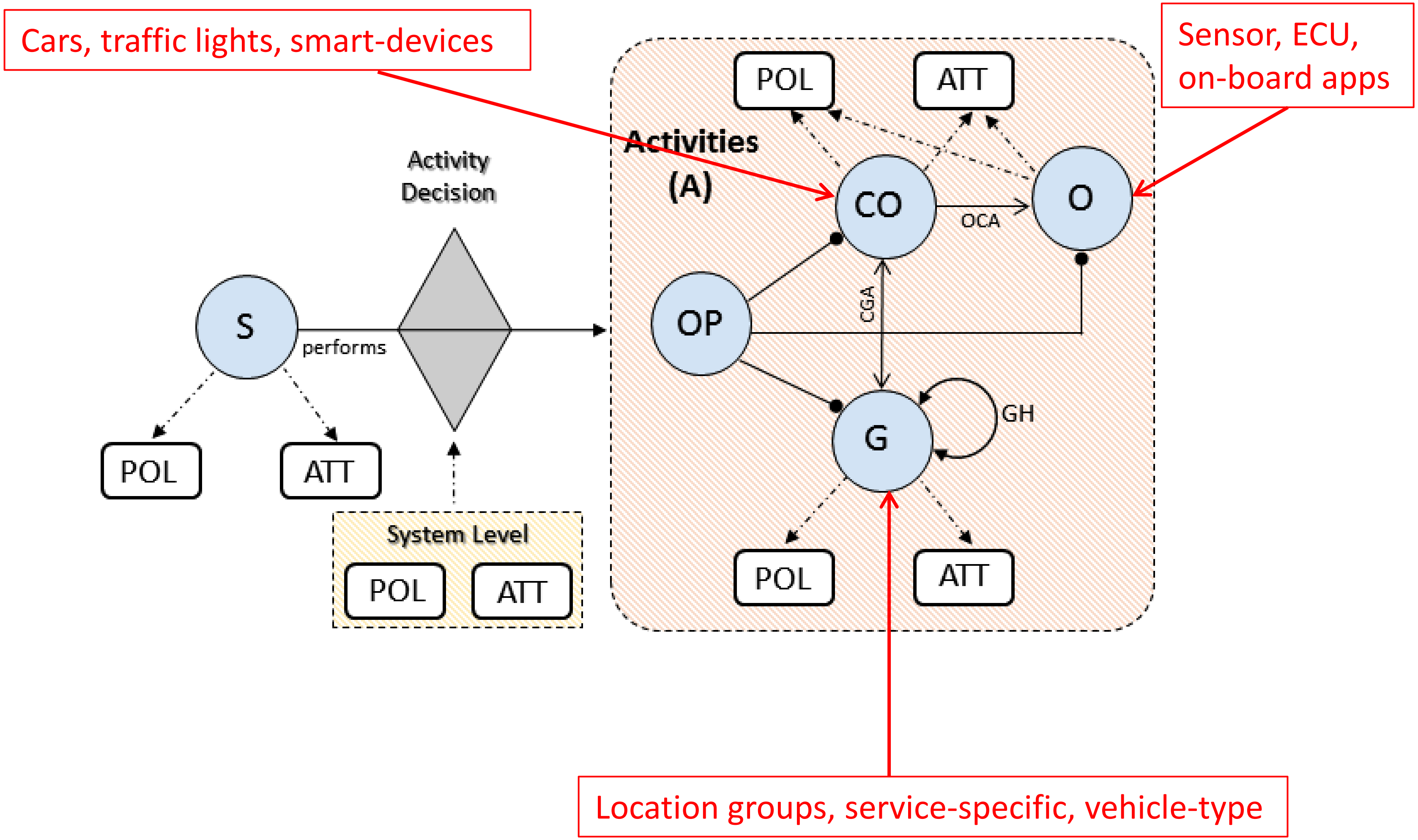
{ location, size, IP, direction, speed, VIN, cuisine-type}

Model Components



{ read, write, control, notify, administrative actions }

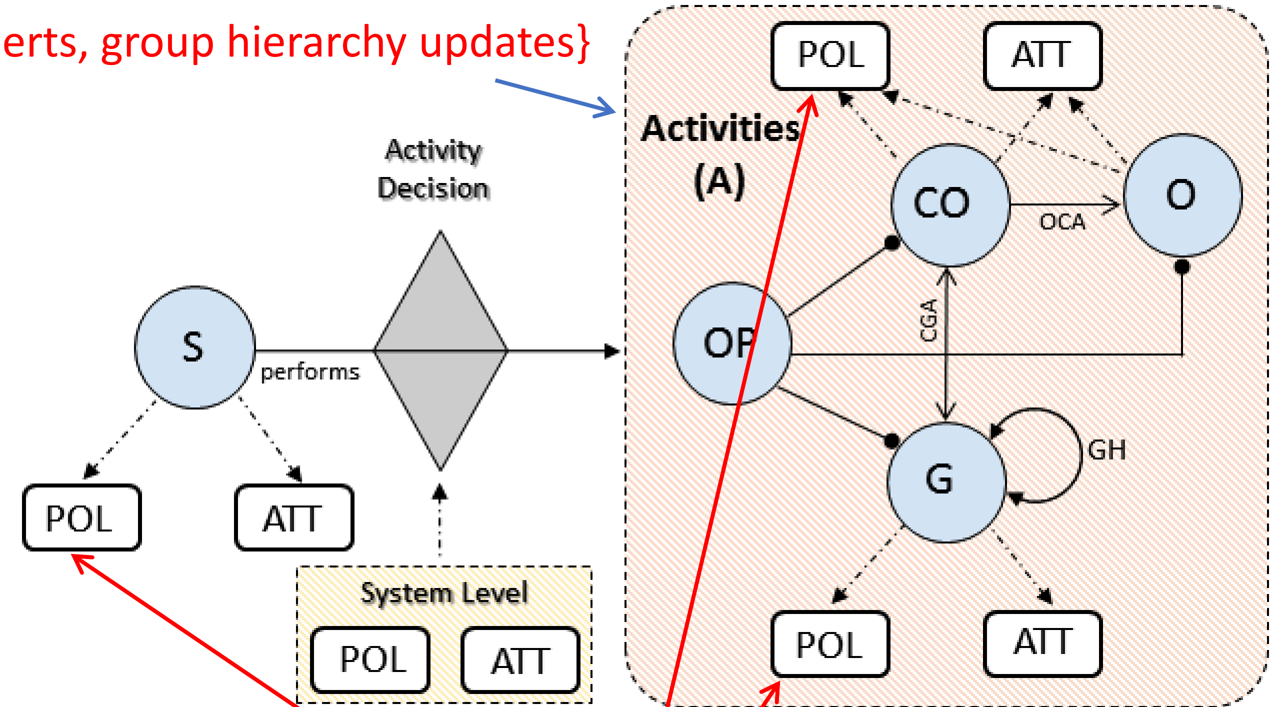
Model Components



Model Components

Operational and Administrative Activities

{notification, alerts, group hierarchy updates}



System Wide Policies

Individualized Privacy Policies

© Maanak Gupta

Formal Specification

Basic Sets and Functions

- S, CO, O, G, OP are finite sets of sources, clustered objects, objects, groups and operations respectively [blue circles in Figure 4].
- A is a finite set of activities which can be performed in system.
- ATT is a finite set of attributes associated with S, CO, O, G and system-wide. Attribute Function
- For each attribute att in ATT, Range(att) is a finite set of atomic values.
- attType: ATT = {set, atomic}, defines attributes to be set or atomic valued. Attribute Type
- Each attribute att in ATT maps entities in S, CO, O, G to attribute values. Formally,

$$\mathrm{att} : S \cup CO \cup O \cup G \cup \{\text{system-wide}\} \rightarrow \begin{cases} \mathrm{Range(att)} \cup \{\perp\} & \text{if attType(att) = atomic} \\ 2^{\mathrm{Range(att)}} & \text{if attType(att) = set} \end{cases}$$

- POL is a finite set of authorization policies associated with individual S, CO, O, G.
- directG : CO $\rightarrow$ G, mapping each clustered object to a system group, equivalently CGA $\subseteq$ CO $\times$ G.
- parentCO : O $\rightarrow$ CO, mapping each object to a clustered object, equivalently OCA $\subseteq$ O $\times$ CO.
- GH $\subseteq$ G $\times$ G, a partial order relation $\geq_g$ on G. Equivalently, parentG : G $\rightarrow 2^G$, mapping group to a set of parent groups in hierarchy.

Group Hierarchy

Attribute Mapping

Formal Specification

Effective Attributes of Groups, Clustered Objects and Objects (Derived Functions)

– For each attribute att in ATT such that attType(att) = set:

- $\mathrm{effG}_{\mathrm{att}} : G \rightarrow 2^{\mathrm{Range(att)}}$, defined as $\mathrm{effG}_{\mathrm{att}}(g_i) = \mathrm{att}(g_i) \cup (\bigcup_{g \in \{g_j | g_i \geq_g g_j\}} \mathrm{effG}_{\mathrm{att}}(g))$.
- $\mathrm{effCO}_{\mathrm{att}} : CO \rightarrow 2^{\mathrm{Range(att)}}$, defined as $\mathrm{effCO}_{\mathrm{att}}(co) = \mathrm{att}(co) \cup \mathrm{effG}_{\mathrm{att}}(\mathrm{directG}(co))$.
- $\mathrm{effO}_{\mathrm{att}} : O \rightarrow 2^{\mathrm{Range(att)}}$, defined as $\mathrm{effO}_{\mathrm{att}}(o) = \mathrm{att}(o) \cup \mathrm{effCO}_{\mathrm{att}}(\mathrm{parentCO}(o))$.

– For each attribute att in ATT such that attType(att) = atomic:

- $\mathrm{effG}_{\mathrm{att}} : G \rightarrow \mathrm{Range(att)} \cup \{\perp\}$, defined as $\mathrm{effG}_{\mathrm{att}}(g_i) = \begin{cases} \mathrm{att}(g_i) & \text{if } \forall g' \in \mathrm{parentG}(g_i).\ \mathrm{effG}_{\mathrm{att}}(g') = \perp \\ \mathrm{effG}_{\mathrm{att}}(g') & \text{if } \exists\, \mathrm{parentG}(g_i).\ \mathrm{effG}_{\mathrm{att}}(\mathrm{parentG}(g_i)) \neq \perp \text{ then select} \\ & \text{parent } g' \text{ with } \mathrm{effG}_{\mathrm{att}}(g') \neq \perp \text{ updated most recently.} \end{cases}$
- $\mathrm{effCO}_{\mathrm{att}} : CO \rightarrow \mathrm{Range(att)} \cup \{\perp\}$, defined as $\mathrm{effCO}_{\mathrm{att}}(co) = \begin{cases} \mathrm{att}(co) & \text{if } \mathrm{effG}_{\mathrm{att}}(\mathrm{directG}(co)) = \perp \\ \mathrm{effG}_{\mathrm{att}}(\mathrm{directG}(co)) & \text{otherwise} \end{cases}$
- $\mathrm{effO}_{\mathrm{att}} : O \rightarrow \mathrm{Range(att)} \cup \{\perp\}$, defined as $\mathrm{effO}_{\mathrm{att}}(o) = \begin{cases} \mathrm{att}(o) & \text{if } \mathrm{effCO}_{\mathrm{att}}(\mathrm{parentCO}(o)) = \perp \\ \mathrm{effCO}_{\mathrm{att}}(\mathrm{parentCO}(o)) & \text{otherwise} \end{cases}$

Attributes more Dynamic

Attributes Inheritance

Policy Language

Authorization Functions (Policies)

– Authorization Function: For each op ∈ OP, $Auth_{op}$(s : S, ob : CO ∪ O ∪ G) is a propositional logic formula returning true or false, which is defined using the following policy language:

- $\alpha ::= \alpha \wedge \alpha \mid \alpha \vee \alpha \mid (\alpha) \mid \neg\alpha \mid \exists\, x \in set.\alpha \mid \forall\, x \in set.\alpha \mid set \vartriangle set \mid atomic \in set \mid atomic \notin set$
- $\vartriangle ::= \subset \mid \subseteq \mid \not\subseteq \mid \cap \mid \cup$
- set ::= $eff_{att}(i)$ | att(i) — for att ∈ ATT, i ∈ S ∪ CO ∪ O ∪ G ∪ {system-wide}, attType(att) = set
- atomic ::= $eff_{att}(i)$ | att(i) | value — for att ∈ ATT, i ∈ S ∪ CO ∪ O ∪ G ∪ {system-wide}, attType(att) = atomic

❖ Administrators in the police department can send alert to location-groups in city limits.

$Auth_{alert}$(u:U, g:G) :: dept (u) Police ∧ parent-city(g) = Austin ∧ Austin ∈ jursidiction (u).

❖ Only mechanic in the technician department from Toyota-X dealership must be able to read sensor in Camry LE. Further, this operation must be done between time 9 am to 6 pm.

$Auth_{read}$(u:U, co:CO) :: role (u) Technician ∧ employer(u) = Toyota-X ∧ make (co) = Toyota ∧ model(co) = Camry LE ∧ operation_time(u) ∈ {9am,10,11…6pm}

© Maanak Gupta

Activity Authorization Decision

Authorization Decision

– A source $s \in S$ is allowed to perform an activity $a \in A$, stated as Authorization$(a : A, s : S)$, if the required policies needed to allow the activity are included and evaluated to make final decision. These multi-layer policies must be evaluated for individual operations ($op_i \in OP$) to be performed by source $s \in S$ on relevant objects ($x_i \in CO \cup O \cup G$).

$$\boxed{\text{Formally, Authorization}(a : A, s : S) \Rightarrow \text{Auth}_{op_1}(s : S, x_1), \text{Auth}_{op_2}(s : S, x_2), \ldots\ldots\ldots\ldots, \text{Auth}_{op_n}(s : S, x_3)}$$

Evaluate all relevant policies to make a decision

A restaurant in group A must be allowed to send notifications to all vehicles in location group A and group B.

System defined

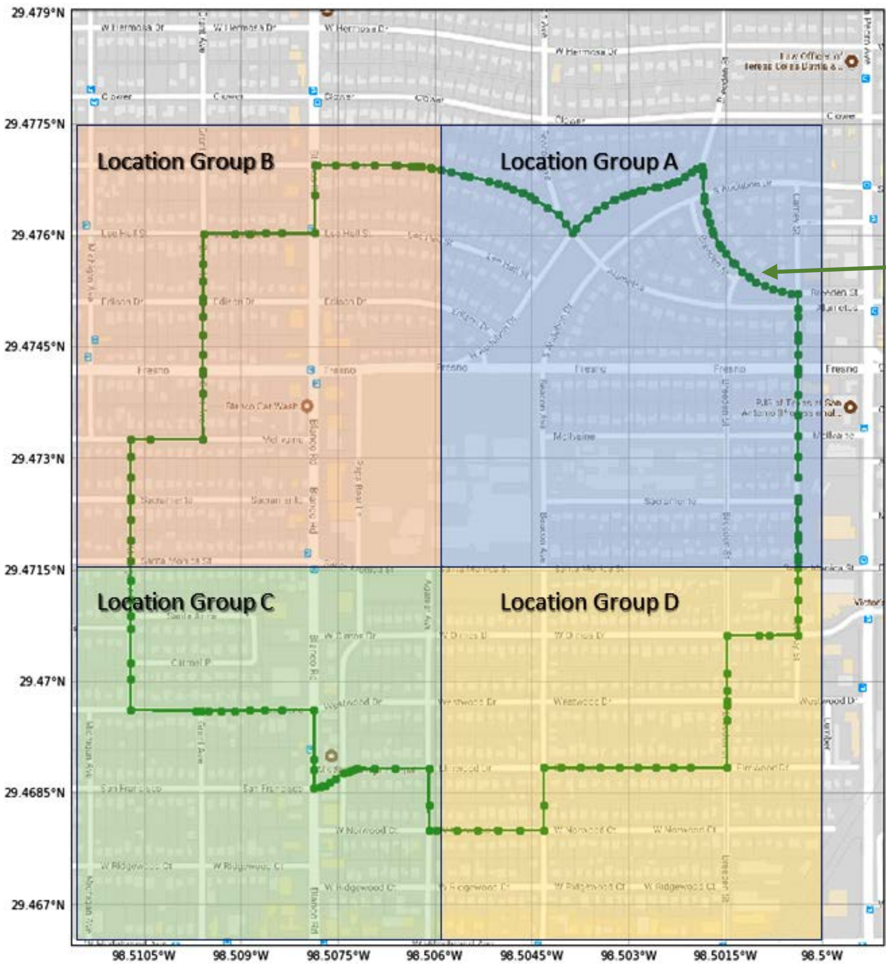
DECISION

User Preference

I only want notifications from Cheesecake factory.

Implementation in Amazon Web Services (AWS)

Vehicles and Groups



4 Location Groups (static demarcation)

Vehicles movement (coordinates generated using Google API)

```
('Received new coordinates from:', 'Vehicle-1')
Sun May 27 02:56:30 2018
  Location A
     Car-A : [u'Vehicle-1', u'Vehicle-2']
     Bus-A : []
  Location B
     Car-B : []
     Bus-B : [u'Vehicle-6']
  Location C
     Car-C : [u'Vehicle-3', u'Vehicle-4']
     Bus-C : []
  Location D
     Car-D : []
     Bus-D : [u'Vehicle-5']
```

Snapshot (table keeps changing)

Implemented Policies

- Administrative Policy
  - Road side motion sensor with [id = 1] and current GPS in group [Location-A] can only [modify] attribute [Deer Threat] to value [ON, OFF] for group [Location-A].

- Operational Policy

Restaurant Notification Use Case

System Defined Policy

- A restaurant located within group [Location-A] can only [send notifications] to members of groups [Location-A, Location-B].

User Preferences

- Send notifications only between [7 pm to 9 pm] only on [Wednesdays].

© Maanak Gupta

Performance Metrics

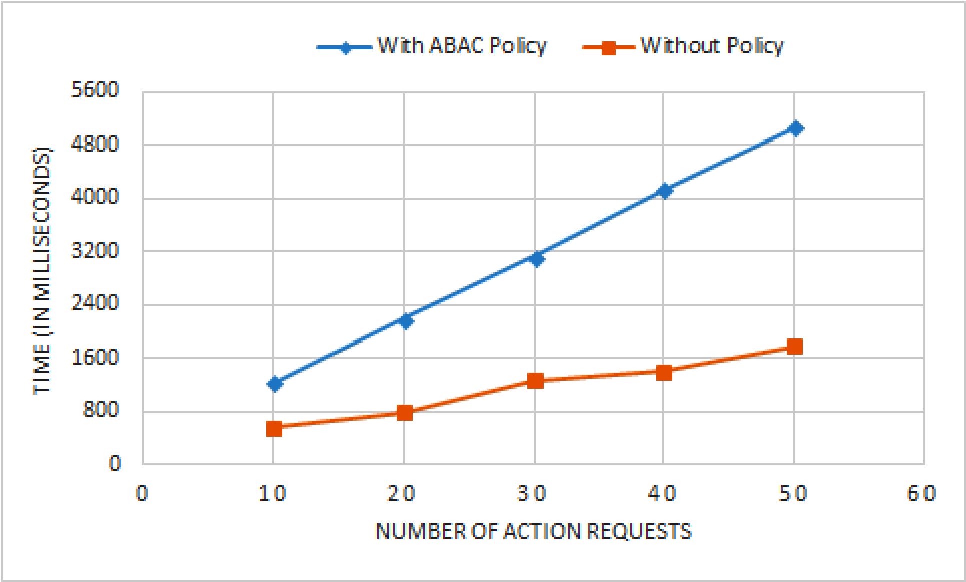
Number of Requests	Policy Enforcer Execution Time (in ms)
10	0.0501
20	0.1011
30	0.1264
40	0.1630
50	0.1999

Policy Enforcement Time

	CARS NOTIFIED	
n^{th} Request	With ABAC Policy	Without Policy
41^{st}	2	5
42^{nd}	3	5
43^{rd}	5	5
44^{th}	3	5
45^{th}	2	5
46^{th}	3	5

Relevance of Alerts and Notifications

Performance Metrics



Comparing Policy vs No Policy Execution Time

© Maanak Gupta

Conclusion and Future Work

- Proposed an Attribute Based Access Control solution for cloud assisted Smart Cars.
  - Introduced Dynamic Groups
  - Supports User Privacy Preferences and Location Centric
  - Proof of Concept implementation in AWS

- Future Research
  - Extensive and detailed evaluation
  - V2V and V2I secure trusted communication using Edge
  - Location preserving approaches

© Maanak Gupta