The Institute for Cyber Security

Center for Security and Privacy Enhanced Cloud Computing

Operation and Administration of Access Control in IoT Environments

Ph.D. Dissertation Defense

Mehrnoosh Shakarami
Institute for Cyber Security (ICS),
Department of Computer Science
The University of Texas at San Antonio

Committee:

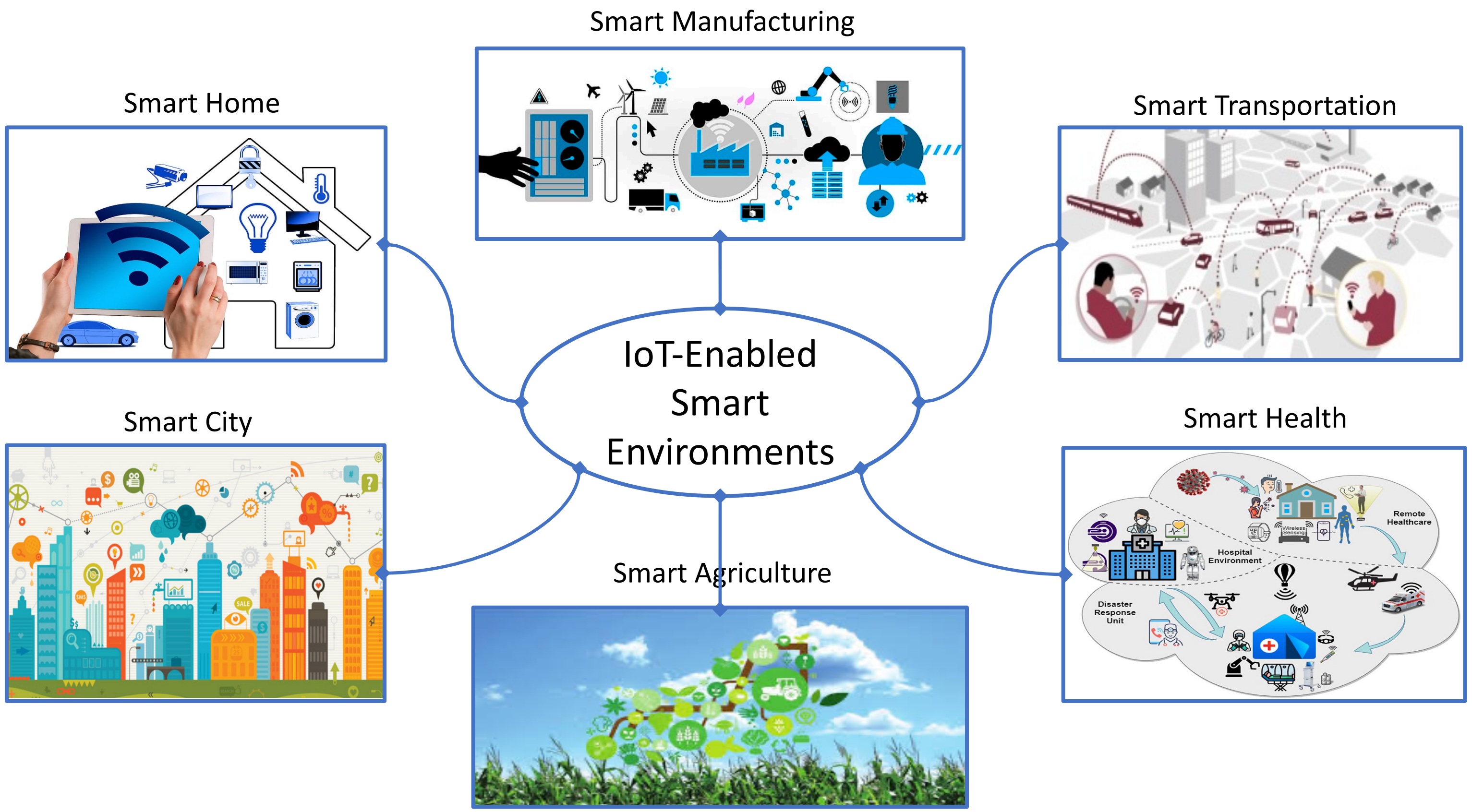
Prof. Ravi Sandhu (Advisor and Chair)
Dr. Murtuza Jadliwala
Dr. Ram Krishnan
Dr. Palden Lama
Dr. Xiaoyin Wang

April 2022

UTSA Computer Science

Internet of Things

- The Internet of Things (IoT) denotes a network of evolving and expanding number of technologies embedded in smart things with at least one network interface to connect, interact, and exchange data and information.



Access Control in IoT Environments

Policy Specification	Granularity	●	●	●	●	●
	Context Awareness	●	●	●	●	●
Policy Management	Handling the complexity of Environment	○	●	●	◑	●
	Usability	●	○	○	●	○
	Multi-domain Administration	○	●	●	◑	●
Policy Enforcement	Minimum user involvement	◑	●	●	●	●
	Light-weight	◑	◑	◑	●	●
	Reliability and Availability	●	●	●	●	●

- We recognize smart home IoT unique characteristics necessitate oriented authorization models to be particularly designed.
- Little attention has been paid to administration of access in IoT environments.
- The potential security violations which may happen in device-to-device interactions are largely uninvestigated.

Problem Statement

- The uptick of smart home technology adoption while its authorization is less investigated compared to other domains.
- We investigate three major access control-related topics which affect or directly provide authorization in the home IoT environment.
- ABAC is a widely adopted paradigm to provide access control in different IoT access, including smart home. Using ABAC expose smart environments to the risk of access violation due to inconsistency.
- While overall system security is crucially dependent on both administrative and operational authorization models, administration of access in smart home environments is overlooked.
- There is a hype around utilizing blockchain for access control in IoT environments, but it is not clear if the provided benefits could overcome inherent blockchain drawbacks.
- A holistic view of home automation demands catered access control specifications to facilitate device-to-device interactions.

Thesis Statements

- *The established paradigm of role-based access control can be utilized for access* ***administration*** *of user-to-device in corresponding operational access control models, which could be based on either role-based or attribute-based access control.*

- *If the operational authorization is based on attribute-based access control, a detailed analysis of required* ***consistency*** *for both mutable and immutable attribute values can ultimately benefit the overall safety of the system by providing a decision point with most recent values of attribute credentials.*

- *The established paradigm of attribute-based access control can be adapted and extended to provide a* ***device-to-device*** *access control approach towards considering heterogeneous IoT devices in a smart home as an ecosystem with intercommunication.*

Summary of Contributions

Operation and Administration of Access Control in IoT Environments

Safety and Consistency issue

1. Introduce/Formal definition of consistency levels for subject attributes.
2. Introduce *refresh* instead of *revoke* for increased safety and availability.
3. Formal definition of consistency levels using refresh.
4. Consistency levels for mutable attributes.

User-to-Device Access Control

RBAC Administration

1. Proposing RBAC administrative model with Smart home use case.
2. Blockchain-based Enforcement Architecture.
3. Proof-of-concept implementation using smart contracts on top of Ethereum.

Device-to-Device Access Control

Message-Based Access Control

Formal Specification of a device-to-device operational access control model using a message-based paradigm.

Scenario-Driven Access Control

Extension of the message-based to a scenario-driven access control model and define priorities to handle conflicts.

1. AWS-based enforcement architecture
2. proof-of-concept implementation for a smart home use case.

Summary of Contributions

Operation and Administration of Access Control in IoT Environments

Safety and Consistency issue

1. Introduce/Formal definition of consistency levels for subject attributes.
2. Introduce *refresh* instead of *revoke* for increased safety and availability.
3. Formal definition of consistency levels using refresh.
4. Consistency levels for mutable attributes.

User-to-Device Access Control

RBAC Administration

1. Proposing RBAC administrative model with Smart home use case.
2. Blockchain-based Enforcement Architecture.
3. Proof-of-concept implementation using smart contracts on top of Ethereum.

Device-to-Device Access Control

Message-Based Access Control

⇩

Formal Specification of a device-to-device operational access control model using a message-based paradigm.

Scenario-Driven Access Control

⇩

Extension of the message-based to a scenario-driven access control model and define priorities to handle conflicts.

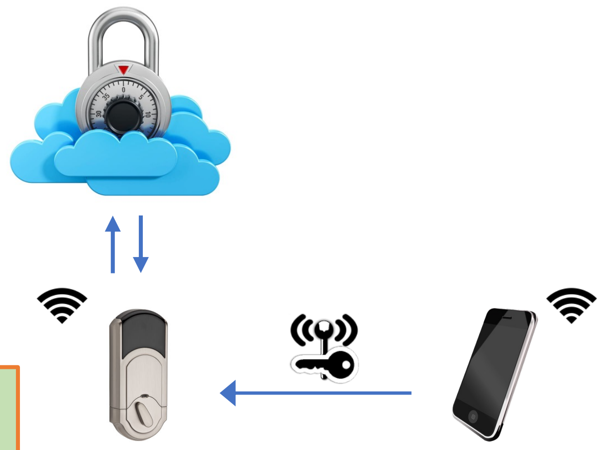
1. AWS-based enforcement architecture
2. proof-of-concept implementation for a smart home use case.

Safety and Consistency- a Smart Home IoT Use Case

Consider a smart lock with its access control deployed on the vendor's cloud.

Smart lock would authenticate and pair with a user's mobile in its Bluetooth range, then receive the status of that specific user's access key from the cloud.

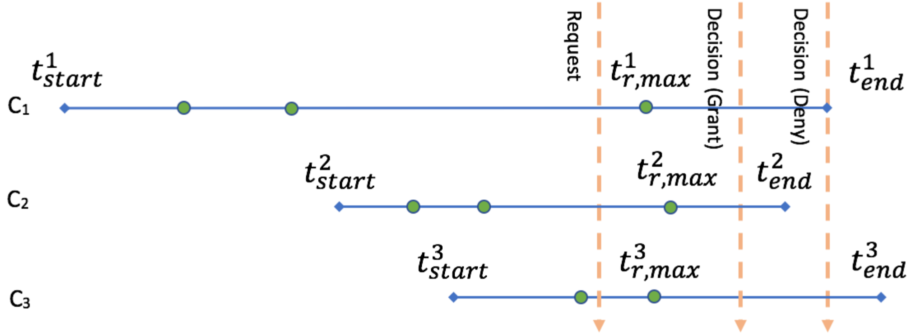
The key status would be also saved in the local database of the lock.



State Consistency Attack

- Due to intermittent Internet connection and limited storage space of home IoT devices, required authorization information might not be available in real time.
- There is an increased risk of making access control decisions based on outdated information.
- This problem may arise in any attribute-based access control environment in which attributes are provided incrementally to the decision point.
- We investigate this problem in general, interpreted with use cases in a smart home IoT environment.

Exposure of decision point to stale attributes, hence access violation.

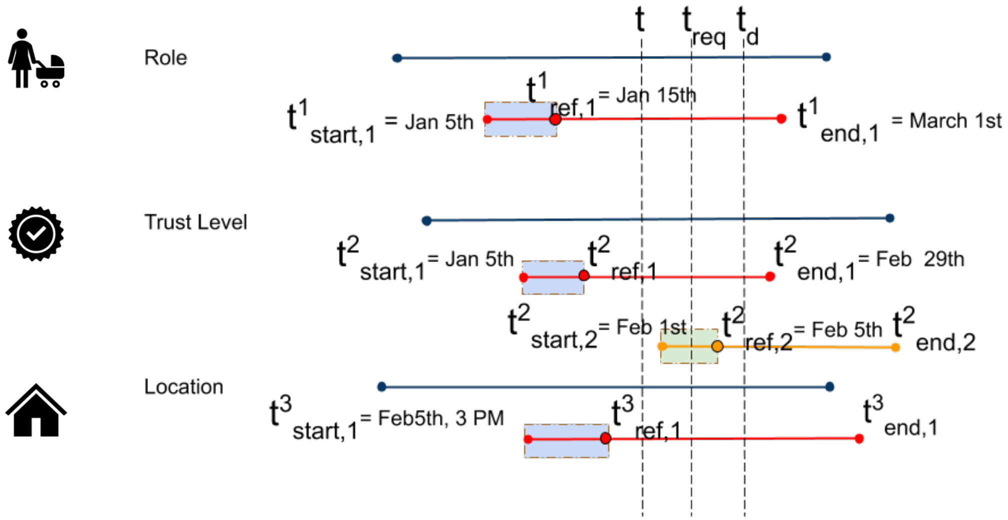


Refresh instead of Revoke – A smart Home IoT Use Case

Refresh replaces an older value with a newer one, while Revoke simply invalidates the old value.

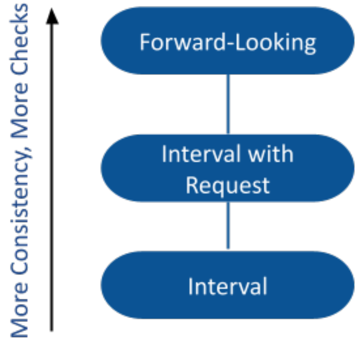
(s: user, o: thermostat, op: operation)

$$\leftrightarrow \Big((op.trustLevel \leq u.trustLevel) \bigwedge \big((u.role == \text{“parent”}) \vee \big((u.rolr \in \{\text{”parent”}, \text{”kid”}\}) \wedge (u.location == \text{“home”}) \big) \big) \Big)$$



Interval Consistency with Request Time

What if babysitter tries to adjust the thermostat after leaving home?

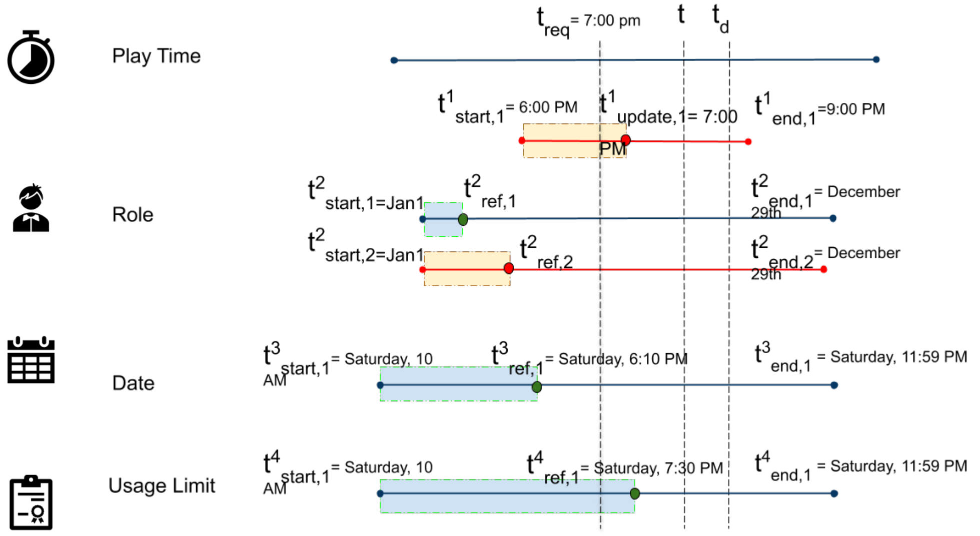


Refresh-Based Consistency Levels

Safety and Consistency of Mutable Attributes Using Quotas – A smart Home IoT Use Case

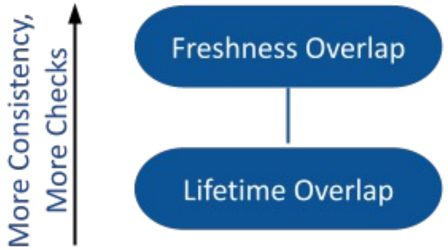
Mutable Attribute: Attribute changes are the consequence of access utilization by subject. We claim the revocation scenario to be inappropriate for mutable attributes.

$$(s{:}\,user, o{:}\,playstation, op{:}\,operation)$$
$$\leftrightarrow \Big((u.quota > 0) \bigwedge \big((u.role == \text{“parent”}) \vee ((u.rolr == \text{”kid”}) \wedge (day\_of\_week \in \{\text{”Saturday”}, \text{”Sunday”}\})) \big) \Big)$$



Lifetime Overlap Consistency Level

What if parents decide to reduce the usage limit?



Summary of Contributions

Operation and Administration of Access Control in IoT Environments

Safety and Consistency issue

1. Introduce/Formal definition of consistency levels for subject attributes.
2. Introduce *refresh* instead of *revoke* for increased safety and availability.
3. Formal definition of consistency levels using refresh.
4. Consistency levels for mutable attributes.

User-to-Device Access Control

RBAC Administration

1. Proposing RBAC administrative model with Smart home use case.
2. Blockchain-based Enforcement Architecture.
3. Proof-of-concept implementation using smart contracts on top of Ethereum.

Device-to-Device Access Control

Message-Based Access Control

Formal Specification of a device-to-device operational access control model using a message-based paradigm.

Scenario-Driven Access Control

Extension of the message-based to a scenario-driven access control model and define priorities to handle conflicts.

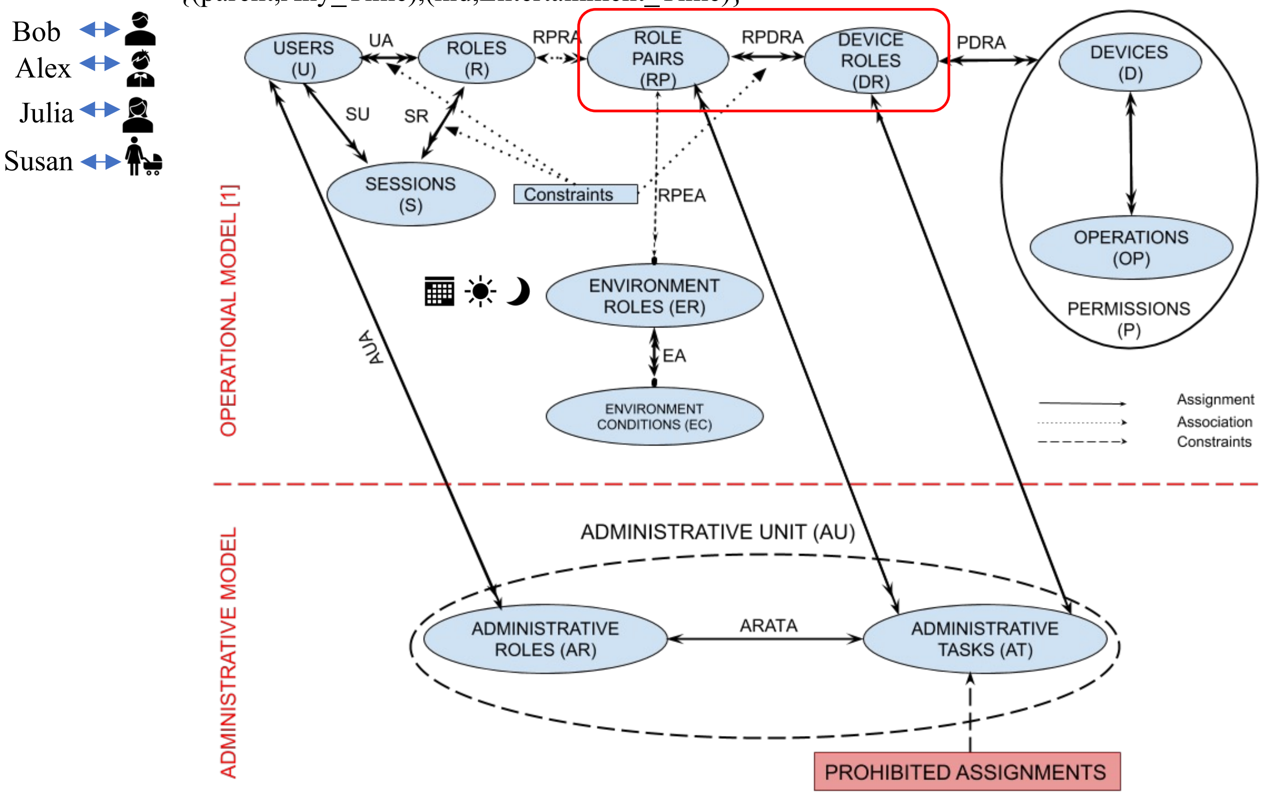
1. AWS-based enforcement architecture
2. proof-of-concept implementation for a smart home use case.

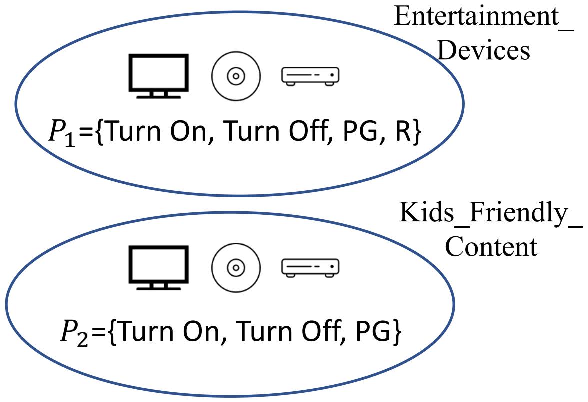
Administration of Access

- Access administration has been overlooked.
  - Overall system security is crucially dependent on both the administrative and operational models.
  - Many lack the support of a formal model and rely on informally assumed policy objectives.
- Administration of access must be carefully crafted to ensure that policy does not drift away from its original objectives.
- Access administration in a smart home environment is a particular access management problem due to:
  - lack of expertise in home users.
  - shared ownership of IoT devices.
- We follow PEI (Policy, Enforcement Architecture, Implementation) as our reference model.
- We opt for EGRBAC as our underlying operational model, for it being:
  - Granular at permission level, instead of device level.
  - Capture the environment conditions and provide contextuality.

PEI: Policy – Basic Administrative Model

{(parent,Any_Time),(kid,Entertainment_Time)}

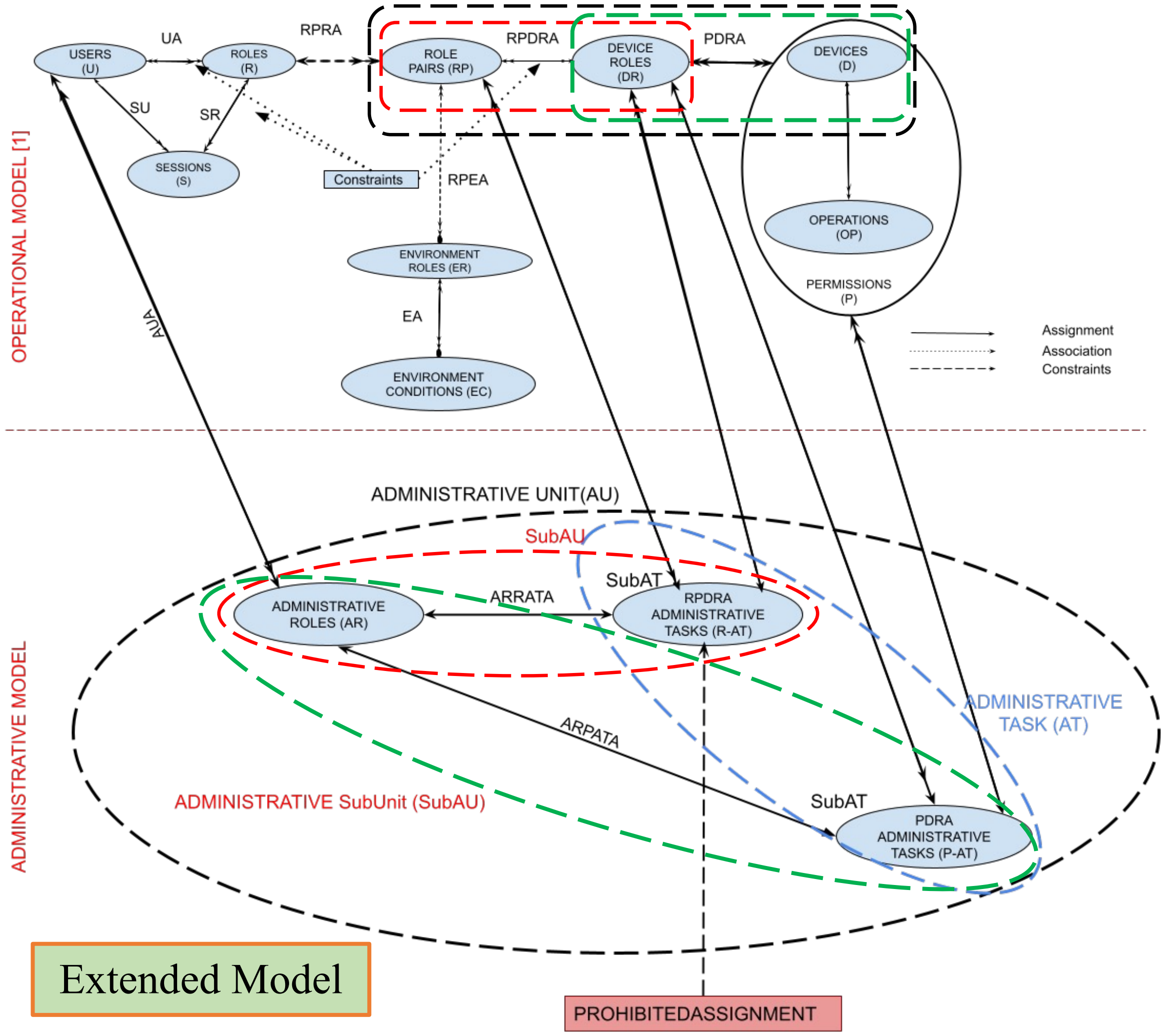




Basic Model to manage RPDRA Assignment

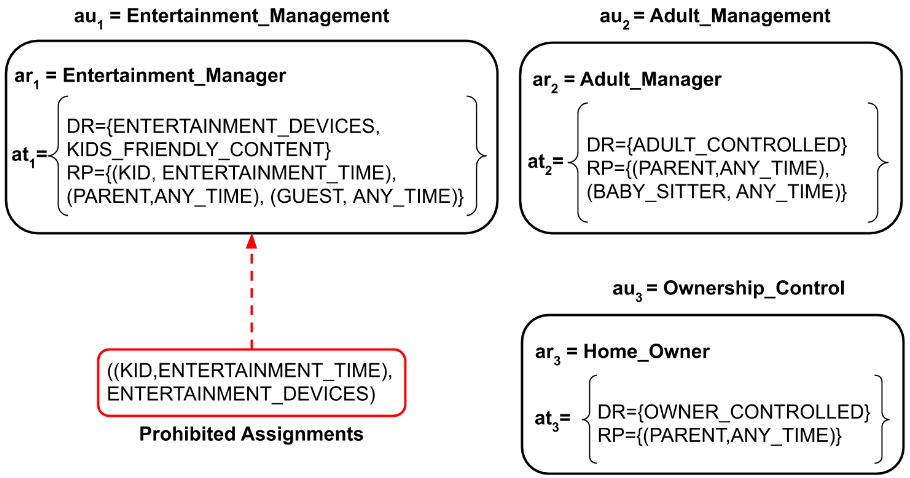
- We recognize administration is best to be done decentralized. Decentralization provided through Administrative Units (AU).
- We define one administrative unit per operational assignment to be managed, which includes a unique administrative role (AR) and a set of administrative tasks (AT).
- Authorization is scoped as a set of administrative tasks defined to manage corresponding assignments in the operational model.

PEI: Policy – Extended Administrative Model

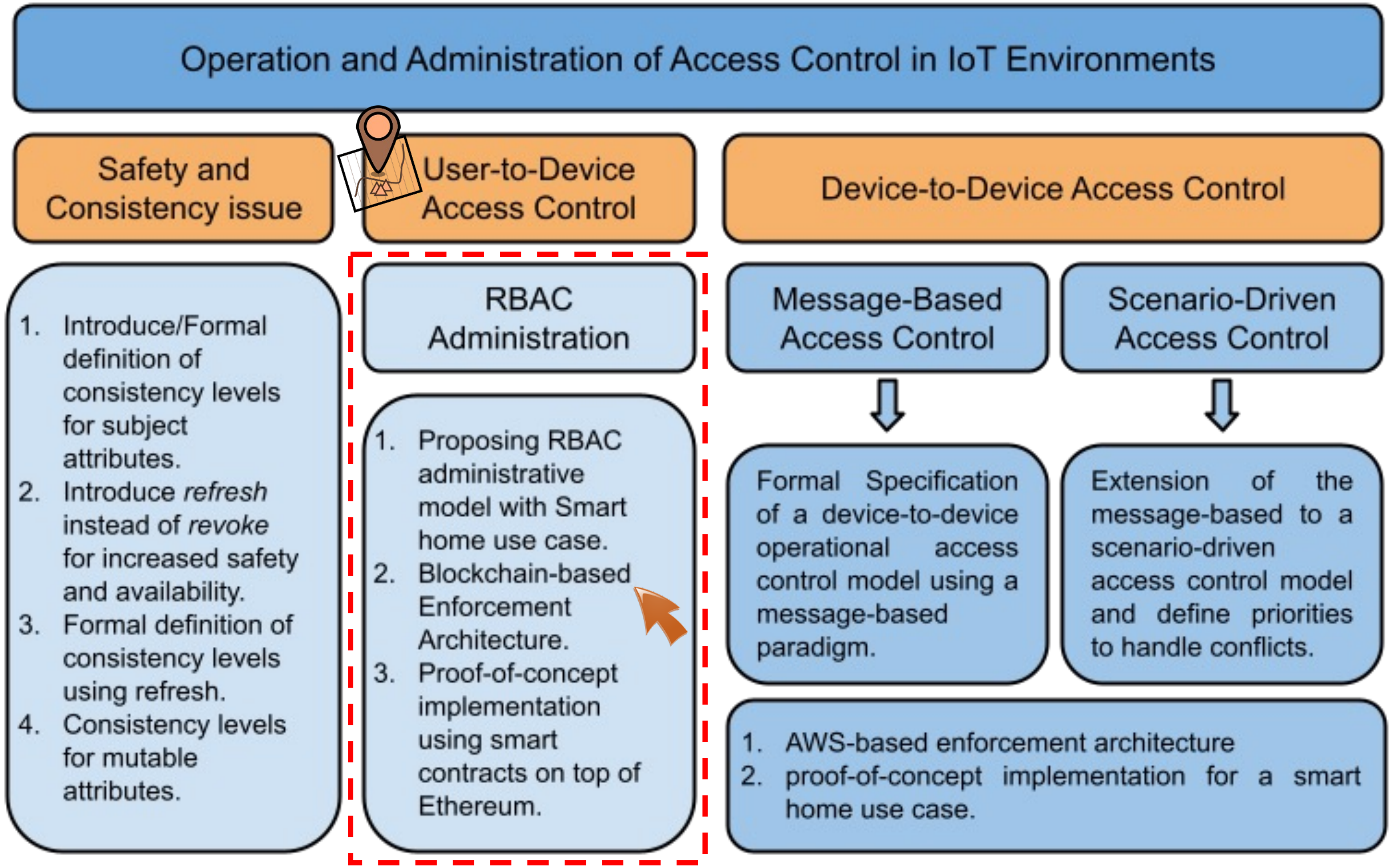


Extended Model

- We extended our administrative model by defining one administrative unit per operational assignment to be managed.
- Each administrative unit includes a unique administrative role which controls a predefined set of administrative tasks which represents its scope of administration.



Summary of Contributions



PEI: Blockchain-Based Enforcement Architecture

- Blockchain for Access Control:

Benefits:
- Decentralized Control
- Transparency and Auditability
- Distributed Information
- Tamper-proof

Why NOT Blockchain for Operational Access Control:
- IoT Constraints
- Long Transaction Confirmation Time
- Financially Prohibitive

Why Blockchain for Administrative Access Control:
- Less Frequency of Administrative Tasks
- Posteriori Analysis
- Scalable
- No need for IoT devices to be engaged in blockchain

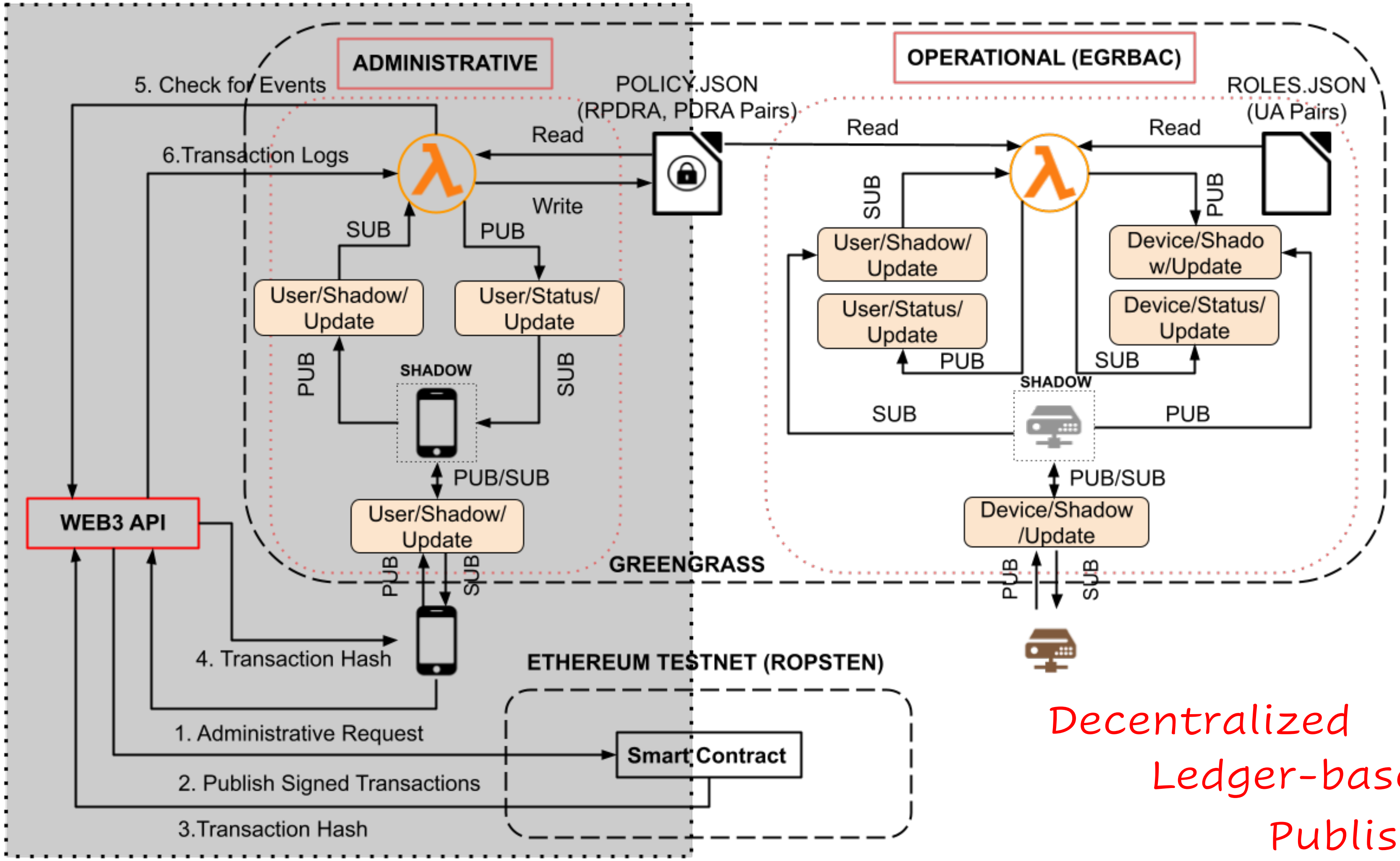
- Threat Model and Blockchain Benefits:

Threat Model:
- Insider Attack:
  - Spoofing, Tampering, Privilege Escalation, Repudiation

Assumptions:
- Users' communication with edge is secure over local network.
- Routing attacks are out-of-scope
- Attacks against Web3 API are out-of-scope
- API attacks against user's private keys in their wallets considered to be out-of-scope

Blockchain Security Benefits:
- Administrator account cannot be faked
- Administrative policy is encoded in a smart contract recorded to ledger via consensus
- System is equipped with transparency and auditability

PEI: Enforcement Architecture

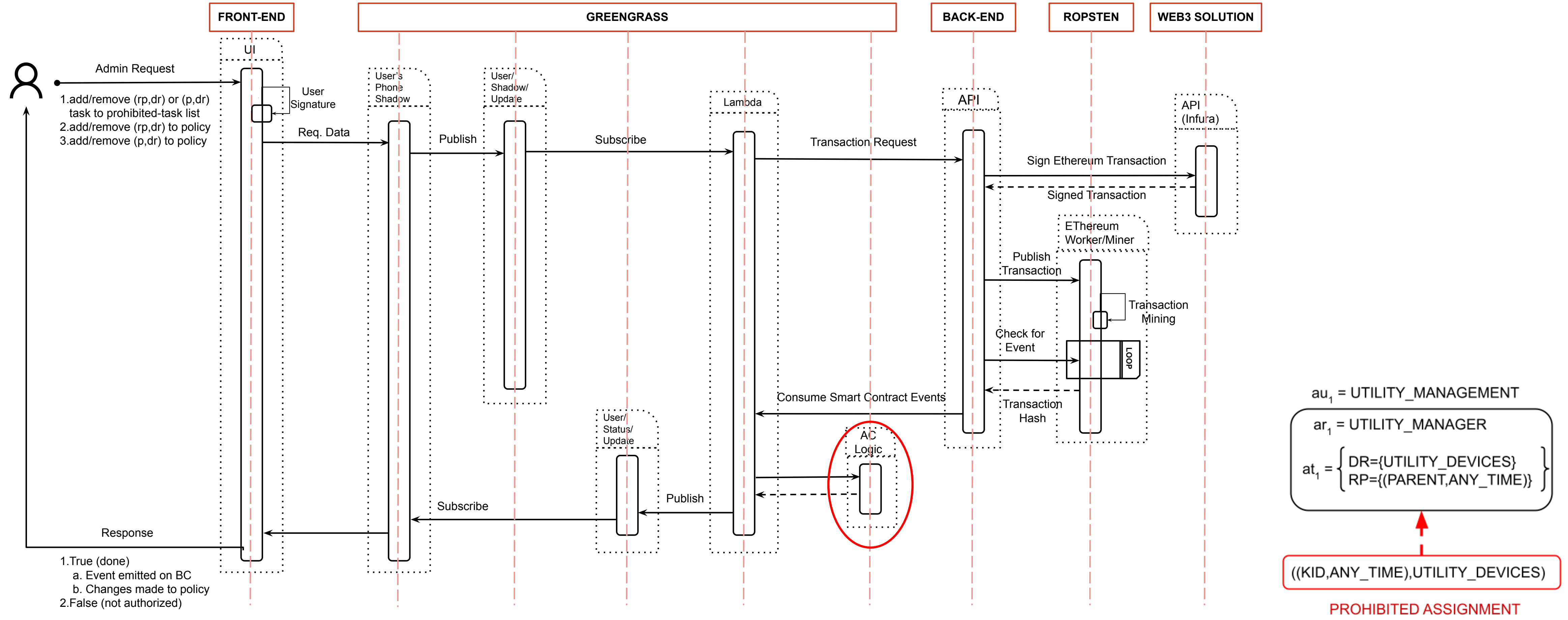


Decentralized

Ledger-based

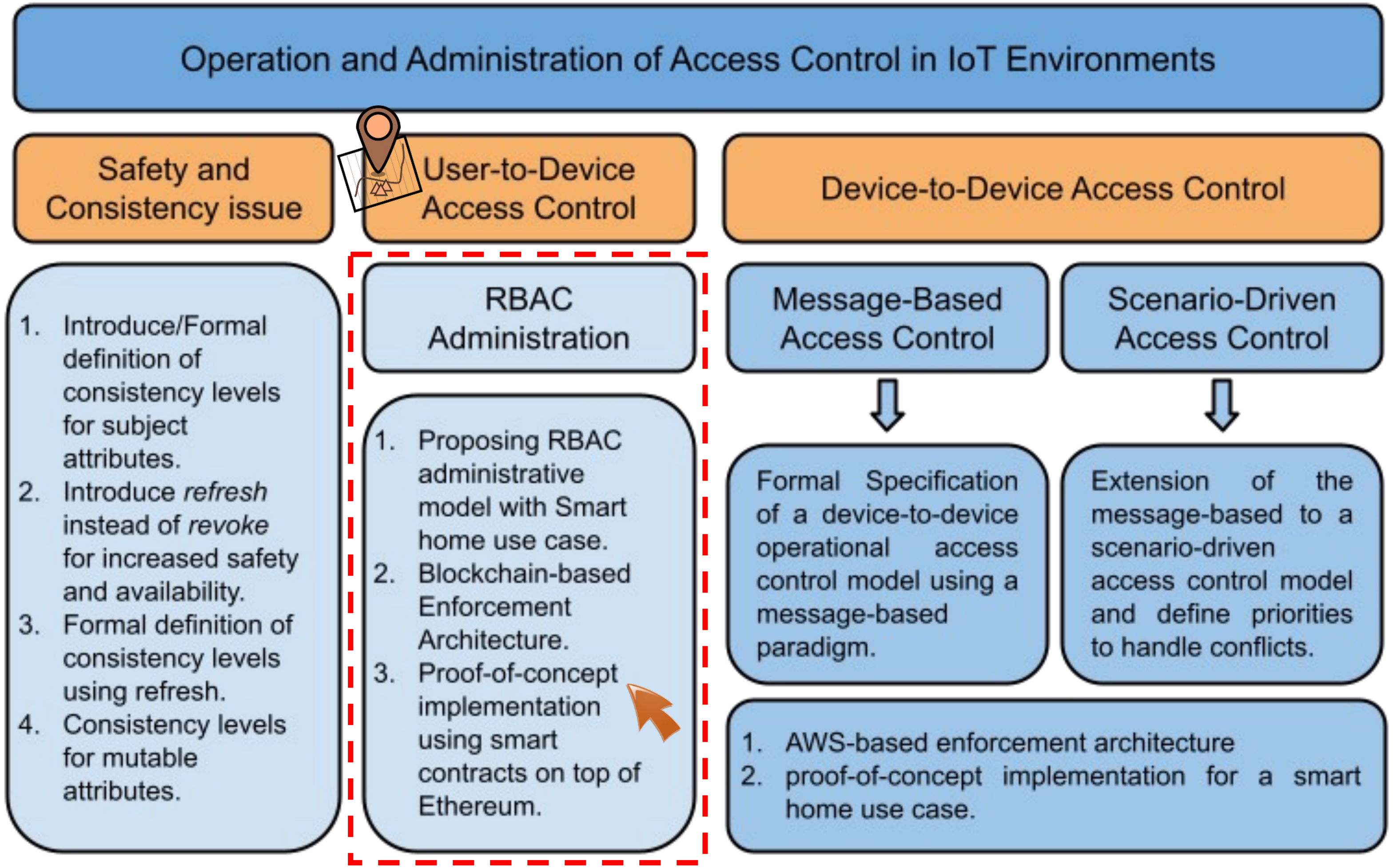
Publish-subscribe

Sequence Diagram



World-Leading Research with Real-World Impact!

Summary of Contributions



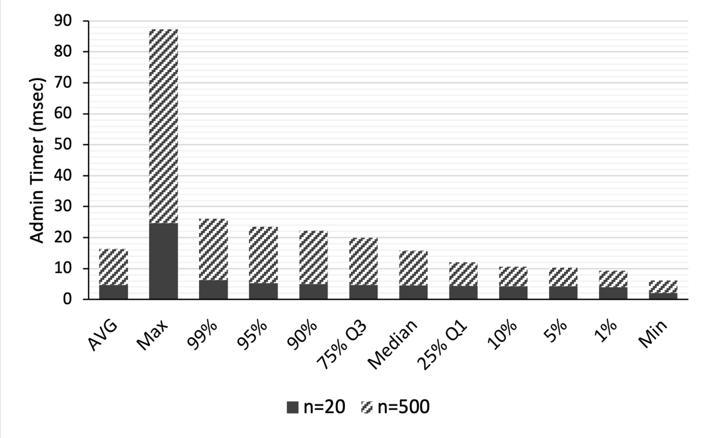
PEI: Implementation Setup

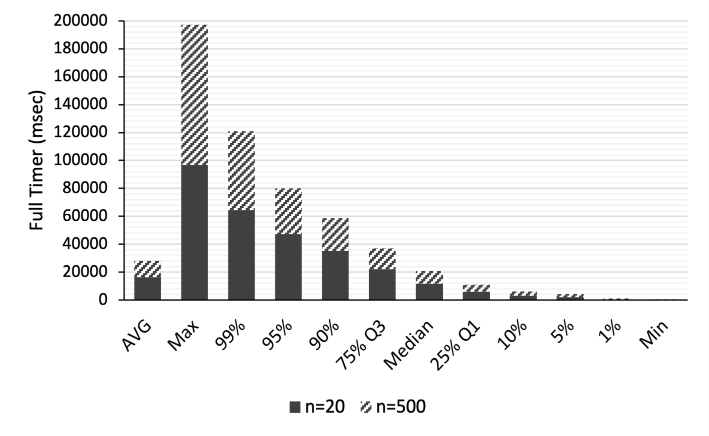
- Administrative access control policy implemented in a single smart contract on the Ropsten.
- Different administrative controls are coded as functions, which would be triggered by transactions.
- Smart contract is programmed in Solidity and tested it on Remix IDE.
- Infura is used as web3.0 API to interact with blockchain.

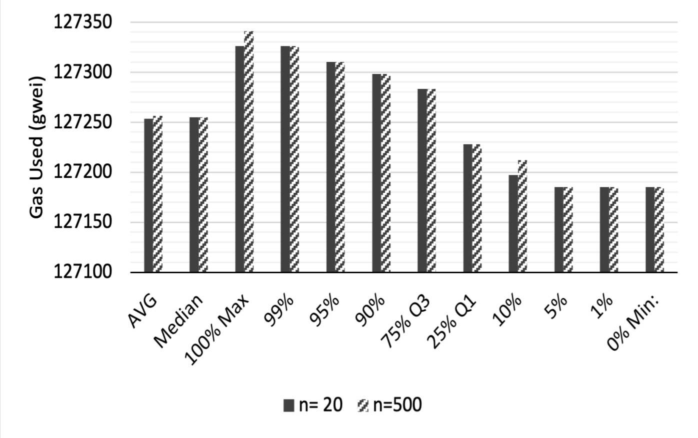
- Experiment Environment:
  - AWS IoT Greengrass v1.
  - Greengrass runs on a dedicated virtual machine: one virtual CPU, 2 GB of RAM and 20 GB hard drive.
  - The virtual machine's operating system is Ubuntu 20.4.2 LTS and it is connected to a 1 Gbps network.
  - Our AWS lambda code on the Greengrass is written in Python 3.8 and is running in a long-lived isolated runtime environment with limited RAM of 256 MB

- Experiments are done for a normal distribution with a 99.9% confidence interval.
- We ran our experiments in two settings with the policy sizes of n=20 and n=500.
  - Both experiments were run for a total of 500 times.

Implementation Results:







Admin Timer: After a transaction has been successfully mined, Lambda checks the logs to search out the succeeded transactions. Then, it makes appropriate changes to the "policy.json" file and publishes the results to the User/Status/Update to inform the user about his/her administrative request.

Full Timer: Complete cycle of an administrator submitting a request, to that request being mined, and the lambda function processing the results and updating as necessary.

Gas Used: the actual amount of gas which was used during execution. Gas prices are denoted in GWEI, which equals to 10^{-9} ETH. We calculated the monetary cost of each transaction to be 28 cents.

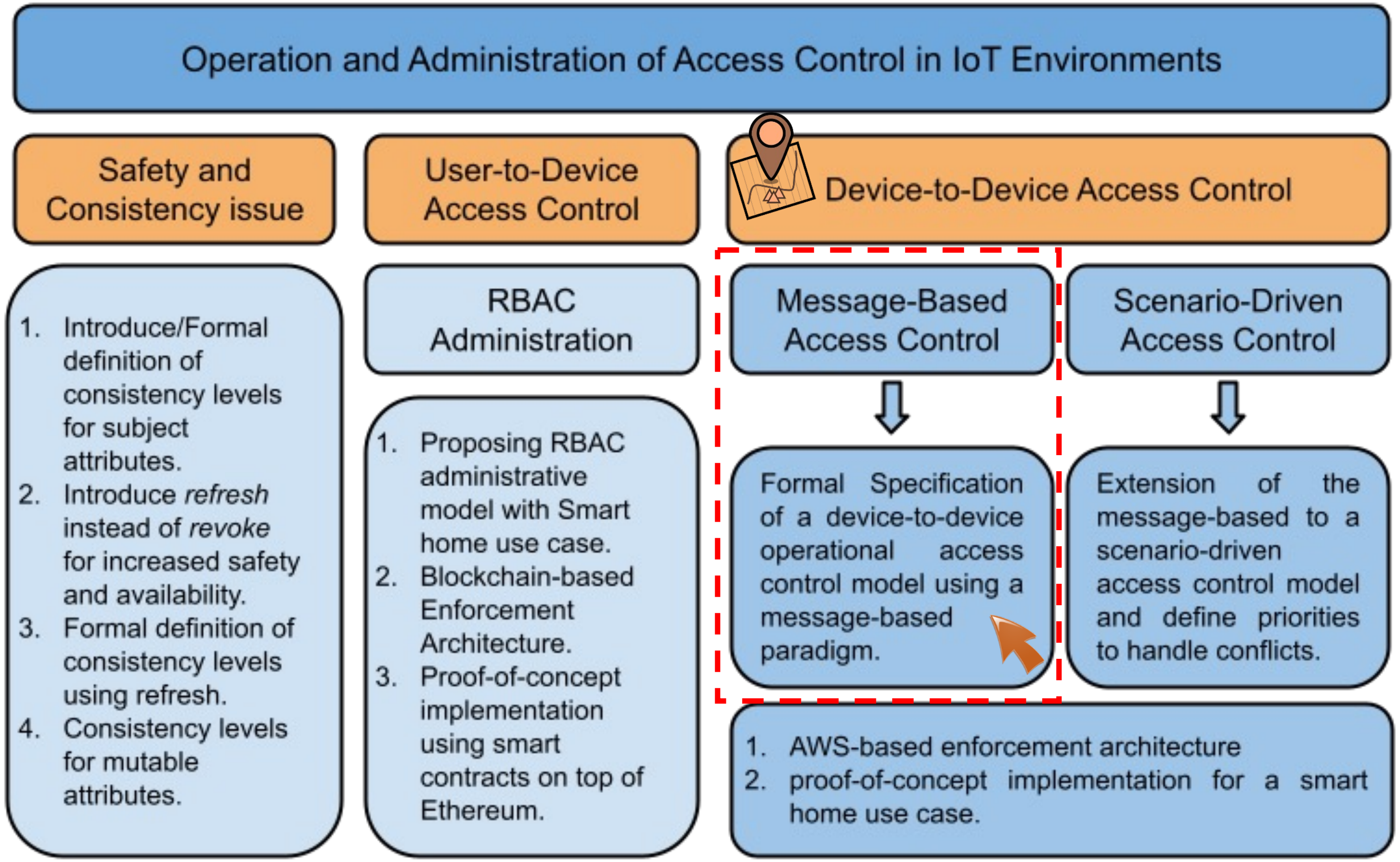
Discussions

- Our administrative model features:
  - Decoupled Assignment and Revocation
  - Symmetric Assignment and Revocation
  - Generalizability
  - Transparency and Auditability
  - Privacy
- Security considerations specific to our architecture:
  - Smart Contract Security
  - Device-Cloud Communications
- Limitations:
  - Continuous Access control and Mutability
  - Handling Conflicts
- Our implementation results are reassuring that although the use of blockchain for operational access control is NOT promising, BUT it is promising to be utilized at administrative level.

Blockchain Hype

EGRBAC is not chosen as a de-facto!

Summary of Contributions

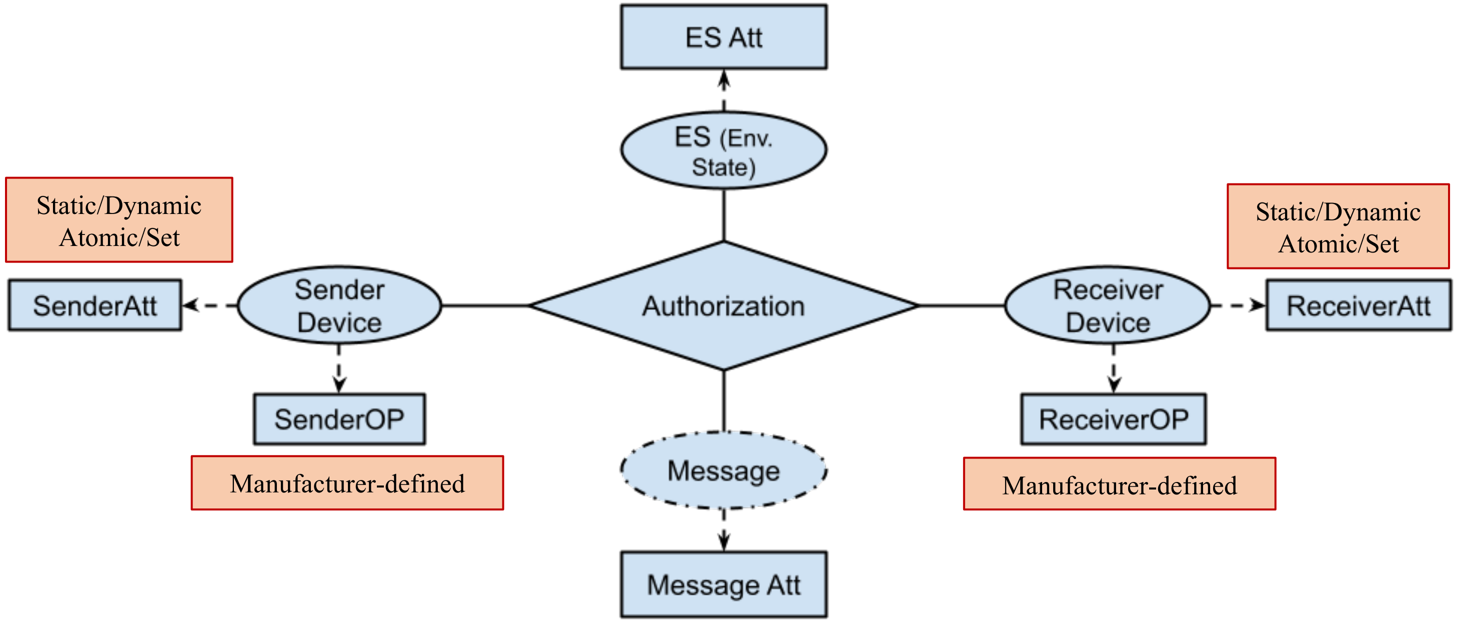


Motivation

- Seamless interoperability among IoT devices, device-to-device communication, is imperative for incipient evolution of the IoT ecosystem.
- Many standardization efforts as well as proposals for integration of heterogenous IoT platforms are going on in both academic and industry, but there is no de-facto standard and there would be none in foreseeable future.
- There is heterogeneity in all levels of IoT technology, including device, networking, middleware, application, and data/semantics of IoT scenarios, makes heterogeneous IoT devices cumbersome.
- There is no access control model specification to provide an access control model for device-to-device interoperability.
- Scenarios of device-to-device interactions are inevitable for real-life home automation, which brings added convenience but also inherent security risks.

We present an access control model which governs authorized flow of information for device-to-device interactions in a smart home IoT using Attribute-Based Access Control (ABAC) for the first time, utilizing a message passing paradigm.

PEI: Message-Based Device-to-Device ABAC Access Control Model



$$M = \{m_i : m_i = ((att_1 = value_1), (att_2 = value_2), \ldots, (att_n = value_n))\}$$

$$CheckAtt = True \leftrightarrow typeSetAtt(m) = \begin{cases} \subseteq 2^{DAA(r)} & if \quad m.value_1 = \text{“query”} \\ .\in DOA(r) & if \ m.value_1 = \text{“command”} \\ \subseteq 2^{DAA(s)} & if \quad m.value_1 = \text{“info} \end{cases}$$

Core Components

– D is a set of smart home IoT devices deployed by homeowner.
– OP is a set of operations available on different devices in the system (manufacturer specified).
– $ES = \{current\}$ is the singleton set, representing the environment state at the current time instant.
– $Ent = D \cup ES \cup M$ is the set of entities in the system, where the set of messages M is defined below.
– $DOA : D \rightarrow 2^{OP}$ is a one to many relation which associates a device to its available operations as specified by the device manufacturer.

Attribute Functions

– DAA, EAA are respectively sets of attribute functions which associate a device or the current environment state with attribute values.
– $attValueType : DAA \cup EAA \rightarrow \{atomic, set\}$
$\forall att \in DAA \cup EAA$, $Range(att)$, is the attribute range, a finite set of atomic values.
–Each $att \in DAA \cup EAA$ maps a device/environment to a single *atomic* value or to a finite *set* of values, as follows:

$$-att : DAA \cup EAA \rightarrow \begin{cases} Range(att) & \text{if } attValueType(att) = atomic \\ 2^{Range(att)} & \text{if } attValueType(att) = set \end{cases}$$

$$-attAssignType : DAA \cup EAA \rightarrow \begin{cases} static & \text{set/changed via administrative actions} \\ dynamic & \text{set/changed automatically by deployed sensors in home IoT} \end{cases}$$

Message and Message Functions

– $M = \{m\}$ is the set of all messages in the system.
– $m = \{(att_1, value_1), (att_2, value_2), \ldots, (att_n, value_n)\}$, represents any single message in the system with n different attributes, each of which is indicated as a $(key, value)$ pair.
– $typeSet = \{\text{"query"}, \text{"command"}, \text{"info"}\}$ is a mandatory first attribute in every message which indicates its *type* and thereby the rest of message attributes.
–For each $m \in M$, we assume the first attribute determines the type of the message: $att_1 \in typeSet$
– $typeSetAtt : M \rightarrow 2^{DAA} \cup 2^{DOA}$, is a function which indicates the set of attribute keys required to be communicated based on the message type, supposed to be communicated via $\{att_2, \ldots, att_n\}$ in each message.

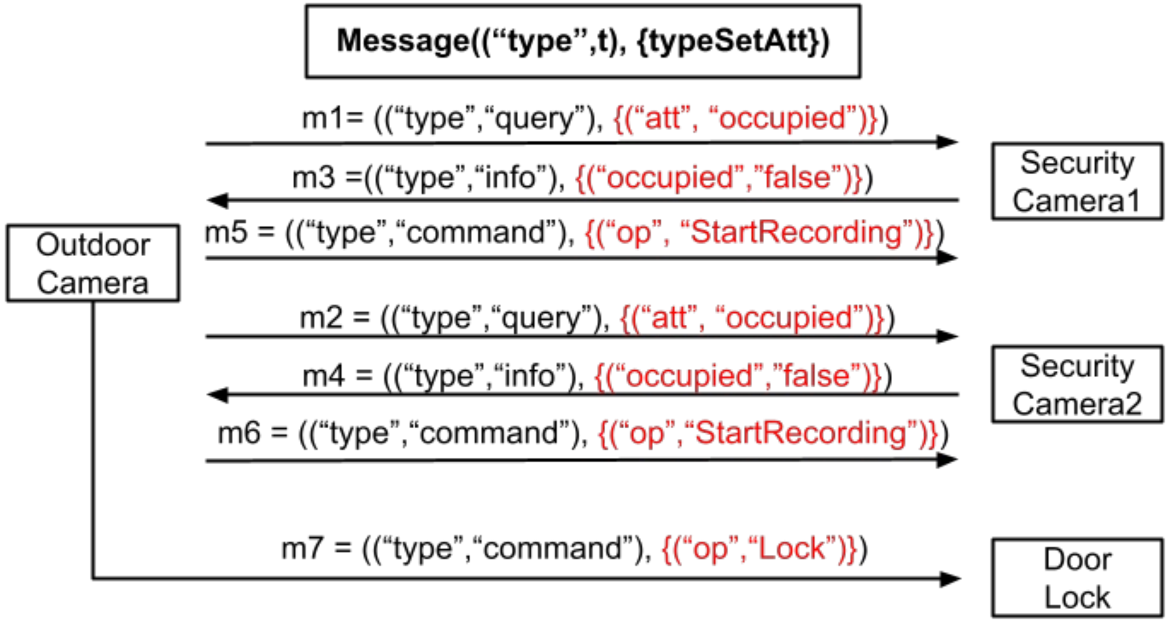
Check Access Predicate

– $CheckAccess$ is evaluated when a sender device (s) tries to send a message (m) to a receiver device (r) in context of current environment state ($current$) and is evaluated based on following formula:
– $CheckAccess(s : D, m : M, r : D, current : ES) \equiv$
$CheckAtt(s : D, m : M, r : D, current : ES) \wedge Authorization(s : D, m : M, r : D, current : ES)$

$$-CheckAtt = True \iff typeSetAtt(m) = \begin{cases} \subseteq 2^{DAA(r)} & \text{if } m.value_1 = \text{"query"} \\ \in DOA(r) & \text{if } m.value_1 = \text{"command"} \\ \subseteq 2^{DAA(s)} & \text{if } m.value_1 = \text{"info"} \end{cases}$$

– $Authorization(s : D, m : M, r : D, current : ES)$ is a logical proposition which could be evaluated to either True or False and is created using following policy rules.
– $p \equiv (p) \mid \neg p \mid p \wedge p \mid p \vee p \mid \exists x \in set.p \mid \forall x \in set.p \mid set \ \Delta \ set \mid atomic \in set$
– $\Delta \equiv \subset \mid \subseteq \mid \not\subseteq \mid \cap \mid \cup$

A Smart Home IoT Use Case



CheckAccess(s:D, m:M, r:D, current:ES) ≡ $CheckAtt(s{:}D, m{:}M, r{:}D, current{:}ES) \wedge Authorization(s{:}D, m{:}M, r{:}D, current{:}ES)$

$Authorization(s{:}D, m{:}M, r{:}D, current{:}ES) = \big((m.att_1 = \text{“query”}) \wedge (typeSetAtt(m) \in \{\text{”recording”},\text{”occupied”}\}) \wedge (type(s) = type(r) = \text{“cameras”}) \wedge (location(s) = \text{”outdoor”}) \wedge (location(r) = \text{“indoor”})\big) \vee \big((m.att_1 = \text{“info”}) \wedge (typeSetAtt(m) \in \{\text{”recording”},\text{”occupied”}\}) \wedge (type(r) = type(s) = \text{”cameras”}) \wedge (location(s) = \text{“indoor”}) \wedge (location(r) = \text{“outdoor”})\big) \vee \big((m.att_1 = \text{“command”}) \wedge (typeSetAtt(m) \in \{\text{”StartRecording”},\text{”StopRecording”}\}) \wedge (type(r) = type(s) = \text{”cameras”}) \wedge (location(s) = \text{“outdoor”}) \wedge (location(r) = \text{“indoor”})\big) \vee \big((m.att_1 = \text{“command”}) \wedge (typeSetAtt(m) \in \{\text{”Lock”},\text{”Unlock”}\}) \wedge (type(s) = \text{“cameras”}) \wedge (type(r) = \text{“locks”}) \wedge (location(s) = \text{”outdoor”}) \wedge (location(r) = \text{“mainEntrance”})\big)$

Threat Model

- IoT devices' susceptibility to cyberattacks could make them disturbing security holes.
  - A hacker with access to your thermostat could fiddle with it, causing your HVAC system to malfunction.
  - An attacker may lock you out of your home in case the door lock is hacked.

- We adopt the Dolev-Yao (DY) threat model in which communicating endpoints cannot be assumed as trusted nodes in the network.
  - An adversary can tamper with the data through modification, deletion, or insertion of fake information as the communications rely on wireless medium.

- Our Assumptions:
  - As many IoT devices are not IP-enabled, using a gateway (GW) node in the network is inevitable. We assume the GW node in our model is trustworthy and available, which is a common assumption.
  - Attacker is assumed to be an outsider to the network with the goal of obtaining illegitimate access to available functionalities/operations of smart home IoT devices.
  - We do not consider adversaries to have physical access to IoT devices.

Our access control approach provides a defense-in-depth prevention/protection against any outsider attack, through restricting the set of authorized messages being communicated among IoT devices.

Summary of Contributions

Operation and Administration of Access Control in IoT Environments

Safety and Consistency issue

1. Introduce/Formal definition of consistency levels for subject attributes.
2. Introduce *refresh* instead of *revoke* for increased safety and availability.
3. Formal definition of consistency levels using refresh.
4. Consistency levels for mutable attributes.

User-to-Device Access Control

RBAC Administration

1. Proposing RBAC administrative model with Smart home use case.
2. Blockchain-based Enforcement Architecture.
3. Proof-of-concept implementation using smart contracts on top of Ethereum.

Device-to-Device Access Control

Message-Based Access Control

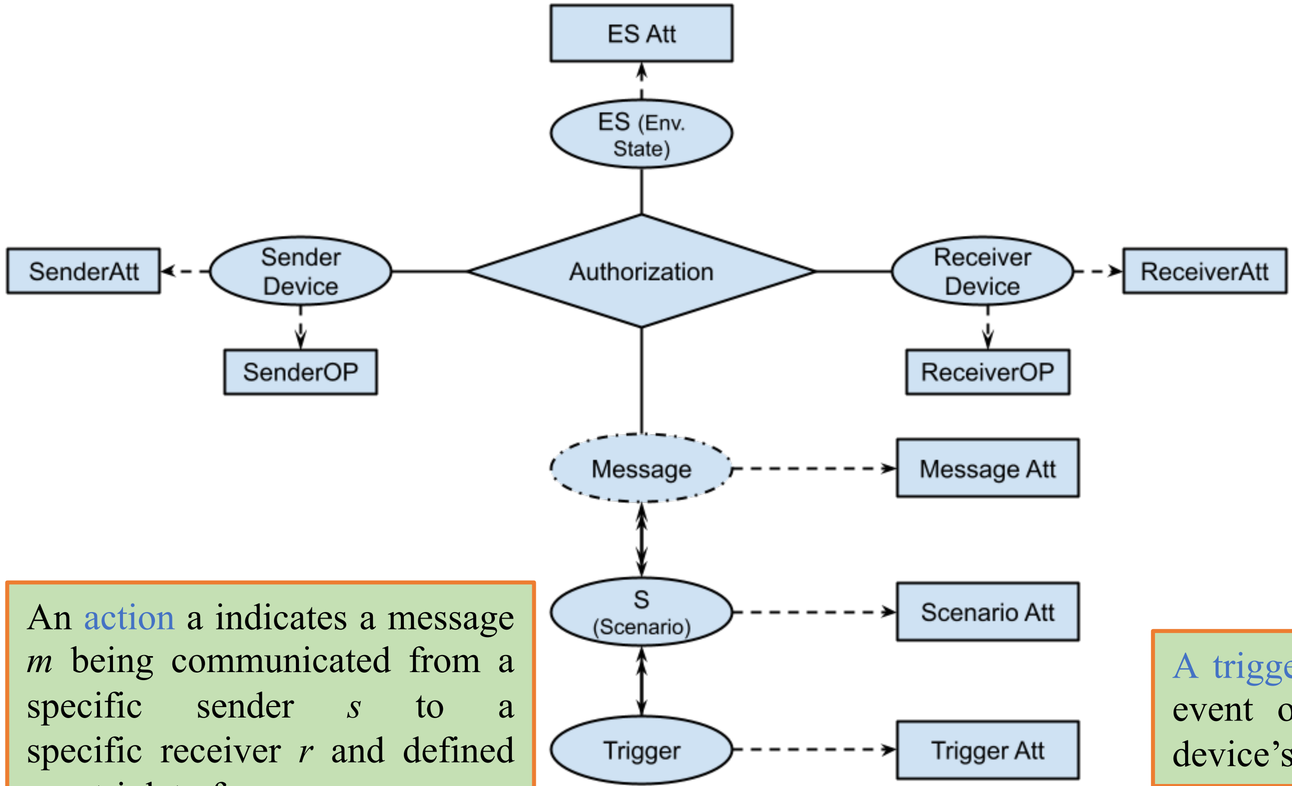
Formal Specification of a device-to-device operational access control model using a message-based paradigm.

Scenario-Driven Access Control

Extension of the message-based to a scenario-driven access control model and define priorities to handle conflicts.

1. AWS-based enforcement architecture
2. proof-of-concept implementation for a smart home use case.

PEI: Scenario-Based Device-to-Device ABAC Access Control Model



$-TeSA : TE \rightarrow 2^S$ is a one-to-many relation which defines a (set) of scenario(s) that would be provoked by a triggering event $te \in TE$.

–For each $s \in S : s =$ $\left(Action_s \subseteq A, tr_s : TE, pr_s : (pr, \prec), active : \{"true", "false"\}, id\right)$, and $pr_s = prA(tr_s)$.

$$-active(s : S) = \begin{cases} "true" & \text{while } tr_s \text{ is in effect.} \\ "false" & \text{As soon as } tr_s\text{, triggering event, reverts.} \end{cases}$$

$-typeSet = \{"query", "command", "info"\}$ is a mandatory attribute of every message which indicates its *type* and thereby the rest of message attributes.

–For each $m \in M$, we assume the first attribute must determine the type of the message and the second attribute must determines its priority: $att_1 \in typeSet, att_2 \in (pr, \prec)$

$-typeSetAtt : M \rightarrow 2^{DAA} \cup 2^{DOA}$, is a function which indicates the set of attribute keys required to be communicated based on the message type, supposed to be communicated via $\{att_3, ..., att_n\}$ in each message.

$-msgPrA : D \times M \times D \rightarrow (pr, \prec)$ is a function which is checked each time a device s_d wants to create a command message m and assigns proper priority to it, in order to be sent to device r_d .

$$-msgPrA(s : D, m : M, r : D) = \begin{cases} pr_s & \text{if } \left(m.att_1 = "command"\right) \wedge \left(\exists s \in S : active(s) = "true"\right) \wedge \\ & \left(\exists a \in Action_s : a.s = s_d \wedge a.r = r_d \wedge typeSetAtt(a.m) = typeSetAtt(m)\right) \\ \perp & \text{otherwise} \end{cases}$$

Chech Access Predicate

$-CheckAccess$ is evaluated when a sender device (s) wants to send a message (m) to a receiver device (r) in the context of current environment state ($current$) and is evaluated based on following formula:

$-CheckAccess(s : D, m : M, r : D, current : ES) \equiv$ $CheckAtt(s : D, m : M, r : D, current : ES) \wedge CheckPriority(s : D, m : M, r : D, current : ES) \wedge$ $Authorization(s : D, m : M, r : D, current : ES)$

$$-CheckAtt = True \iff typeSetAtt(m) = \begin{cases} \subseteq 2^{DAA(r)} & \text{if } m.value_1 = "query" \\ \in DOA(r) & \text{if } m.value_1 = "command" \\ \subseteq 2^{DAA(s)} & \text{if } m.value_1 = "info" \end{cases}$$

$$-CheckPriority(s : D, m : M, r : D, current : ES) \equiv \begin{cases} "false" & \text{if } (m.att_1 = "command") \wedge \\ & [((m."op", currentOP(r)) \in conflict(r)) \wedge (m.value_2 \prec currentPR(r)] \\ "true" & \text{otherwise} \end{cases}$$

$-Authorization(s : D, m : M, r : D, current : ES)$ is a logical proposition which could be evaluated to either True or False and is created using following policy rules.

$-p \equiv (p) \mid \neg p \mid p \wedge p \mid p \vee p \mid \exists x \in set.p \mid \forall x \in set.p \mid set\ \Delta\ set \mid atomic \in set$

$-\Delta \equiv \subset \mid \subseteq \mid \not\subseteq \mid \cap \mid \cup$

An action a indicates a message m being communicated from a specific sender s to a specific receiver r and defined as a triplet of:

$$a = (s{:}\,D, m{:}\,M, r{:}\,D)$$

A triggering event is a specific event or status from the IoT device's operation

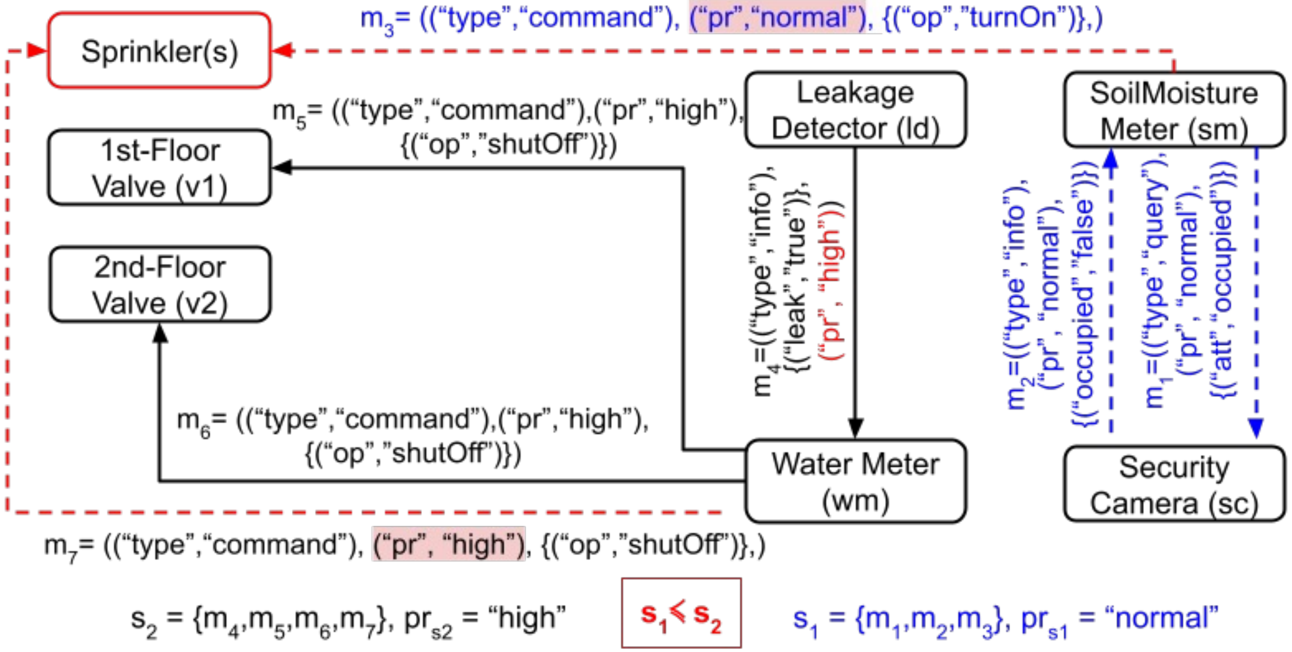
A scenario is defined as a set of actions to be done in the smart home

Priority is a totally ordered set relation, depicted as (pr, ≺) between any two triggering events tr_i and tr_j and is reflected in their (administratively) assigned priority values. In a smart home environment, the priority values could be defined as:

$$(pr, \prec) = (\perp \prec low \prec medium \prec high \prec)$$

Scenario-driven access control model resolves a class of conflicts due to receiving two conflicting command messages by the the same device .

A Smart Home IoT Use Case



CheckAccess(s:D, m:M, r:D, current:ES) $\equiv CheckAtt(s{:}\,D, m{:}\,M, r{:}\,D, current{:}\,ES) \wedge CheckPriority(s{:}\,D, m{:}\,M, r{:}\,D, current{:}\,ES) \wedge Authorization(s{:}\,D, m{:}\,M, r{:}\,D, current{:}\,ES)$

$Authorization(s{:}\,D, m{:}\,M, r{:}\,D, current{:}\,ES) = \big((m.att_1 = \text{“}info\text{”}) \wedge (typeSetAtt(m) \in \{\text{”}leak\text{”}\}) \wedge (type(r) = \text{”}valves\text{”}) \wedge (subtype(r) = \text{“}watermeter\text{”})\big) \vee \big((m.att_1 = \text{“}command\text{”}) \wedge (typeSetAtt(m) \in \{\text{”}ShutOff\text{”}, \text{”}TurnOn\text{”}\}) \wedge (type(r) = type(s) = \text{”}valves\text{”}) \wedge (subtype(s) = \text{“}watermeter\text{”})\big) \vee \big((m.att_1 = \text{“}query\text{”}) \wedge (typeSetAtt(m) \in \{\text{”}occupied\text{”}\}) \wedge (type(s) = \text{“}detectors\text{”}) \wedge (subtype(s) = \text{”}soil\text{”}) \wedge (type(r) = \text{“}cameras\text{”}) \wedge (location(r) = \text{“}outdoor\text{”})\big) \vee \big((m.att_1 = \text{“}info\text{”}) \wedge (typeSetAtt(m) \in \{\text{“}occupied\text{”}\}) \wedge (type(s) = \text{“}camera\text{”}) \wedge (\mathrm{location(s)} = \text{“}outdoor\text{”}) \wedge (type(r) = \text{“}detectors\text{”}) \wedge (subtype(r) = \text{“}soil)\big)$

Summary of Contributions

Operation and Administration of Access Control in IoT Environments

Safety and Consistency issue

1. Introduce/Formal definition of consistency levels for subject attributes.
2. Introduce *refresh* instead of *revoke* for increased safety and availability.
3. Formal definition of consistency levels using refresh.
4. Consistency levels for mutable attributes.

User-to-Device Access Control

RBAC Administration

1. Proposing RBAC administrative model with Smart home use case.
2. Blockchain-based Enforcement Architecture.
3. Proof-of-concept implementation using smart contracts on top of Ethereum.

Device-to-Device Access Control

Message-Based Access Control

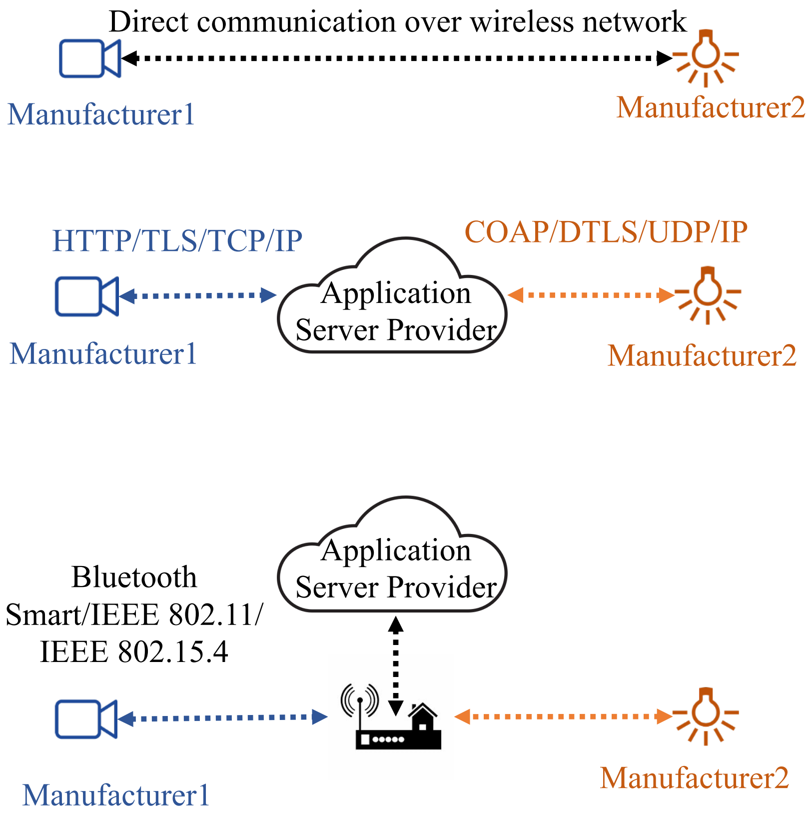
Formal Specification of a device-to-device operational access control model using a message-based paradigm.

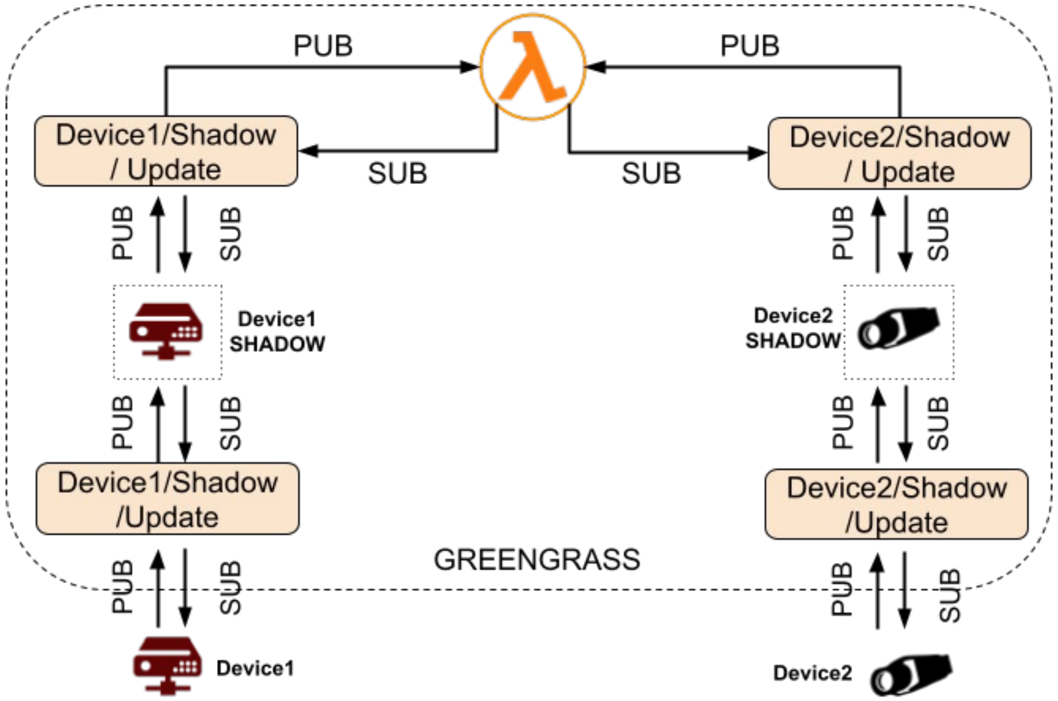
Scenario-Driven Access Control

Extension of the message-based to a scenario-driven access control model and define priorities to handle conflicts.

1. AWS-based enforcement architecture
2. proof-of-concept implementation for a smart home use case.

PEI: Enforcement Architecture





- Greengrass utilization enables devices to autonomously react to local events and securely communicate with each other over the local network.
- Using MQTT, devices will communicate on their private topics, device/shadow/update, to update their shadows, and trigger the local computation function, known as Lambda function.
- When Lambda is triggered, it executes the code we developed to check the policy.

Summary of Contributions

Operation and Administration of Access Control in IoT Environments

Safety and Consistency issue

1. Introduce/Formal definition of consistency levels for subject attributes.
2. Introduce *refresh* instead of *revoke* for increased safety and availability.
3. Formal definition of consistency levels using refresh.
4. Consistency levels for mutable attributes.

User-to-Device Access Control

RBAC Administration

1. Proposing RBAC administrative model with Smart home use case.
2. Blockchain-based Enforcement Architecture.
3. Proof-of-concept implementation using smart contracts on top of Ethereum.

Device-to-Device Access Control

Message-Based Access Control

Formal Specification of a device-to-device operational access control model using a message-based paradigm.

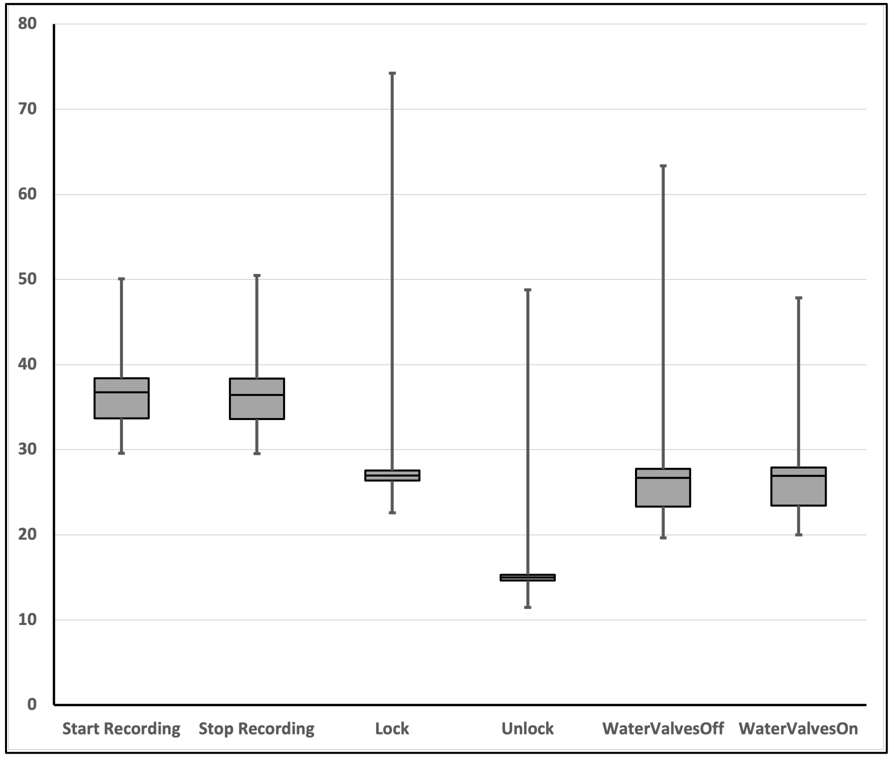
Scenario-Driven Access Control

Extension of the message-based to a scenario-driven access control model and define priorities to handle conflicts.

1. AWS-based enforcement architecture
2. proof-of-concept implementation for a smart home use case.

PEI: Implementation

Experiment	Min	10%	25%	Median	75%	90%	95%	99%	Max
Full Timer	24.2	25.4	26.0	34.9	36.7	39.1	42.9	47.2	67.2
State Update	3.5	3.6	3.7	5.7	6.6	6.9	7.4	10.2	21.8



A proof-of-concept implementation of two previously discussed smart home use cases. Statistics collected across 500 trials. All statistics are in milliseconds.

Full timer: a device to execute an action using our model/architecture, have lambda process it, and update the respective devices. The average full timer is 35 milliseconds.

State Update: The time to update a device's state, e.g., a door lock going from unlocked to locked, is on average 6 milliseconds.

Time per Action: depicted in the box-whisker plot. The box shows the 25 percentile and 75 percentile of the data, with median as the line contained inside the box.

Discussion

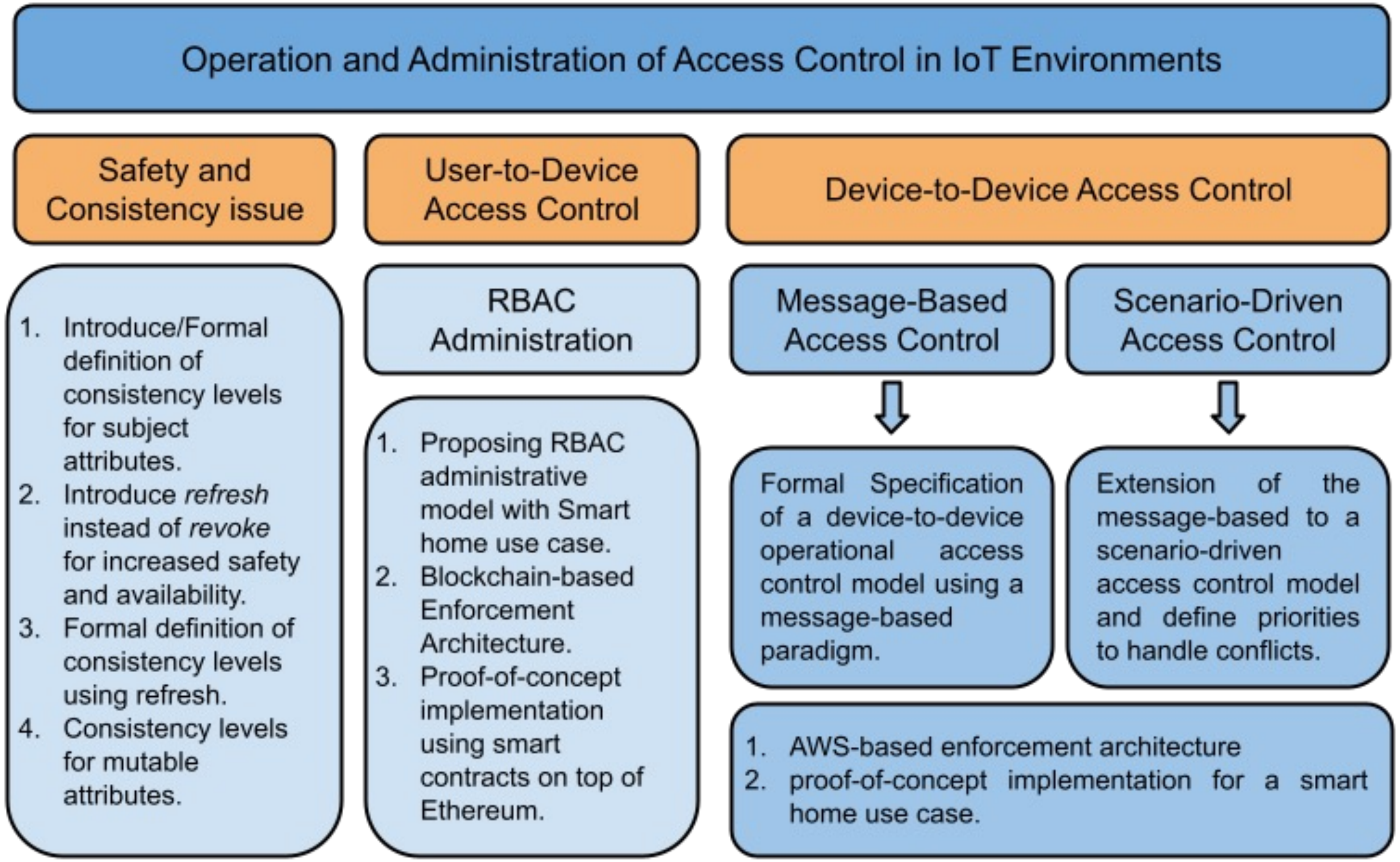
- Our model provides specification for mediating access in device-to-device communications for the first time. Presented model is context-aware, dynamic and lightweight.

- Continuity of Access: Any change in participating devices' attributes triggers the lambda which then re-evaluates the policy and adjusts the access authorizations accordingly.
  - In order to provide continuity as one of the model's elements, regardless of its enforcement method, the continuous retrieval and evaluation of entity/environment attributes should be incorporated in the model.
  - Continuity of access control could be concluded when the proposed model is able to revoke the previously granted access in case of any unintended change in attributes.

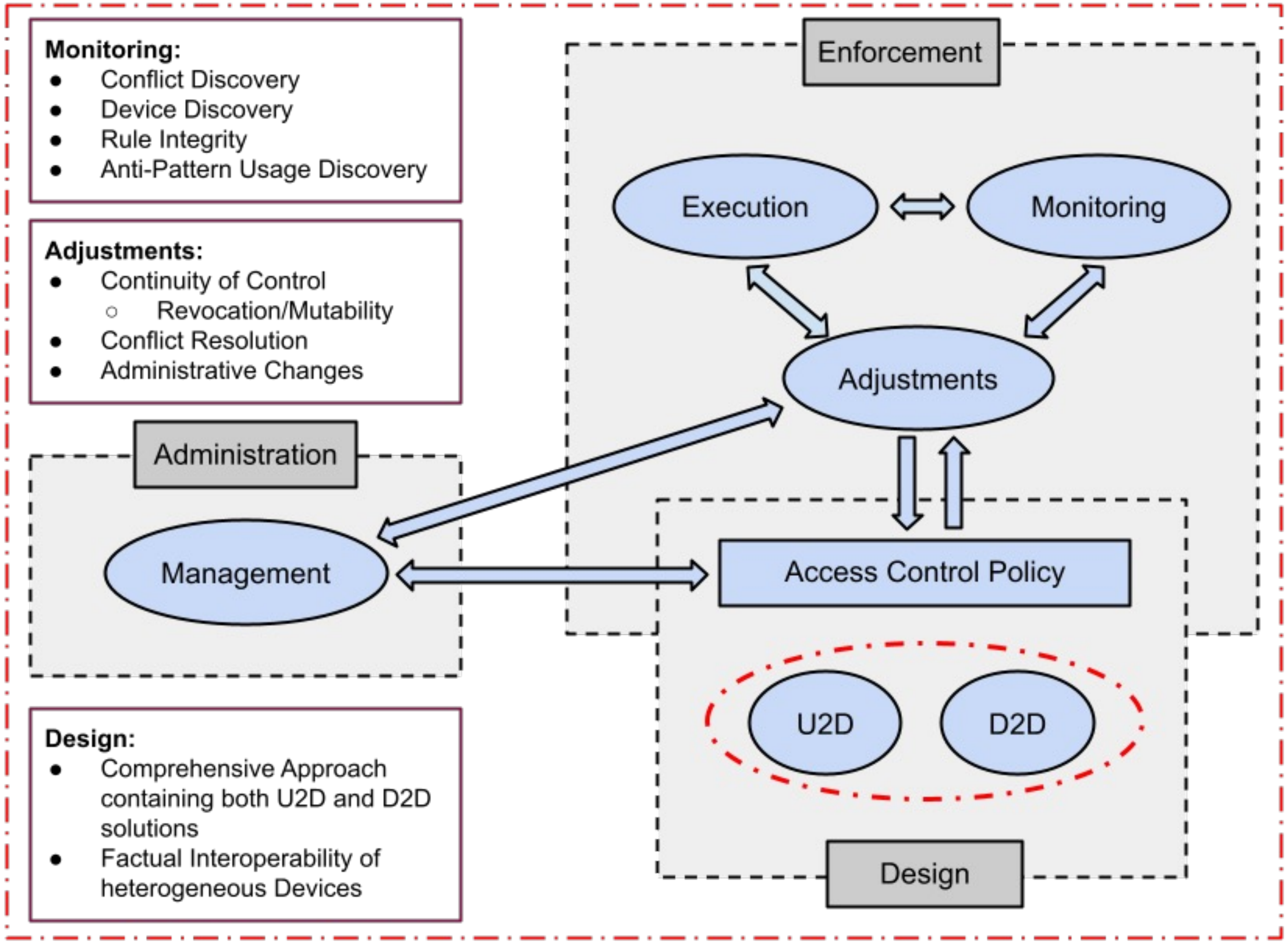
- Post-Authorization: Any message communication may result in a change of its sender/receiver attributes which is indicated through *Impact* function in our model.

- Architecture Agnostic: Our model is not peculiar to Amazon AWS.
  - We use AWS because of its simplicity, security and flexibility and being agnostic to device type and OS.
  - AWS IoT Device Management is agnostic to device type and OS.

Conclusion


Operation and Administration of Access Control in IoT Environments
Safety and Consistency issue
User-to-Device Access Control
Device-to-Device Access Control
1. Introduce/Formal definition of consistency levels for subject attributes.
2. Introduce *refresh* instead of *revoke* for increased safety and availability.
3. Formal definition of consistency levels using refresh.
4. Consistency levels for mutable attributes.
RBAC Administration
1. Proposing RBAC administrative model with Smart home use case.
2. Blockchain-based Enforcement Architecture.
3. Proof-of-concept implementation using smart contracts on top of Ethereum.
Message-Based Access Control
Scenario-Driven Access Control
Formal Specification of a device-to-device operational access control model using a message-based paradigm.
Extension of the message-based to a scenario-driven access control model and define priorities to handle conflicts.
1. AWS-based enforcement architecture
2. proof-of-concept implementation for a smart home use case.

Some Future Directions



Dissertation Publications

Conference/Workshop Papers:

Chapter 3

1. Mehrnoosh Shakarami, and Ravi Sandhu. "Safety and Consistency of Subject Attributes for Attribute-Based Pre-Authorization Systems", In Proceedings of National Cyber Summit (NCS'19), pp. 248-263. Springer, Cham, 2019. **(Published)**

2. Mehrnoosh Shakarami, and Ravi Sandhu. "Refresh instead of revoke enhances safety and availability: A formal analysis." In IFIP Annual Conference on Data and Applications Security and Privacy (DBSec'19), pp. 301-313. Springer, Cham, 2019. **(Published)**

3. Mehrnoosh Shakarami, and Ravi Sandhu. "Safety and Consistency of Mutable Attributes Using Quotas: A Formal Analysis." In 2019 First IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS'19), pp. 1-9. IEEE, 2019. **(Published)**

Chapter 4

4. Mehrnoosh Shakarami, and Ravi Sandhu. "Role-Based Administration of Role-Based Smart Home IoT", In Proceedings of the 2021 ACM Workshop on Secure and Trustworthy Cyber-Physical Systems (SaT-CPS'21), pp. 49-58. 2021. **(Published)**

5. Mehrnoosh Shakarami, James Benson, and Ravi Sandhu. "Blockchain-Based Administration of Access in Smart Home IoT", In Proceedings of the 2022 ACM Workshop on Secure and Trustworthy Cyber-Physical Systems (SaT-CPS'22), 2022. **(Published)**

Chapter 5

6. Mehrnoosh Shakarami, James Benson, and Ravi Sandhu. "Scenario-Driven Device-to-Device Access Control in Smart Home IoT". IFIP Annual Conference on Data and Applications Security and Privacy (DBSec'22), Springer, Cham, 2022. **(To be Submitted at Mid-April)**

Q & A

THANKS FOR TAKING THE TIME TO BE HERE!