An Attribute-Based Access Control for Cloud Enabled Industrial Smart Vehicles

Maanak Gupta, Feras Awaysheh, James Benson, Mamoun Alazab,
Farhan Patwa, and Ravi Sandhu

***Abstract*—Smart cities' vision will encompass connected industrial vehicles, which will offer data-driven and intelligent services to the user. Such interaction within dispersed connected objects, sometimes referred as the Industrial Internet-of-Vehicles (IIoV). The prime motivation of Intelligent Transportation System (ITS) is ensuring the safety of the drivers and offering a comfortable experience to the user. However, such complex infrastructures opens broad attack surfaces to the adversaries, which can remotely exploit and control the critical mechanics in the smart vehicles, including engine and brake systems. Security and privacy concerns are significant barriers to the wide adoption of this revolutionary technology that has to be addressed before a comprehensive implementation of the real vision of ITS. This research is a stepping stone to address access control issues in the IIoV ecosystem and propose a formal Attribute-Based Access Control system (referred to ITS-ABAC$_{\mathrm{G}}$). The proposed model introduces the notion of groups, which are assigned to various smart entities based on the different attributes. It also offers the implementation of fine-grained security policies and considers individualized privacy preferences along with system-wide policies to accept or reject notification, alerts, and advertisements from different participating smart entities. We present the prototype implementation of our proposed model in the Amazon Web Services IoT platform together with extensive performance evaluation, to reflect the practicality and wide-scale adoption of the proposed system.**

***Index Terms*—Intelligent Transportation System, Cloud Computing, Attribute-Based Access Control, Security Policies, Smart Connected Vehicles, Privacy, Industrial IoV**

I. Introduction and Motivation

The vision of industrial IoV which enables ubiquitous data transfer and sharing between vehicles is to offer a safe, efficient and smart travel experience [1]. Several architectures have been proposed to enable the communication and interaction in distributed IoT and IoV systems, which have brought together the unlimited computational capabilities of cloud infrastructures [2], [3], [4] together with a real-time application using edge systems [5]. In the case of ITS, most of the applications are location-centric that also involve dynamic pairing together with continuous movement of smart connected entities such as vehicles. This mobility enables industrial connected vehicles to interact with each other (Vehicle to Vehicle - V2V), with the smart sensor, enable roadside units like traffic signals (Vehicle to Infrastructure - V2I) or with everything (V2X) in real-time to enable information sharing among them. It is imperative that such dynamic and untrusted environment where the communicating entities have no prior trust, only legitimate connected vehicles are allowed to exchange messages, or issue operations at other vehicles and infrastructures enrolled in the system. Cyberattacks can be orchestrated on connected industrial vehicles which can not only compromise one vehicle but the entire fleet, thereby injecting fake messages, data leakages, sensors hacking and remote manipulation, or spoofed over the air updates. To prevent such exploits, formal and foundational security models are needed to be developed and adapted to fit the needs of industrial IoV and control systems.

Security mechanisms such as access control [6], [7] have been extensively used to provide policy-based restricted and authorized access to resources in a system. A fine-grained mechanism like attribute-based access control (ABAC) [8], [9], [10] offer the ability to provide flexible authorization mechanism most applicable in a distributed system like IoT, multi-tenant cloud environments, smart transportation, etc.

This article focuses on access control issues in the industrial ITS ecosystem and proposes a formal attribute-based access control system (referred to ITS-ABAC$_{\mathrm{G}}$). ITS-ABAC$_{\mathrm{G}}$ introduces the notion of groups, which are assigned to various smart entities on the fly based on different attributes, including location, direction, and speed, among others. Its novel implementation provides fine-grained security policies and considers individualized privacy preferences of the user along with system-wide policies to accept or reject notification, alerts, and advertisements from different participating smart entities in the ecosystem. Such security models can also help in understanding the potential of artificial intelligence (AI) to profound impacts on the design and implementation of security solutions within the industrial ITS realm. The interaction between the ITS security mechanisms and AI science will garner insights and ensuring the security and privacy of its infrastructure. Ontologies and AI based expert systems can be created for the proposed ABAC security system which can dynamically evaluate access requests. The proposed fine grained ABAC model has minimal impact on the performance of cloud assisted industrial smart vehicles ecosystem. We also demonstrate our novel security solution as a stand-alone external authorization service in the widely accepted Amazon Web Services[1] (AWS) cloud platform, along with extensive performance evaluation ad analysis. The key **contributions** of this paper are as follows:

Maanak Gupta is with the Department of Computer Science, Tennessee Tech. University, TN, 38505 USA E-mail: mgupta@tntech.edu

Feras Awaysheh is with the CiTIUS Research Center, University of Santiago de Compostela, Spain. Email: feras.awaysheh@usc.es

James Benson, Farhan Patwa, and Ravi Sandhu are with the Institute for Cyber Security and Department of Computer Science, University of Texas at San Antonio, San Antonio, TX, 78249, USA. E-mail: james.benson@utsa.edu, farhan.patwa@utsa.edu, ravi.sandhu@utsa.edu

Mamoun Alazab is with the College of Engineering and IT, Charles Darwin University, Australia. Email: Alazab.M@ieee.org

[1]https://aws.amazon.com/

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2020.3022759, IEEE Transactions on Industrial Informatics

- It identifies security requirements in industrial transportation and highlights existing ITS technologies.
- It presents a fine grained formal ITS-ABAC$_G$ security model along with cloud assisted architecture for ITS access control among industrial smart vehicles and IoV.
- It enforces the proposed architecture and model in AWS to reflect the efficiency and practicality, along with comparative analysis on performance metrics.

The remainder of this paper is organized as follows. Section II briefly highlights some important work in the domain. Section III defines the formal ABAC model (ITS-ABAC$_G$) for cloud-assisted industrial ITS. Section IV demonstrates the implementation of the proposed system in AWS, together with performance evaluation. Section V concludes this paper.

II. Relevant Background and Technologies

There is an extensive discussion in the literature regarding recent advancement, challenges, and opportunities in both vehicle intelligence and Vehicular Ad-hoc Networks (VANETs) enabled systems [11], [12], [13], [14]. VANETs play a critical role in modern ITS, as it exhibits several unique features due to its high mobility nature. Ikram et al [12] discuss this concept and classify several security schemes within the VANETs domain. A cryptographic point of view on the associated security concern with the VANET security was also reported in [13]. Exploring some future trends that shape the research in vehicle intelligence protocols for ITS was the aim of Yasin Firat and M. Amac research in [14]. The adaption of AI and machine learning [15] is imperative in releasing the next-generation vehicle platforms, according to the findings in this research. An in-depth study of anonymous authentication schemes and different trust management models within the vehicle realm was reported in [11], [16]. Recent work in Quality of Service (QoS) related to in time delivery and data dissemination pertinent to vehicluar clouds and fog/edge assisted technologies have been elaborated in [17], [18]. The edge of a network, on the other hand, is also a decisive factor in both V2I [19], [5] and V2V communications [20], where the privacy-preserving systems are compulsory [21] to support industrial intelligent vehicle applications [22], [23]. Also, a fine-grained access policy and collusion prevention in cloud computing is reported [24].Work towards evaluating connected vehicles in cloud was introduced in [25] to meet the requirement of modern ITS.

The role of big data (BD) analytics [26], [27], [28] in the ITS realm is a subject of intensive studies in terms of challenges, opportunities, and carrying out new technical methods. Work in [27] proposed a multi-tier ITS security framework called SITS. This framework classifies the literature solutions, products, and services for validating the usability of the proposed security criteria of vehicular clouds and IoV [29] for a better selection of these criteria among practitioners. European Union supported Cooperative Intelligent Transport Systems (C-ITS) [30], [31] developed a trust model based on PKI to enable integrity and confidentiality of messages exchanged among vehicles. Similar efforts have been witnessed in Security Credential Management System [32] to ensure trustworthiness among vehicles. Although researchers are working extensively in this domain, security policy based solutions are still missing and novel architectures are needed to be deployed to enforce such systems. This paper is an effort in this direction and will foster more similar research.

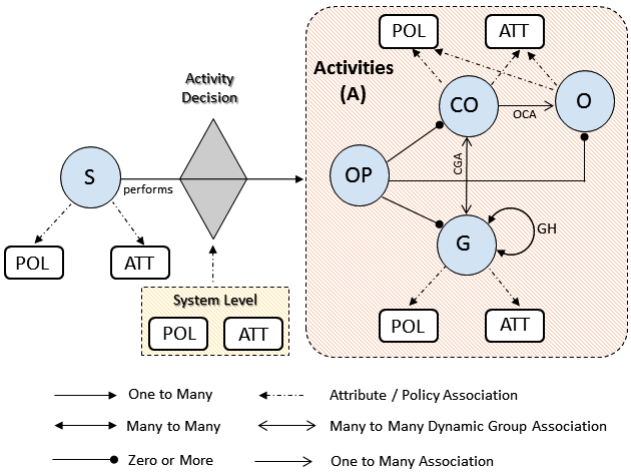


Fig. 1: A Conceptual ITS-ABAC$_G$ Model

III. ABAC Model for Cloud ITS

The proposed ITS-ABAC$_G$ model captures the need of location-specific and time-sensitive applications via cloud-assisted ITS ecosystem for industrial IoV. In this section, we first discuss the model components followed by formal definitions of its entities. Figure 1 represent the abstract model of ITS-ABAC$_G$, and Table I details the formal definitions of its components. This model has the following elements: Sources (S), Clustered Objects (CO), Objects in clustered objects (O), Groups (G), Operations (OP), Activities (A), Authorization Policies (POL), and Attributes (ATT).

A. Model Components

A source (S) starts activities on different connected objects, groups, and applications in the ecosystem. A source can be an application, client, device, clustered object such as a vehicle which are part of the system. Clustered Objects (CO) have several sensors within itself like a smart truck or an industrial vehicle. Such individual sensors in CO are represented as Objects (O) which can be tire pressure sensor, cameras or applications like lane departure system etc. Groups (G) represent a logical collection of COs with same needs and characteristics, for example, location groups, emergency vehicle groups, trucks with same destination etc. Group hierarchy (GH) also exists in the system to support attributes inheritance. Operations (OP) are primitive actions including read, write or notify, alert etc. which also include administrative operations. An activity (A) can be made up of single or multiple operations, and include both operational and administrative activities which can be conducted by different sources. Each activity needs system-defined policies together with user privacy preferences to be evaluated to allow or deny an activity. For instance, a broadcast to vehicles in the locations nearby using location groups can

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2020.3022759, IEEE Transactions on Industrial Informatics

TABLE I: Formal ITS-ABAC$_{\mathrm{G}}$ Model Definitions

Basic Sets and Functions

– S, A, O, CO, OP, G are finite sets of sources, activities, objects, clustered objects, operations and groups respectively .
– ATT defines a set of attributes for entities and with system-wide attributes.
– Range(att) defines a finite set of atomic values for each attribute att in ATT.
– Each attribute att is set or atomic valued, defined by function attType: ATT = {set, atomic}.
– Entities in S, O, G, and CO are mapped to attribute values for every attribute att in ATT. Mathematically,

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

– Individual entities in S, O, CO, and G have associated policies as defined by set POL.
– Each clustered object is mapped to a system group, defined by directG : $\text{CO} \rightarrow \text{G}$.
– Each object is mapped to a clustered object, defined by parentCO : $\text{O} \rightarrow \text{CO}$.
– Group Hierarchy is a partial order relation $\succeq_g$ on G, defined as $\text{GH} \subseteq \text{G} \times \text{G}$,
This is equivalent to a group mapped to set of parent groups, stated as parentG : $\text{G} \rightarrow 2^{\text{G}}$.

Derived Effective Attributes of Clustered Objects, Groups, and Objects

– For each attribute att in ATT such that attType(att) = set:
- $\text{Geff}_{\text{att}} : \text{G} \rightarrow 2^{\text{Range(att)}}$, defined as $\text{Geff}_{\text{att}}(\text{g}_\text{i}) = \text{att}(\text{g}_\text{i}) \cup \left(\bigcup_{\text{g} \in \{\text{g}_\text{j} | \text{g}_\text{i} \succeq_g \text{g}_\text{j}\}} \text{Geff}_{\text{att}}(\text{g}) \right)$.
- $\text{COeff}_{\text{att}} : \text{CO} \rightarrow 2^{\text{Range(att)}}$, defined as $\text{COeff}_{\text{att}}(\text{co}) = \text{att}(\text{co}) \cup \text{Geff}_{\text{att}}(\text{directG}(\text{co}))$.
- $\text{Oeff}_{\text{att}} : \text{O} \rightarrow 2^{\text{Range(att)}}$, defined as $\text{Oeff}_{\text{att}}(\text{o}) = \text{att}(\text{o}) \cup \text{COeff}_{\text{att}}(\text{parentCO}(\text{o}))$.

– For each attribute att in ATT such that attType(att) = atomic:
- $\text{Geff}_{\text{att}} : \text{G} \rightarrow \text{Range(att)} \cup \{\perp\}$, defined as $\text{Geff}_{\text{att}}(\text{g}_\text{i}) = \begin{cases} \text{att}(\text{g}_\text{i}) & \text{if } \forall \text{g}' \in \text{parentG}(\text{g}_\text{i}).\ \text{Geff}_{\text{att}}(\text{g}') = \perp \\ \text{Geff}_{\text{att}}(\text{g}') & \text{if } \exists\ \text{parentG}(\text{g}_\text{i}).\ \text{Geff}_{\text{att}}(\text{parentG}(\text{g}_\text{i})) \neq \perp \text{then} \\ & \text{select parent g}' \text{with } \text{Geff}_{\text{att}}(\text{g}') \neq \perp \text{ updated most} \\ & \text{recently.} \end{cases}$
- $\text{COeff}_{\text{att}} : \text{CO} \rightarrow \text{Range(att)} \cup \{\perp\}$, defined as $\text{COeff}_{\text{att}}(\text{co}) = \begin{cases} \text{att}(\text{co}) & \text{if } \text{Geff}_{\text{att}}(\text{directG}(\text{co})) = \perp \\ \text{Geff}_{\text{att}}(\text{directG}(\text{co})) & \text{otherwise} \end{cases}$
- $\text{Oeff}_{\text{att}} : \text{O} \rightarrow \text{Range(att)} \cup \{\perp\}$, defined as $\text{Oeff}_{\text{att}}(\text{o}) = \begin{cases} \text{att}(\text{o}) & \text{if } \text{COeff}_{\text{att}}(\text{parentCO}(\text{o})) = \perp \\ \text{COeff}_{\text{att}}(\text{parentCO}(\text{o})) & \text{otherwise} \end{cases}$

Authorization Functions (Policies)

– Authorization Function: For each op $\in$ OP, $\text{Auth}_{\text{op}}(\text{s} : \text{S}, \text{ob} : \text{CO} \cup \text{O} \cup \text{G})$ is a propositional logic formula returning true or false, which is defined using the following policy language:
- $\alpha ::= \alpha \wedge \alpha \mid \alpha \vee \alpha \mid (\alpha) \mid \neg\alpha \mid \exists\ \text{x} \in \text{set}.\alpha \mid \forall\ \text{x} \in \text{set}.\alpha \mid \text{set} \triangle \text{set} \mid\ \text{atomic} \in \text{set} \mid\ \text{atomic} \notin \text{set}$
- $\triangle ::= \subset \mid \subseteq \mid \not\subseteq \mid \cap \mid \cup$
- $\text{set} ::= \text{eff}_{\text{att}}(\text{i}) \mid \text{att}(\text{i})$ for att $\in$ ATT, i $\in$ S $\cup$ CO $\cup$ O $\cup$ G $\cup$ {system-wide}, attType(att) = set
- $\text{atomic} ::= \text{eff}_{\text{att}}(\text{i}) \mid \text{att}(\text{i}) \mid \text{value}$ for att $\in$ ATT, i $\in$ S $\cup$ CO $\cup$ O $\cup$ G $\cup$ {system-wide}, attType(att) = atomic

Authorization Decision

– A source $\text{s} \in \text{S}$ is allowed to perform an activity $\text{a} \in \text{A}$, stated as $\text{Authorization}(\text{a} : \text{A}, \text{s} : \text{S})$, 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 ($\text{op}_\text{i} \in$ OP) to be performed by source $\text{s} \in$ S on relevant objects ($\text{x}_\text{i} \in$ CO $\cup$ O $\cup$ G). Formally,
$\text{Authorization}(\text{a} : \text{A}, \text{s} : \text{S}) \Rightarrow \text{Auth}_{\text{op}_1}(\text{s} : \text{S}, \text{x}_1), \text{Auth}_{\text{op}_2}(\text{s} : \text{S}, \text{x}_2), \text{Auth}_{\text{op}_3}(\text{s} : \text{S}, \text{x}_3), \ldots\ldots\ldots\ldots\ldots, \text{Auth}_{\text{op}_\text{n}}(\text{s} : \text{S}, \text{x}_3)$

be generated by a requestor for vehicle pooling notifications. At the same time drivers of vehicles must receive or respond to that request based on individual preferences. An access control system can decide based on such policies to make a decision for such an activity. The ITS-ABAC$_{\mathrm{G}}$ model supports security policies (POL), and attributes (ATT) for different entities like source, clustered objects, objects, and groups, to ensure fine grained access control solution. Such policies include personal privacy preferences together with system wide rules (as shown in Figure 1) which are evaluated together. These policies are set by the system administrators or individual users, and are relatively static in nature as compared to attributes of entities which are more dynamic. These attributes highlight the characteristics of different entities in the system like source, clustered objects, or sensors (objects). Example of such attributes are GPS location, direction, vehicle speed, size, dimensions, company/fleet to which vehicles belong etc. Activities among entities are evaluated based on their attribute, personal preferences and system defined policies to ALLOW or DENY a request. The model expects that no attributes or policies are altered during an activity evaluation process.

B. Model Definitions

Table I represents a set of objects, clustered objects, sources, and groups that can be allocated from a set of singular discrete values (indicated by Range(att)) for an attribute att $\in$ ATT. In this scenario, the attribute is either a set or atomic, which is based on its type and fixed by the attType function. Entities have two value status; single for the atomic and multiple for set value from the attribute range - the single value can also hold a null ($\perp$) value. POL, on the contrary, is the set of ABAC security policies. Based on the preferences and requirements of the system, different cluster objects can be assigned to various groups. For instance, a vehicle is attached to a location group according to its GPS coordinates. An object in a cluster can be attached directly to any group at a similar hierarchy level. This assignment is defined by the directG function in our model, relying on the hierarchy theory to create a systematic hierarchy tree. A clustered object can be assigned to only one parent group to ensure inheritance of attributes as groups inherit their attributes from the parent groups. While industrial vehicles can be accessed through several sources, they have compact sensors and applications. Hence, parentCO function is a one to many mapping which defines the clustered object to which an object is part of. This is based on the following proposition: an object can only belong to one CO while a CO can have various objects. Besides, the group hierarchy GH, can be defined using the following proposition.

PROPOSITION: a partial order relation on G defined by $\succeq_g$, where $g_1 \succeq_g g_2$ illustrate g_1 is sub group of g_2 and g_1 gets all the attributes of parent g_2. For a child group, parentG function defines set of parent groups, as shown in the group hierarchy in Figure 1.

While introducing groups offer many advantages, among them the ease of administration. Using a single administrative operation, a member of a cluster can assign or remove many attributes. Since attributes inheritance is possible from parent groups to sub groups, therefore, when an attribute is set valued its effective attribute value for att for a group $\mathrm{g_i}$ (denoted by $\mathrm{Geff_{att}(g_i)}$) can be calculated as the union of direct values for att and the effective value for att from all parent groups. This is a well formed definition as $\succeq_g$ is a partial order. The base for this recursive definition will be $\mathrm{Geff_{att}(g_j)}$ = att($\mathrm{g_j}$), for a maximal group $\mathrm{g_j}$. $\mathrm{COeff_{att}}$ defines the effective attribute value of a clustered object for att, which can be computed with the direct and the inherited values from the member group as stated by directG. Likewise, sensors in vehicles can inherit attributes from the vehicle itself (e.g., make, model, location) besides direct attributes as function $\mathrm{Oeff_{att}}$ calculates the attributes of objects. Union operation will be adequate to set-valued attributes, however, it is not valid for atomic attributes. In this model, the recent assigned attributes of parent groups will overwrite the non null values of child groups.

For each operation op $\in$ OP, authorization function is defined, which are fine grained policies defined in the system. POL is the set of all authorization functions, $\mathrm{Auth_{op}(s : S, ob : CO \cup O \cup G)}$ that define the conditions under which source s $\in$ S can execute operation op $\in$ OP on object ob $\in$ CO $\cup$ O $\cup$ G. Such policies include privacy preferences set by users for an individual clustered object, objects, and groups or can be system-wide by security administrators. The conditions can be specified as a propositional logic formula using policy language stated in Table I. A set of policies should be complied and checked to allow or deny an activity. Authorization function, $\mathrm{Authorization(a : A, s : S)}$, defines the all the policies which must validated to allow an activity a $\in$ A by source s $\in$ S. The proposed model enables user personalized policies together with attributes and dynamic groups assignment to make activity decision. It is expected that the shared attributes from different entities are trusted and validated. The proposed approach uses shared information, like location coordinates, sent by a industrial vehicles to make an access control and notification decisions.

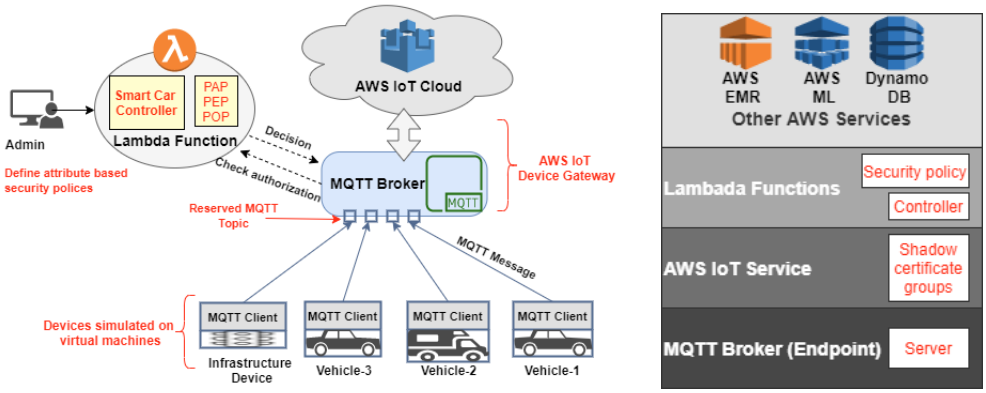
Algorithm 1 represents the grouping process of industrial connected vehicle authorization in the cloud-assisted system. The input of this algorithm is the list of entities and attributes associated with the system. The process consists of three main stages; at first, it starts with grouping the atomic values attributes. Next, the algorithm defines attributes to be a set

Algorithm 1 Fine-grained grouping process of industrial connected vehicle authorization in cloud assisted system (FGAG).

Let μ be the union of the *entities*
Let ϕ = be a special set of $\langle$*G, CO, and O*$\rangle$
Let att = atomic if att $\cup\{\perp\}$ = true else att = set.
Input: entities and attributes associated with the system
Output: assign fine-grained access police for each group
for $\forall[\mu \rightarrow$*μ:attType(att) = atomic]* $\in attribute$ **do**
  1- atomic grouping:
  for $\mathrm{Geff_{att}}$: G $\rightarrow$Range(att) $\cup\{\perp\}$, *defined as* $\mathrm{Geff_{att}(g_i)}$ **do**
    if $\forall \mathrm{g'} \in \mathrm{parentG(g_i)}$. $\mathrm{Geff_{att}(g')} = \perp$ **then**
      $att(g_i) \rightarrow \mathrm{parentG(g_i)}$. $\mathrm{Geff_{att}(g')} = \perp$
    else
      $\mathrm{Geff}_{att}(g^{'}) \rightarrow \mathrm{Geff}_{att}(g') \neq \perp$ *updated most*
    end
  end
end
for $\forall$ *attribute* μ *in ATT such that attType(att) = set* **do**
  2- set grouping:
  $\mu : \phi \rightarrow 2^{\mathrm{Range(att)}}, defined as$
  $\mu(\phi) = att(\phi) \cup (\bigcup_{\phi \in \phi' | \phi \succeq_{\phi'} \phi'} \mu(\phi))$
end

for $\forall$ *op* $\in OPAuth_{op}(s : S, ob : CO \cup O \cup G)$ **do**
  3- Police Authorization Function
  $\alpha ::= \alpha \wedge \alpha \mid \alpha \vee \alpha \mid (\alpha) \mid \neg\alpha \mid \exists$ x $\in$ set.$\alpha \mid \forall$ x $\in$ set.$\alpha \mid$
  if *set* $\triangle$ *set* $\mid\mid$ atomic $\notin$ set **then**
    for $\mu \in$ ATT, where attType(att) = set
    set ::= $\mathrm{eff_{att}(i)} \mid \mathrm{att(i)}$
  else
    for $\mu \in$ ATT, where attType(att) = atomic
    atomic ::= $\mathrm{eff_{att}(i)} \mid \mathrm{att(i)} \mid$ value
  end
end

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2020.3022759, IEEE Transactions on Industrial Informatics



(a) Overall Architecture (b) AWS Services

Fig. 2: Implementation Architecture

value group. The authorization functions of policies take place in the third phase which will return true or false with a propositional logic formula. We assume that μ be the union of the entities, while ϕ is defined as a special set of the variable store the value of entities. In the first stage, if the μ can fulfill all target constraints, the group elements are evaluated and assign atomic. Otherwise, the process moves to the next policy and update the most recent policy. Following the attributes are checked for the set values. At the operation stage, Algorithm 1 assigns the police function by comparing the value of atomic and set attributes of the the target element. If the request can fulfill all target constraints, the set elements are evaluated and updated. Apparently, if a single condition is not satisfied, the returned value is false, and the user's request is not granted access. Finally, the algorithm continues with the remaining loop and returns the value of the final group. A source is authorized to complete an activity stated as permission if the required policies needed to allow the activity are included and evaluated to make the final decision. These multi-layer policies need to be evaluated for every operation to be executed by the source on relevant objects.

IV. ITS-ABAC$_G$ Model Implementation in AWS

This section presents a proof of concept implementation of proposed ITS-ABAC$_G$ model using AWS IoT service[2]. The prototype implementation highlights how multi-layer security policies and location groups assignment can be realized in AWS. We simulated real smart cars and infrastructure using IoT things. In the implementation, no long term vehicular data was stored in remote cloud, which pacifies privacy concerns of users and fosters large-scale adoption among practitioners.

A. System Architecture

The complete architecture of implemented prototype using AWS IoT cloud service is shown in Figure 2. We simulated smart vehicles and infrastructures as VMs having a client MQTT. These VMs send MQTT messages to a central broker in AWS. In addition, a custom end point is provided to connect devices with AWS IoT services, with a REST API at the endpoint for every connected device. AWS IoT provides a MQTT broker which allows devices with clients to subscribe and publish to reserved and secure topics to communicate messages with all connected devices through the central cloud. These reserved topics allow a device to get, update or delete information in the device shadow. As communication with the reserved topics need permissions, it ensures only authorized devices can communicate. AWS Lambda[3] function has been used to enforce ABAC policies [33] defined with the proposed model. Figure 2b shows details of AWS cloud components, reflecting where device shadows[4] , certificates[5] and groups[6] are created in AWS IoT with MQTT broker acts like a server, offering a client-server architecture for the proof of concept.

B. Use Case Scenarios

Location centric notification and services are an integral part of ITS ecosystem. Our implemented use-cases satisfy cloud-assisted real-world applications using the group's hierarchy. Enforced security policies cater to the following scenarios:

Deer Threat Alerts - Sensors in smart city and ITS deployment can notice surroundings to generate notifications for groups relevant to the changes. This use case deploys a roadside sensor that observes a deer in the vicinity and modifies the Deer_Threat attribute of the sensor to ON, of the corresponding location group. This change in the attribute value triggers an alert to all the members of the location group including the smart vehicles. This implementation can be extended for accident alerts, over speeding cars, worker on road alerts or for marketing purposes also.

Pooling Notifications - In this scenario, a commuter is requesting a ride to Location-A using his/her mobile app. The user generates a pooling request to nearby smart cars/vehicles going to the same destination. AWS cloud receives the request to find out the appropriate vehicles using the location of the

[2]https://aws.amazon.com/iot/

[3]https://aws.amazon.com/lambda/

[4]https://docs.aws.amazon.com/iot/latest/developerguide/iot-device-shadows.html

[5]https://docs.aws.amazon.com/iot/latest/developerguide/iot-security-identity.html

[6]https://docs.aws.amazon.com/iot/latest/developerguide/thing-groups.html

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2020.3022759, IEEE Transactions on Industrial Informatics

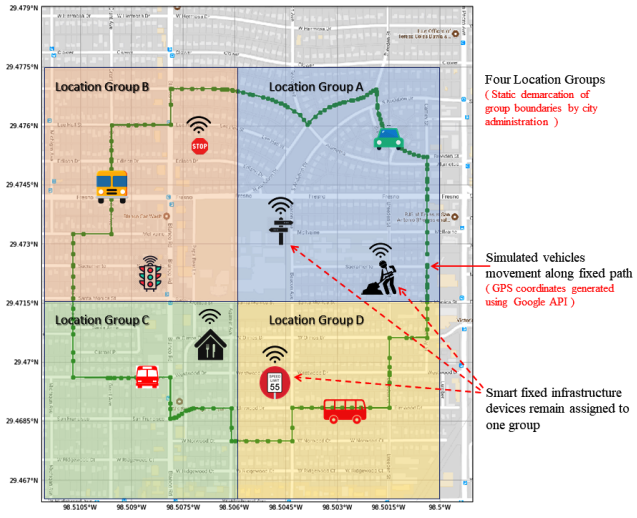


Fig. 3: A simulation of vehicles route and location groups

```
('Received new coordinates from:', 'Vehicle-1')
Sun May 27 02:56:30 2018
  Location A
     Car-A : [u'Vehicle-1', u'Vehicle-2', u'Vehicle-13',
     Bus-A : [u'Vehicle-10', u'Vehicle-42', u'Vehicle-49'
  Location B
     Car-B : [u'Vehicle-9', u'Vehicle-27', u'Vehicle-50',
     Bus-B : [u'Vehicle-6', u'Vehicle-11', u'Vehicle-35',
  Location C
     Car-C : [u'Vehicle-3', u'Vehicle-4', u'Vehicle-8',
     Bus-C : []
  Location D
     Car-D : [u'Vehicle-14', u'Vehicle-45', u'Vehicle-31'
     Bus-D : [u'Vehicle-5']
```

Fig. 4: Snapshot of Table Showing Dynamic Groups and Associated Connected Vehicles at One Point of Time

```
{
  "Deer_Threat": {
     "Source": {
        "Sensor-X": {
           "Location": {
              "Location-A": {"Group": ["Location-A"]},
              "Location-B": {"Group": ["Location-B"]}
           }
        },
     }
  },
  "car_pool_notification": {
     "Source": {
        "Location-A": {
           "destination": {
              "Location-A": {"Notification": ["Car-A"]},
              "Location-B": {"Notification": ["Car-A", "Car-B", "Car-C"]},
              "Location-C": {"Notification": ["Car-C", "Car-D"]},
              "Location-D": {"Notification": ["Car-A", "Car-C", "Car-D"]}
           }
        },
     }
  }
```

Policy Operation
Source Attributes
Object Attributes

Fig. 5: Snippet of Attribute Based Policies in AWS

user along with other attributes, including their current group. Also, to extend the case, not all the member vehicles of the groups may be part of the car pooling service or are barring specific requests due to previous experience or user ratings. Personal user preferences are also checked before notification is generated for a prospective customer.

C. Proof of Concept

The implementation of ITS-ABAC$_G$ involves two steps. The first part includes the administrative phase, and the second includes the operational phase. The administrative aspect deals with the development of hierarchical groups in geography, assigning moving smart cars to various defined groups, inheriting attributes and alerts from the assigned groups, and also the change in the attributes of different entities in the system. An administrator pushes some of these while others depend on the environmental conditions. The operational aspect deals with the authorization and activity control including deployment and enforcement of the ABAC policies. It deals with how groups can be used to ensure the relevance of notification and alerts to authorized entities. Both of these phases require multi-layer authorization security policies. The implementation involves development of ABAC policy decision (PDP) and enforcement point (PEP) [33] together with our deployed external policy evaluation engine, which is attached to AWS IoT service to enforce ABAC authorization.

Administrative Phase: A group hierarchy is created in AWS IoT services. It has three levels of hierarchy, starting with the County-XYZ at the topmost level, which is divided into four separate non-overlapping location groups, Location-A, B, C, and D. These location groups, in turn, have two subgroups each, one for car and other for the bus to reflect what kind of vehicle can be part of that group. We simulated 50 moving smart cars using a python script designed to send MQTT messages to their corresponding virtual vehicles (shadows). These messages contain the GPS coordinates, which are generated with Google API[7]), iterating over green dotted line shown in Figure 3. The county-wide area is separated into four different locations, and a moving smart vehicle can be a member of any one of the subgroups in these location groups, meaning the vehicle can be a car or a bus. The fixed roadside sensor devices remain part of the same location group all the time. Let us assume that Vehicle-1 has a current location in Location-D, and sends the following MQTT message:

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

to its shadow AWS topic. With this message, the vehicle becomes part of the Location-A group, and because it is a regular car, it is assigned to the Car-A subgroup in Location-A, as clarified in Figure 4. To keep track the vehicle's movement among locations, the GPS coordinates and additional relevant attributes are sent to the cloud, and the table is continuously updated. Vehicle type, together with the current coordinates of the moving vehicle, are used in on-the-fly assignment of the vehicle to the relevant group. We implemented this functionality via stand-alone service using Lambda function and Boto[8] AWS SDK for Python. Also, in deer threat alert scenario, a location sensor is simulated, which helps to update the 'Deer_Threat' attribute of the corresponding location group, and generates a notification for all the member vehicles.

An attribute-based policy is defined to control which sensors are allowed to change the 'Deer_Threat' attribute of location groups. Figure 5 shows the snippet of policies implemented

[7]https://cloud.google.com/maps-platform/

[8]https://aws.amazon.com/sdk-python/

in our prototype. The JSON format policy file defines a set of policies for two operations: one for Deer_Threat and another for car_pool_notification, as marked by the red box. The blue box specifies the attributes of the source, also known as the initiator of operation request, whereas the green box specifies the attributes of the target object to which the action is requested. Our defined policy for Deer_Threat operation checks that a motion sensor with name = 'Sensor-X' and currently a member of the group Location-A can update the value of attribute Deer_Threat for location group Location-A only. If the sensor is relocated to Location-B, it can update the same attribute for the Location-B group only. This policy ensures that the sensor must be in that location group for which it is updating Deer_Threat attribute, which is needed security requirement as we do not want adversaries to change attributes and trigger unwanted alerts for vehicles remotely.

A mobile smart vehicle continuously updates its current location coordinates to AWS shadow. These coordinates, together with the attributes of moving vehicles and groups, help to determine if a vehicle can be a member. It the implemented policy approves the vehicle to join the group, both the group and the vehicle are notified, and the new member vehicle gets all the attributes of its new group. Any change in the values of the attributes of the group is also propagated to the current member vehicles. The attribute inheritance from the parent group to child group is implemented via `update_thing_group` and `update_thing` methods.

In the implemented use-cases, attributes inheritance happens between Location-A and both subgroups Car-A and Bus-A, and also to the member vehicles in Car-A and Bus-A. Henceforth, in case an attribute 'Deer_Threat' is changed to value ON in group Location-A, its new attributes using Boto `describe_thing_group` command are:

```
{'Center-Latitude': '39.3256',
'Center-Longitude': '-89.998',
 'Deer_Threat': 'OFF'}
```

This inherits the attributes to Car-A child group whose effective attributes will now be:

```
{'Center-Latitude': '39.3256',
'Center-Longitude': '-89.998',
'Deer_Threat': 'OFF', 'Location': 'B'}
```

As shown in Figure 4, both Vehicle-1 and Vehicle-2 are members of Car-A sub-group, therefore, the effective attributes of Vehicle-2 are:

```
{'Center-Latitude': '39.3256',
'Center-Longitude': '-89.998',
'Deer_Threat': 'OFF', 'Location': 'B',
'Type': 'Car', 'VIN': '9246572903752',
'thingName': 'Vehicle-2'}
```

Operational Phase: In this phase, policies enforce restriction of alerts, notification, and services with various response time to regulation signals in the IIoV environment [34]. This evaluation can involve single or multi-level policies together with user privacy preferences, to make final decision about an activity. In the case of a pooling scenario, the enforced security policies limit the alerts to a subset of vehicles to which these requests were relevant. The requesting user publishes current and future locations in an MQTT message to reserved AWS shadow topic `$aws/things/Requestor/shadow/update`. Based on this the subgroup to which request is sent is determined.

```
{"state": {"reported": {"policy":
"car_pool_notification",
 "source": "Location-A",
 "destination": "Location-B"}}}
```

In case of pooling request, the policy as shown in Figure 5 assumes that if the current location of the requesting user is in group 'Location-A,' and is requesting car pool for another location which is part of 'Location-A,' then only vehicle which is part of subgroup 'Car-A' are must be advertised. At the same time, in case the destination is in 'Location-B,' then all the vehicles which are a member of Car-A, Car-B, and Car-C must be notified. This approach enforces security policy limits the number of vehicles which should be notified, in comparison to all the vehicles enrolled in the system based on the attributes. This implemented use case reflects how location-centric ITS services can be enforced. Similar to other location-based notification, including alerts and marketing services can be limited based on these attribute-based policies.

TABLE II: Policy Enforcement Time (in Milliseconds)

Number of Action Requests	Policy Enforcer Execution Time	
	Deer-Threat	**Car-Pool**
10	0.0813	0.0922
20	0.1551	0.2003
30	0.2369	0.2872
40	0.3150	0.3953
50	0.3903	0.5196

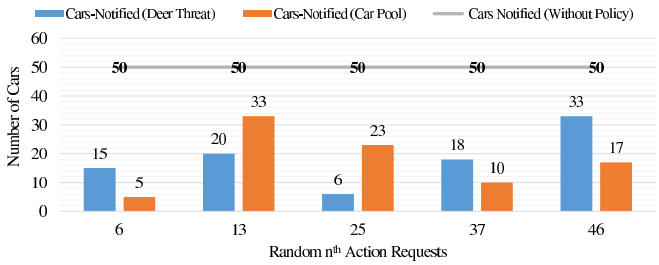
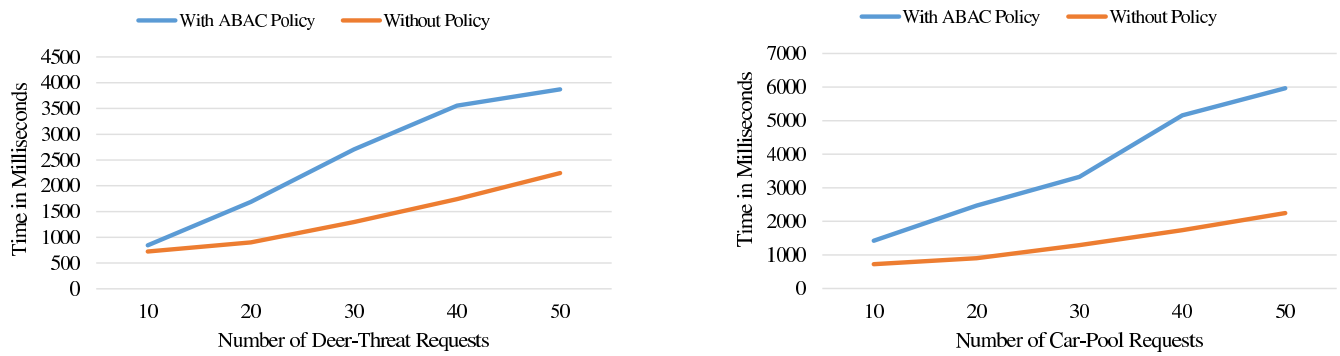


Fig. 6: Comparing the Scoping and Relevance of Alerts with and without Policy

D. Performance Evaluation Metrics and Analysis

It is essential to appraise the performance of the proposed ITS-ABAC$_G$ model where metrics are discussed with the objective of understanding the impact of stand-alone external service in order to have an industrial smart-vehicle ecosystem that has enhanced security features. AWS is used to assess the performance of the model. Fifty moving vehicles have been stimulated for emulating the ITS environment. These

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2020.3022759, IEEE Transactions on Industrial Informatics



(a) Deer Threat Use Case

(b) Car Pool Use Case

Fig. 7: A Comparison of the Performance with and without ABAC policy

vehicles are made to spread arbitrarily, with the help of a smart-vehicle controller, over the four location sub-groups which are predefined and already exhibited in the Figure 3. Two different types of metrics have been provided. The first one describes the implementation time required to enforce the policy enforcer's security policies (in Figure 5). The second one compares the situation where ABAC policies are implemented with a situation where there are no policies executed. The execution time of the engine for policies created for pooling as well as deer-threat use cases is described in Table II. This time (in milliseconds) reveals the time that is consumed for evaluating the executed policies for various operations. The table aggregates the evaluation time for the policy regarding different action requests. For instance, 0.2003 ms is the time which is required for evaluating the policy, which is applicable for twenty arbitrary pooling requests. When used in the smart-vehicle system enabled by cloud, the engine has minimal impact and is found to be very efficient.

Further, the scope, as well as the relevance of the alerts that is received by the smart cars is dependent on the effect of execution of the policies in the system. One of the major advantages provided by the connected vehicles is that they can have alerts and on-board advertisements, which further offer safety as well as convenience. It is crucial to ensure that irrelevant notifications do not bother the drivers and distract their attention. Proposed ABAC policies should be effective in order to ensure the same. The number of cars that have received notifications regarding deer-threat as well as the pooling notifications irrespective of the implementation of the policy is shown in Figure 6. In a scenario where no policy is implemented, all the vehicles (in this case 50), without considering their location. Next, the driver's preferences, get the notifications when a random request is created. However, enforcement of cloud-based policies makes sure that notifications are relevant to the vehicles. For instance, in Figure 6, on 25th request shows that, instead of all the vehicles, the notification of a car-pool request reached only 23 vehicles and one of them was almost 20 miles away from the person who requested. The calculation of the subset of the vehicles depends on the number of vehicles present in the location groups. Similarly, in case of deer-threat alerts, only those cars which are close enough to the deer get the alerts. It is vital to notice that in both cases, notified cars are clubbed together, and the same is represented in Figure 6. Although the nth request that represents the scenario of deer-threat is entirely different from the nth request that represents the scenario of car-pool. These metrics primary objective is to reveal the impact of the policies on the relevance and scope of the notifications directed towards the target vehicles.

Performance graphs, as shown in Figure 7, evaluate the execution time when ABAC policy is executed (blue line) against implemented no policy (orange line) for the two use-cases. The metric considers the time to calculate the number of vehicles which are notified with and without the implemented ABAC policy. X-axis in graph describe the total execution requests, meaning how many times deer-threat (Figure 7a) or pool (Figure 7b) alerts are generated. Y-axis defines the overall time (in milliseconds) when the AWS Lambda function gets notification or access request in central cloud till the time the number of vehicles which have been notified is recorded in the system. Because in our proof of concept, the implemented ABAC policies shown in Figure 5 definition for each access request in both the scenarios are very similar, it is observed that the number of access requests proportionately increases the number of times the policy is evaluated, which impacts the overall evaluation time of the policies. Some variations in blue and orange lines, as shown in graphs Figure 7 are because of AWS API endpoint calls being made from Lambda function to measure the number of vehicles alerted in both the cases. The developed external policy engine has a minor impact on the performance (as shown with blue line) as opposed to without policy implementation. Nevertheless, we assume that this system, when implemented in broad city scenarios, this enforcement time will be subsumed by cloud supported ITS alerts and notifications to all industrial vehicles as compared to a subset of vehicles allowed by the policy evaluation system.

The ITS-ABAC$_G$ model is demonstrating how to enable the relevance of notices and alerts of service, which works perfectly with some trade-off. The paper proposed the specification and introduction of ABAC policies in a cloud-assisted

smart vehicle environment without focusing on any one cloud platform. It is expected that a more real world experimental setup is needed to capture practical needs to deploy this system in vast settings and to perform more detailed stress test spreading wide geography and large number of vehicles. It should be noted that AWS is one of the cloud-based platforms to realize the proposed model and similar prototype can be implemented in other cloud computing services including Microsoft Azure[9], Google Cloud[10] or Openstack[11].

V. Conclusion

This research develops an attribute-based access control model for cloud-assisted ITS, to enable location-specific and in time notifications and alerts in smart transportation ITS environment. The proposed security system, in addition to the fine-grained ABAC model, introduces the element of groups which are dynamically assigned to moving vehicles based on their attributes. This policy-based solution considers both the system's extensive rules in addition to the individualized privacy preferences to allow or deny various activities in the system. Multiple real-world use-cases have been implemented together with a prototype implementation in AWS to reflect the practical usability of the solution. For the future, we plan to extend this model to offer in-vehicle access control security solutions, to provide trust-based risk-aware adaptive models. Also, it is primitive to complement location privacy-preserving mechanisms, including homomorphic encryption, to anonymize the real-time location and mitigate privacy concerns of the user. It is also expected to provide a V2X edge assisted solution for trusted communication, which needs further investigation.

Acknowledgment

This work is partially supported by NSF CREST Grant HRD-1736209.

References

[1] Z. Zhou *et al.*, "Social big-data-based content dissemination in internet of vehicles," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 2, pp. 768–777, 2018.

[2] M. Aazam and et al, "Cloud of things: Integrating internet of things and cloud computing and the issues involved," in *Proc. of IBCAST*, Jan 2014, pp. 414–419.

[3] R. Lea and M. Blackstock, "City hub: A cloud-based iot platform for smart cities," in *Proc. of IEEE CloudCom*, Dec 2014, pp. 799–804.

[4] M. Gupta and R. Sandhu, "Authorization framework for secure cloud assisted connected cars and vehicular internet of things," in *Proc. of ACM SACMAT*, 2018, pp. 193–204.

[5] M. Gupta, J. Benson, F. Patwa, and R. Sandhu, "Secure V2V and V2I communication in intelligent transportation using cloudlets," *arXiv preprint arXiv:2001.04041*, 2020.

[6] R. S. Sandhu and P. Samarati, "Access control: principle and practice," *IEEE communications magazine*, vol. 32, no. 9, pp. 40–48, 1994.

[7] D. F. Ferraiolo *et al.*, "Proposed NIST standard for role-based access control," *ACM Transactions on Information and System Security (TISSEC)*, vol. 4, no. 3, pp. 224–274, 2001.

[8] V. C. Hu, D. R. Kuhn, and D. F. Ferraiolo, "Attribute-based access control," *IEEE Computer*, no. 2, pp. 85–88, 2015.

[9]https://azure.microsoft.com/en-us/services/iot-hub/

[10]https://cloud.google.com/iot-core/

[11]https://www.openstack.org/

[9] X. Jin *et al.*, "A unified attribute-based access control model covering DAC, MAC and RBAC," in *Proc. of DBSec*. Springer, 2012, pp. 41–55.

[10] S. S. L. Chukkapalli *et al.*, "A smart-farming ontology for attribute based access control," in *6th IEEE International Conference on Big Data Security on Cloud (BigDataSecurity 2020)*, 2020.

[11] Z. Lu, G. Qu, and Z. Liu, "A survey on recent advances in vehicular network security, trust, and privacy," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 2, pp. 760–776, 2018.

[12] I. Ali *et al.*, "Authentication and privacy schemes for vehicular ad hoc networks (VANETs): A survey," *Vehicular Communications*, 2019.

[13] M. N. Mejri, J. Ben-Othman, and M. Hamdi, "Survey on vanet security challenges and possible cryptographic solutions," *Vehicular Communications*, vol. 1, no. 2, pp. 53–66, 2014.

[14] Y. F. Payalan and M. A. Guvensan, "Towards next-generation vehicles featuring the vehicle intelligence," *IEEE Transactions on Intelligent Transportation Systems*, 2019.

[15] U. Ahmad, H. Song, A. Bilal, M. Alazab, and A. Jolfaei, "Securing smart vehicles from relay attacks using machine learning," *The Journal of Supercomputing*, pp. 1–18, 2019.

[16] F. M. Awaysheh, J. C. Cabaleiro, T. F. Pena, and M. Alazab, "Poster: A pluggable authentication module for big data federation architecture," in *Proc. of the ACM SACMAT*, 2019, pp. 223–225.

[17] M. Shojafar, N. Cordeschi, and E. Baccarelli, "Energy-efficient adaptive resource management for real-time vehicular cloud services," *IEEE Transactions on Cloud computing*, vol. 7, no. 1, pp. 196–209, 2016.

[18] N. Cordeschi, D. Amendola, M. Shojafar, and E. Baccarelli, "Distributed and adaptive resource management in cloud-assisted cognitive radio vehicular networks with hard reliability guarantees," *Vehicular Communications*, vol. 2, no. 1, pp. 1–12, 2015.

[19] S. Ghane, A. Jolfaei, L. Kulik, K. Ramamohanarao, and D. Puthal, "Preserving privacy in the internet of connected vehicles," *IEEE Transactions on Intelligent Transportation Systems*, 2020.

[20] S. Darbha, S. Konduri, and P. R. Pagilla, "Benefits of V2V communication for autonomous and connected vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 5, pp. 1954–1963, 2018.

[21] A. Kayes *et al.*, "Achieving security scalability and flexibility using fog-based context-aware access control," *Future Generation Computer Systems*, vol. 107, pp. 307–323, 2020.

[22] W. Liu and Y. Shoji, "DeepVM: RNN-based vehicle mobility prediction to support intelligent vehicle applications," *IEEE Transactions on Industrial Informatics*, 2019.

[23] M. Gupta, J. Benson, F. Patwa, and R. Sandhu, "Dynamic Groups and Attribute-Based Access Control for Next-Generation Smart Cars," in *Proc. of the ACM CODASPY*, 2019, pp. 61–72.

[24] J. Li *et al.*, "An efficient attribute-based encryption scheme with policy update and file update in cloud computing," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 12, pp. 6500–6509, 2019.

[25] M. Aladwan *et al.*, "TrustE-VC: Trustworthy Evaluation Framework for Industrial Connected Vehicles in the Cloud," *IEEE Transactions on Industrial Informatics*, 2020.

[26] L. Zhu, F. R. Yu, Y. Wang, B. Ning, and T. Tang, "Big data analytics in intelligent transportation systems: A survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 1, pp. 383–398, 2018.

[27] F. Awaysheh, J. C. Cabaleiro *et al.*, "Big data security frameworks meet the intelligent transportation systems trust challenges," in *Proc. of IEEE TrustCom/BigDataSE*, 2019, pp. 807–813.

[28] F. Awaysheh *et al.*, "Next-generation big data federation access control: A reference model," *Future Generation Computer Systems*, 2020.

[29] M. Aladwan *et al.*, "Common security criteria for vehicular clouds and internet of vehicles evaluation and selection," in *Proc. of IEEE TrustCom/BigDataSE*, 2019, pp. 814–820.

[30] E. Union, *Certificate Policy for Deployment and Operation of European Cooperative Intelligent Transport Systems (C-ITS)*, 2017. [Online]. Available: https://ec.europa.eu/transport/sites/transport/files/c-its_certificate_policy_release_1.pdf

[31] European Union, "Security Policy & Governance Framework for Deployment and Operation of European Cooperative Intelligent Transport Systems (C-ITS)," https://ec.europa.eu/transport/sites/transport/files/c-its_security_policy_release_1.pdf, 2017, accessed: 2020-03-11.

[32] *Connected Vehicles and Your Privacy*, 2014. [Online]. Available: https://www.its.dot.gov/factsheets/pdf/Privacy_factsheet.pdf

[33] V. C. Hu *et al.*, "Guide to attribute based access control (ABAC) definition and considerations," *NIST Publication*, vol. 800-162, 2014.

[34] A. Bilh, K. Naik, and R. El-Shatshat, "Evaluating electric vehicles' response time to regulation signals in smart grids," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 3, pp. 1210–1219, 2018.

Maanak Gupta is an assistant professor in computer science at Tennessee Technological University, Cookeville, USA. He received M.S. and Ph.D. in computer science from the University of Texas at San Antonio (UTSA). He has also worked as a postdoctoral fellow at the Institute for Cyber Security (ICS) at UTSA. His primary area of research includes security and privacy in cyber space focused in studying foundational aspects of access control and their application in technologies including cyber physical systems, cloud computing, IoT and Big data. He has worked in developing novel security mechanisms, models and architectures for next generation smart cars, smart cities, intelligent transportation systems and smart farming. He holds a B.Tech degree in computer science and engineering from Kuruskhetra University, India, and M.S. degree in information systems from Northeastern University, Boston.

Feras M. Awaysheh holds a PhD. in Big Data and Cloud Computing from the University of Santiago de Compostela, Spain. He obtained MSc. Degree from New York Institute of Technology (NYIT) With Honor in 2010, majoring in Information, Computer, and Network Security. Currently, he is a researcher at the CiTIUS research center, Spain, and a visiting fellow at the University of Edinburgh, UK. His main research interest includes large-scale distributed systems and Big Data analytics in general. Besides, developing and running software reliably in production for resource allocation (on-premises and cloud-based clusters), and middlewares for load-balancing and security solutions in HPC, Cloud, IoT, and Big Data deployment architectures.

James Benson received his B.Sc. and M.Sc. Physics degree from Clarkson University in 2007 and 2009 respectively and his M.Sc. Electrical Engineering degree from the University of Texas at San Antonio (UTSA) in 2016. He has worked at the Texas Renewable Energy Institute (TSERI) and Open Cloud Institute (OCI) at UTSA assisting with data analytics and various research projects. He is currently working as a Technology Research Analyst II with the Institute for Cyber Security (ICS) and the Center for Security and Privacy Enhanced Cloud Computing (C-SPECC) at UTSA. His research interests include cyber physical systems, cloud computing, and automation.

Mamoun Alazab is an Associate Professor at the College of Engineering, IT and Environment at Charles Darwin University, Australia. He received his PhD degree in Computer Science from the Federation University of Australia, School of Science, Information Technology and Engineering. He is a cyber security researcher and practitioner with industry and academic experience. Alazab's research is multidisciplinary that focuses on cyber security and digital forensics of computer systems with a focus on cybercrime detection and prevention. He has more than 150 research papers in many international journals and conferences. He is a Senior Member of the IEEE. He is the founding chair of the IEEE Northern Territory (NT) Subsection.

Farhan Patwa received his B.Sc. and M.Sc. in Electrical Engineering from University of Texas at Arlington. He is a systems engineer with over 20 years of professional experience working in the telecom industry, cloud computing and software security solutions. He has worked for Nortel and Ericsson leading projects for high capacity test of their 3G and 4G wireless telecom products. He currently works for Wind River Systems, designing embedded security solutions. He also works part-time as the Associate Director and Chief Architect at the Institute for Cyber Security at the University of Texas at San Antonio.

Ravi Sandhu is the founding Executive Director and Chief Scientist at the Institute for Cyber Security at the University of Texas at San Antonio, TX, where he holds the Lutcher Brown Endowed Chair in Cyber Security. He is a fellow of the ACM, IEEE and AAAS and an inventor on 30 patents. He was the past Editor-in-Chief of the IEEE Transactions on Dependable and Secure Computing, past founding Editor-in-Chief of the ACM Transactions on Information and System Security and a past Chair of ACM SIGSAC. He founded ACM CCS, SACMAT and CODASPY, and has been a leader in numerous other security conferences. His research has focused on security models and architectures, including the seminal role-based access control model. His papers have accumulated over 40,000 Google Scholar citations, including over 9,000 citations for the RBAC96 paper.