




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

Maanak Gupta , Feras M. Awaysheh , James Benson, Mamoun Alazab , *Senior Member, IEEE*, Farhan Patwa, and Ravi Sandhu, *Fellow, IEEE*

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 are sometimes referred as the industrial Internet-of-Vehicles (IIoV). The prime motivation of an 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 article 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_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 to reflect the practicality and wide-scale adoption of the proposed system.

Index Terms—Attribute-based access control (ABAC), cloud computing, Industrial Internet-of-Vehicles (IIoV),

Intelligent Transportation System (ITS), privacy, security policies, smart-connected vehicles.

I. INTRODUCTION AND MOTIVATION

THE VISION of industrial Internet-of-Vehicles (IIoV), 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]–[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]–[10] offers the ability to provide flexible authorization mechanism most applicable in a distributed system like IoT, multitenant cloud environments, smart transportation, etc.

This article focuses on access control issues in the industrial ITS ecosystem and proposes a formal ABAC system (referred to ITS-ABAC_G). ITS-ABAC_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

Manuscript received May 20, 2020; revised July 27, 2020 and August 18, 2020; accepted September 1, 2020. Date of publication September 8, 2020; date of current version March 5, 2021. This work was supported in part by the NSF CREST Grant HRD-1736209. Paper no. TII-20-2182. (Corresponding author: Maanak Gupta.)

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

Feras M. Awaysheh is with the CiTIUS Research Center, University of Santiago de Compostela, Santiago de Compostela 15705, Spain (e-mail: 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, Casuarina, NT 0810, Australia (e-mail: alazab.m@gmail.com).

Color versions of one or more of the figures in this article are available online at <https://ieeexplore.ieee.org>.

Digital Object Identifier 10.1109/TII.2020.3022759

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 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 Services¹ (AWS) cloud platform, along with extensive performance evaluation and analysis. The key *contributions* of this article are as follows.

- 1) It identifies security requirements in industrial transportation and highlights existing ITS technologies.
- 2) 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.
- 3) 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 article 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. Finally, Section V concludes this article.

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]–[14]. VANETs play a critical role in modern ITS, as it exhibits several unique features due to its high mobility nature. Ali *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 Payalan and Guvensan 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] and [16]. Recent work in quality of service (QoS) related to in-time delivery and data dissemination pertinent to vehicular clouds and fog/edge-assisted technologies have been elaborated in [17] and [18]. The

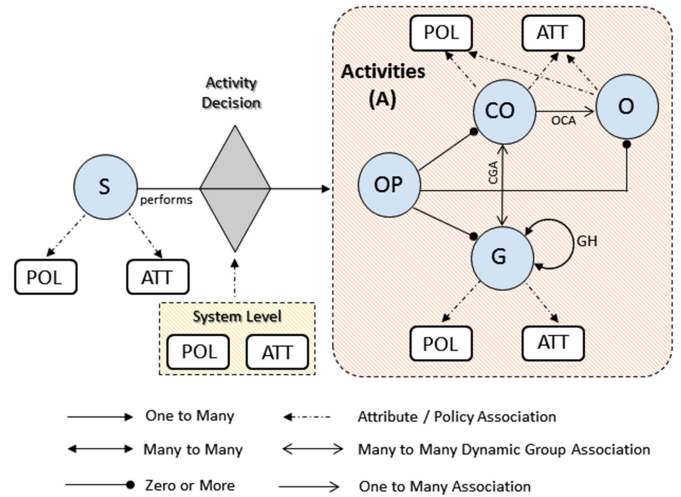


Fig. 1. Conceptual ITS-ABAC_G model.

edge of a network, on the other hand, is also a decisive factor in both V2I [5], [19] 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 toward evaluating connected vehicles in cloud was introduced in [25] to meet the requirement of modern ITS.

The role of big data (BD) analytics [26]–[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 multitier 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 a 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 article is an effort in this direction and will foster more similar research.

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. Fig. 1 represents 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 (COs), objects in clustered objects (O), groups (G), operations (OP), activities (A), authorization policies (POL), and attributes (ATT).

¹[Online]. Available: <https://aws.amazon.com/>

TABLE I
FORMAL ITS-ABAC_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: $ATT = \{\text{set}, \text{atomic}\}$.
- Entities in S, O, G, and CO are mapped to attribute values for every attribute att in ATT. Mathematically,

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

- 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 : CO → G.
- Each object is mapped to a clustered object, defined by parentCO : O → CO.
- Group Hierarchy is a partial order relation \succeq_g on G, defined as $\text{GH} \subseteq G \times G$.
This is equivalent to a group mapped to set of parent groups, stated as parentG : G → 2^G.

Derived Effective Attributes of Clustered Objects, Groups, and Objects

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

- $\text{Geff}_{\text{att}} : G \rightarrow 2^{\text{Range}(\text{att})}$, defined as $\text{Geff}_{\text{att}}(g_i) = \text{att}(g_i) \cup \left(\bigcup_{g \in \{g_j | g_i \succeq_g g_j\}} \text{Geff}_{\text{att}}(g) \right)$.
- $\text{COeff}_{\text{att}} : CO \rightarrow 2^{\text{Range}(\text{att})}$, defined as $\text{COeff}_{\text{att}}(\text{co}) = \text{att}(\text{co}) \cup \text{Geff}_{\text{att}}(\text{directG}(\text{co}))$.
- $\text{Oeff}_{\text{att}} : O \rightarrow 2^{\text{Range}(\text{att})}$, defined as $\text{Oeff}_{\text{att}}(o) = \text{att}(o) \cup \text{COeff}_{\text{att}}(\text{parentCO}(o))$.

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

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

Authorization Functions (Policies)

- Authorization Function: For each $\text{op} \in \text{OP}$, $\text{Auth}_{\text{op}}(s : S, \text{ob} : CO \cup O \cup G)$ is a propositional logic formula returning true or false, which is defined using the following policy language:

- $\alpha ::= \alpha \wedge \alpha \mid \alpha \vee \alpha \mid (\alpha) \mid \neg \alpha \mid \exists x \in \text{set}. \alpha \mid \forall x \in \text{set}. \alpha \mid \text{set} \Delta \text{set} \mid \text{atomic} \in \text{set} \mid \text{atomic} \notin \text{set}$
- $\Delta ::= \subset \mid \subseteq \mid \not\subseteq \mid \cap \mid \cup$
- $\text{set} ::= \text{eff}_{\text{att}}(i) \mid \text{att}(i) \quad \text{for att} \in \text{ATT}, i \in S \cup CO \cup O \cup G \cup \{\text{system-wide}\}, \text{attType}(\text{att}) = \text{set}$
- $\text{atomic} ::= \text{eff}_{\text{att}}(i) \mid \text{att}(i) \mid \text{value} \quad \text{for att} \in \text{ATT}, i \in S \cup CO \cup O \cup G \cup \{\text{system-wide}\}, \text{attType}(\text{att}) = \text{atomic}$

Authorization Decision

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

$$\text{Authorization}(a : A, s : S) \Rightarrow \text{Auth}_{\text{op}_1}(s : S, x_1), \text{Auth}_{\text{op}_2}(s : S, x_2), \text{Auth}_{\text{op}_3}(s : S, x_3), \dots, \text{Auth}_{\text{op}_n}(s : S, x_n)$$

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, CO such as a vehicle, which are part of the system. COs 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 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_G model supports security policies (POL) and attributes (ATT) for different entities like source, COs, objects, and groups, to ensure fine-grained access control solution. Such policies include personal privacy preferences together with system wide rules (as shown in Fig. 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, COs, 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, COs, 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 a 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 CO 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 CO 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 GH can be defined using the following proposition.

PROPOSITION: a partial order relation on G defined by \succeq_g , where $g_1 \succeq_g g_2$ illustrates g_1 is a subgroup 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 GH in Fig. 1.

While introducing groups offers many advantages, among them is 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 subgroups, therefore, when an attribute is set valued

Algorithm 1: Fine-grained Grouping Process of Industrial Connected Vehicle Authorization in Cloud Assisted System (FGAG).

```

Let  $\mu$  be the union of the entities
Let  $\phi =$  be a special set of  $\langle G, CO, \text{and } O \rangle$ 
Let  $att =$  atomic if  $att \cup \{\perp\} = \text{true}$  else  $att = \text{set}$ .
Input: entities and attributes associated with the system
Output: assign fine-grained access police for each group
for  $\forall [\mu \rightarrow \mu:attType(att) = \text{atomic}] \in \text{attribute}$  do
  1- atomic grouping:
  for  $G_{eff\_att} : G \rightarrow \text{Range}(att) \cup \{\perp\}$ , defined as
   $G_{eff\_att}(g_i)$  do
    if  $\forall g' \in \text{parent}G(g_i)$ .  $G_{eff\_att}(g') = \perp$  then
       $att(g_i) \rightarrow \text{parent}G(g_i)$ .  $G_{eff\_att}(g') = \perp$ 
    else
       $G_{eff\_att}(g') \rightarrow G_{eff\_att}(g') \neq \perp$  updated most
    end
  end
end
for  $\forall$  attribute  $\mu$  in  $ATT$  such that  $attType(att) = \text{set}$  do
  2- set grouping:
   $\mu : \phi \rightarrow 2^{\text{Range}(att)}$ , 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
   $\alpha ::= \alpha \wedge \alpha \mid \alpha \vee \alpha \mid (\alpha) \mid \neg \alpha \mid \exists x \in \text{set}.\alpha \mid \forall x \in \text{set}.\alpha \mid$ 
  if  $\text{set} \Delta \text{set} \parallel$  atomic  $\notin$  set then
    for  $\mu \in ATT$ , where  $attType(att) = \text{set}$ 
     $\text{set} ::= \text{eff}_{att}(i) \mid att(i)$ 
  else
    for  $\mu \in ATT$ , where  $attType(att) = \text{atomic}$ 
     $\text{atomic} ::= \text{eff}_{att}(i) \mid att(i) \mid \text{value}$ 
  end
end

```

its effective attribute value for att for a group g_i (denoted by $G_{eff_att}(g_i)$) 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 $G_{eff_att}(g_j) = att(g_j)$, for a maximal group g_j . CO_{eff_att} defines the effective attribute value of a CO 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 O_{eff_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 nonnull 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, $Auth_{op}(s : S, ob : CO \cup O \cup G)$ 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

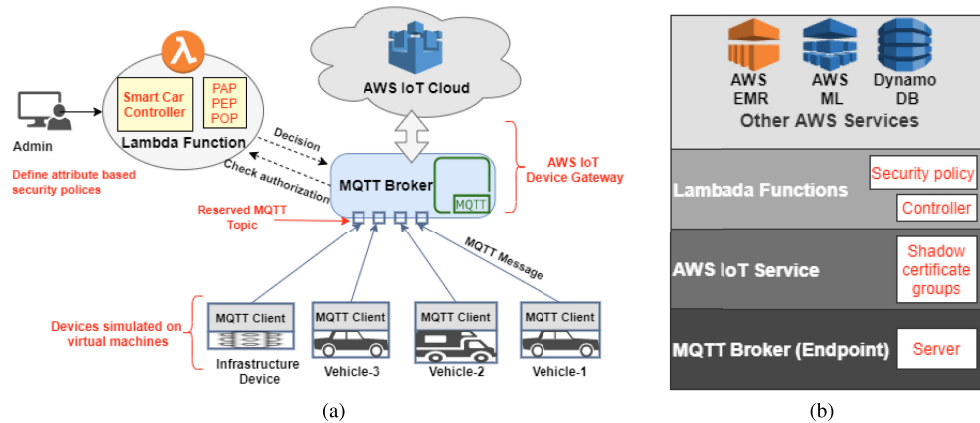


Fig. 2. Implementation architecture. (a) Overall architecture. (b) AWS services.

set by users for an individual CO, 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 compiled and checked to allow or deny an activity. Authorization function $\text{Authorization}(a : A, s : S)$ defines all the policies which must be 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 an 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 value group. The authorization functions of policies take place in the third phase, which will return to 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 variables that store the value of entities. In the first stage, if μ can fulfill all target constraints, the group elements are evaluated and assign atomic values. Otherwise, the process moves to the next policy and update the most recent policy. Following attributes are checked for the set values. At the operation stage, Algorithm 1 assigns the policy function by comparing the value of atomic and set attributes of 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 multilayer 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 an AWS IoT service.² The prototype implementation highlights how multilayer security policies and location groups assignment can be realized in AWS. We simulated real smart cars and infrastructures using IoT things. In the implementation, no long-term vehicular data were 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 Fig. 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 that only authorized devices can communicate. AWS Lambda³ function has been used to enforce ABAC policies [33] defined with the proposed model. Fig. 2(b) shows details of AWS cloud components, reflecting where device shadows,⁴ certificates,⁵ and groups⁶ are created in AWS IoT with MQTT broker acts like a server, offering a client-server architecture for the proof of concept.

²[Online]. Available: <https://aws.amazon.com/iot/>

³[Online]. Available: <https://aws.amazon.com/lambda/>

⁴[Online]. Available: <https://docs.aws.amazon.com/iot/latest/developer-guide/iot-device-shadows.html>

⁵[Online]. Available: <https://docs.aws.amazon.com/iot/latest/developer-guide/iot-security-identity.html>

⁶[Online]. Available: <https://docs.aws.amazon.com/iot/latest/developer-guide/thing-groups.html>

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.

1) *Deer Threat Alerts*: Sensors in the 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.

2) *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 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 multilayer 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.

1) *Administrative Phase*: A GH 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 nonoverlapping 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

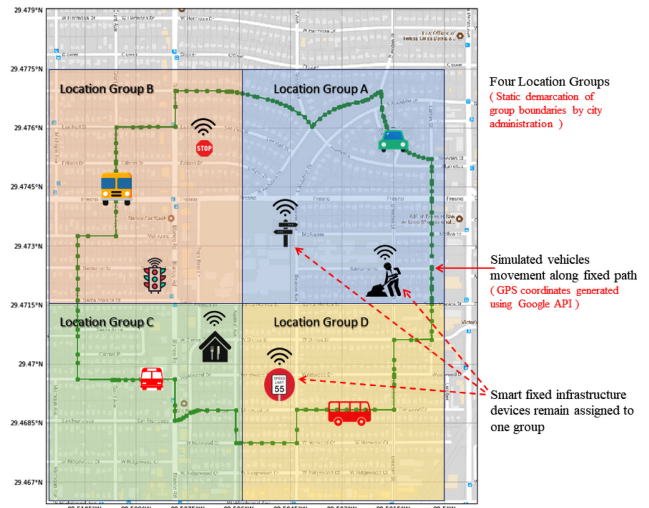


Fig. 3. Simulation of vehicles route and location groups.

```

('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.

coordinates, which are generated with Google (API⁷), iterating over green dotted line shown in Fig. 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 Fig. 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, is used in on-the-fly assignment of the vehicle to the relevant group. We implemented this functionality via stand-alone service using

⁷[Online]. Available: <https://cloud.google.com/maps-platform/>

```

{
  "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.

Lambda function and Boto⁸ AWS SDK for Python. Also, in a 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. Fig. 5 shows the snippet of policies implemented 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. If 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 Fig. 4, both Vehicle-1 and Vehicle-2 are members of Car-A subgroup; 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'
}

```

2) *Operational Phase*: In this phase, policies enforce restriction of alerts, notification, and services with various response time to regulation signals in the IloV environment [34]. This evaluation can involve single or multilevel 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 Fig. 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” 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.

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. A total of 50 moving vehicles have been stimulated for emulating the ITS environment. These vehicles

⁸[Online]. Available: <https://aws.amazon.com/sdk-python/>

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

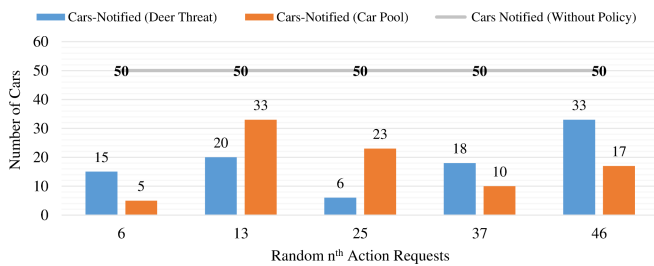


Fig. 6. Comparing the scoping and relevance of alerts with and without policy.

are made to spread arbitrarily, with the help of a smart-vehicle controller, over the four location subgroups, which are predefined and already exhibited in Fig. 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 Fig. 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 20 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.

Furthermore, 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 offers 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 Fig. 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 Fig. 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 the 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 Fig. 6. Although the n^{th} request that represents the scenario of deer-threat is entirely different from the n^{th} 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 toward the target vehicles.

Performance graphs, as shown in Fig. 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 the graph describes the total execution requests, meaning how many times deer-threat [see Fig. 7(a)] or pool [see Fig. 7(b)] 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 Fig. 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 Fig. 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 a blue line) as opposed to without policy implementation. Although, it is expected 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 tradeoff. The article 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

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

¹⁰[Online]. Available: <https://cloud.google.com/iot-core/>

¹¹[Online]. Available: <https://www.openstack.org/>

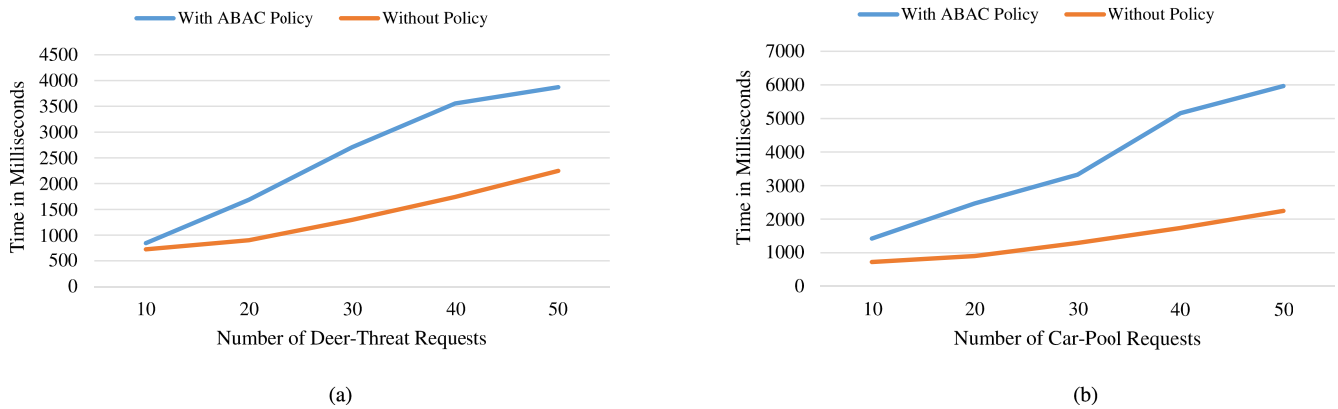


Fig. 7. Comparison of the performance with and without ABAC policy. (a) Deer threat use case. (b) Car pool use case.

V. CONCLUSION

This article developed an ABAC 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, introduced the element of groups, which were dynamically assigned to moving vehicles based on their attributes. This policy-based solution considered 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.

REFERENCES

- [1] Z. Zhou, C. Gao, C. Xu, Y. Zhang, S. Mumtaz, and J. Rodriguez, "Social big-data-based content dissemination in Internet of Vehicles," *IEEE Trans. Ind. Informat.*, vol. 14, no. 2, pp. 768–777, Feb. 2018.
- [2] M. Aazam *et al.*, "Cloud of things: Integrating Internet of Things and cloud computing and the issues involved," in *Proc. 11th Int. Bhurban Conf. Appl. Sci. Technol.*, Jan. 2014, pp. 414–419.
- [3] R. Lea and M. Blackstock, "City hub: A cloud-based IoT platform for smart cities," in *Proc. 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. ACM 23rd ACM Symp. Access Control Models Technologies*, 2018, pp. 193–204.
- [5] M. Gupta, J. Benson, F. Patwa, and R. Sandhu, "Secure V2V and V2I communication in intelligent transportation using cloudlets," 2020, *arXiv:2001.04041*, 2020.
- [6] R. S. Sandhu and P. Samarati, "Access control: Principle and practice," *IEEE Commun. Mag.*, vol. 32, no. 9, pp. 40–48, Sep. 1994.
- [7] D. F. Ferraiolo *et al.*, "Proposed NIST standard for role-based access control," *ACM Trans. Inf. Syst. Secur.*, vol. 4, no. 3, pp. 224–274, 2001.
- [8] V. C. Hu, D. R. Kuhn, and D. F. Ferraiolo, "Attribute-based access control," *IEEE Comput.*, vol. 48, no. 2, pp. 85–88, Feb. 2015.
- [9] X. Jin, R. Krishnan, and R. Sandhu, "A unified attribute-based access control model covering DAC, MAC and RBAC," in *Proc. DBSec*, 2012, pp. 41–55.
- [10] S. S. L. Chukkapalli, A. Piplai, S. Mittal, M. Gupta, and A. Joshi, "A smart-farming ontology for attribute based access control," in *Proc. 6th IEEE Int. Conf. Big Data Secur. Cloud*, 2020, pp. 29–34.
- [11] Z. Lu, G. Qu, and Z. Liu, "A survey on recent advances in vehicular network security, trust, and privacy," *IEEE Trans. Intell. Transport. Syst.*, vol. 20, no. 2, pp. 760–776, Feb. 2019.
- [12] I. Ali, A. Hassan, and F. Li, "Authentication and privacy schemes for vehicular ad hoc networks (VANETs): A survey," *Veh. Commun.*, vol. 16, pp. 45–61, 2019.
- [13] M. N. Mejri, J. Ben-Othman, and M. Hamdi, "Survey on Vanet security challenges and possible cryptographic solutions," *Veh. Commun.*, 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 Trans. Intell. Transport. Syst.*, vol. 21, no. 1, pp. 30–47, Jan. 2020.
- [15] U. Ahmad, H. Song, A. Bilal, M. Alazab, and A. Jolfaei, "Securing smart vehicles from relay attacks using machine learning," *The J. Supercomput.*, vol. 76, pp. 2665–2682, 2020.
- [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. 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 Trans. Cloud Comput.*, vol. 7, no. 1, pp. 196–209, Jan.–Mar. 2019.
- [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," *Veh. Commun.*, 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 Trans. Intell. Transport. Syst.*, to be published.
- [20] S. Darbha, S. Konduri, and P. R. Pagilla, "Benefits of V2V communication for autonomous and connected vehicles," *IEEE Trans. Intell. Transport. Syst.*, vol. 20, no. 5, pp. 1954–1963, May 2019.
- [21] A. Kayes *et al.*, "Achieving security scalability and flexibility using fog-based context-aware access control," *Future Gener. Comput. Syst.*, vol. 107, pp. 307–323, 2020.
- [22] W. Liu and Y. Shoji, "DeepVM: RNN-based vehicle mobility prediction to support intelligent vehicle applications," *IEEE Trans. Ind. Informat.*, vol. 16, no. 6, pp. 39970–4006, Jun. 2020.
- [23] M. Gupta, J. Benson, F. Patwa, and R. Sandhu, "Dynamic groups and attribute-based access control for next-generation smart cars," in *Proc. 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 Trans. Ind. Informat.*, vol. 15, no. 12, pp. 6500–6509, Dec. 2019.
- [25] M. Aladwan *et al.*, "TrustE-VC: Trustworthy evaluation framework for industrial connected vehicles in the cloud," *IEEE Trans. Ind. Informat.*, vol. 16, no. 9, pp. 6203–6213, Sep. 2020.
- [26] L. Zhu, F. R. Yu, Y. Wang, B. Ning, and T. Tang, "Big data analytics in intelligent transportation systems: A survey," *IEEE Trans. Intell. Transport. Syst.*, vol. 20, no. 1, pp. 383–398, Jan. 2019.

- [27] F. Awaysheh, J. C. Cabaleiro, T. F. Pena, and M. Alazab, "Big data security frameworks meet the intelligent transportation systems trust challenges," in *Proc. IEEE TrustCom/BigDataSE*, 2019, pp. 807–813.
- [28] F. Awaysheh *et al.*, "Next-generation big data federation access control: A reference model," *Future Gener. Comput. Syst.*, vol. 108, pp. 726–741, 2020.
- [29] M. Aladwan, F. Awaysheh, J. Cabaleiro, T. Pena, H. Alabool, and M. Alazab, "Common security criteria for vehicular clouds and Internet of Vehicles evaluation and selection," in *Proc. 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)", 2017. [Online]. Available: https://ec.europa.eu/transport/sites/transport/files/c-its_security_policy_release_1.pdf, 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, Gaithersburg, MD, USA, Tech. Rep. 800-162, 2014.
- [34] A. Bilh, K. Naik, and R. El-Shatshat, "Evaluating electric vehicles' response time to regulation signals in smart grids," *IEEE Trans. Ind. Informat.*, vol. 14, no. 3, pp. 1210–1219, Mar. 2018.



Maanak Gupta received the B.Tech. degree in computer science and engineering from Kurukshetra University, Kurukshetra, India, the M.S. degree in information systems from Northeastern University, Boston, MA, USA, in 2012, and the M.S. and Ph.D. degrees in computer science from the University of Texas at San Antonio (UTSA), San Antonio, TX, USA, 2017 and 2018, respectively.

He is currently an Assistant Professor in computer science with Tennessee Technological University, Cookeville, TN, USA. He has also been a Postdoctoral Fellow with the Institute for Cyber Security (ICS), UTSA. His primary area of research includes security and privacy in cyberspace focused in studying foundational aspects of access control and their application in technologies including cyberphysical 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.



Feras M. Awaysheh received the M.Sc. degree (Hons.) majoring in information, computer, and network security from the New York Institute of Technology (NYIT), Old Westbury, NY, USA, in 2010 and the Ph.D. degree in big data and cloud computing from the University of Santiago de Compostela, Santiago, Spain, in 2019.

He is currently a Researcher with the CiTIUS research center, Santiago de Compostela, Spain, and a Visiting Fellow with the University of Edinburgh, Edinburgh, U.K. His main research interests include 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 the B.Sc. and M.Sc. degrees in physics from Clarkson University, Potsdam, NY, USA, in 2007 and 2009, respectively, and the M.Sc. degree in electrical engineering from the University of Texas at San Antonio (UTSA), San Antonio, TX, USA, in 2016.

He has been with the Texas Renewable Energy Institute (TSERI) and Open Cloud Institute (OCI), UTSA assisting with data analytics and various research projects. He is currently a Technology Research Analyst II with the Institute for Cyber Security (ICS) and the Center for Security and Privacy Enhanced Cloud Computing (C-SPECC), UTSA. His current research interests include cyberphysical systems, cloud computing, and automation.



Mamoun Alazab (Senior Member, IEEE) received the Ph.D. degree in computer science from the School of Science, Information Technology and Engineering Federation University of Australia, Mount Helen, VIC, Australia, in 2012.

He is currently an Associate Professor with the College of Engineering, IT and Environment, Charles Darwin University, Casuarina, NT, Australia. He is a Cybersecurity Researcher and practitioner with industry and academic experience. He has more than 150 research papers in many international journals and conferences. He is the founding Chair of the IEEE Northern Territory (NT) Subsection. His research interest is multidisciplinary that focuses on cybersecurity and digital forensics of computer systems with a focus on cybercrime detection and prevention.



Farhan Patwa received the B.Sc. and M.Sc. degrees in electrical engineering from the University of Texas at Arlington, Arlington, TX, USA, in 1997 and 1998, respectively.

He is a Systems Engineer with more than 20 years of professional experience working in the telecom industry, cloud computing, and software security solutions. He has been with 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 with the Institute for Cyber Security, University of Texas at San Antonio, San Antonio, TX, USA.



Ravi Sandhu (Fellow, IEEE) received the Ph.D. degree in computer science from Rutgers University, New Brunswick, NJ, USA, in 1983.

He is currently the founding Executive Director and Chief Scientist with the Institute for Cyber Security, University of Texas at San Antonio, San Antonio, TX, USA, where he holds the Lutchter Brown Endowed Chair in Cybersecurity. His research interests focuses on security models and architectures, including the seminal role-based access control model. His papers have accumulated more than 40 000 Google Scholar citations, including more than 9000 citations for the RBAC96 paper.

Mr. Sandhu 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 CO-DASPY, and has been a leader in numerous other security conferences. He is a Fellow of the ACM and AAAS and an inventor on 30 patents.