Spring 4-2024

Automated Dynamic Policy Generation for Access Control in the Internet of Things

Kaushik Nagarajan Muthusamy Ragothaman

AUTOMATED DYNAMIC POLICY GENERATION FOR ACCESS CONTROL IN THE INTERNET OF THINGS

A dissertation submitted to Dakota State University in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

in

Information Systems

April 2024

By

Kaushik Nagarajan Muthusamy Ragothaman

Dissertation Committee:

Dr. Yong Wang

Dr. David Zeng

Dr. Jun Liu

Dr. Bhaskar Rimal

DAKOTA STATE UNIVERSITY

DISSERTATION APPROVAL FORM

This dissertation is approved as a credible and independent investigation by a candidate for the Doctor of Philosophy degree and is acceptable for meeting the dissertation requirements for this degree. Acceptance of this dissertation does not imply that the conclusions reached by the candidate are necessarily the conclusions of the major department or university.

Student Name: Kaushik Muthusamy Ragothaman Student ID: _____

Dissertation Title:
Automated Dynamic Policy Generation for Access Control in the Internet of Things

---

Graduate Office Verification: DocuSigned by: Abby Chowning F44C8D9E621C417... Date: 05/14/2024

Dissertation Chair/Co-Chair: DocuSigned by: Yong Wang 70AB505BC7B649E... Date: 05/14/2024
Print Name: Yong Wang

Dissertation Chair/Co-Chair: _____ Date: _____
Print Name: _____

Committee Member: DocuSigned by: David Zeng FB53FDB2AA1F41C... Date: 05/14/2024
Print Name: David Zeng

Committee Member: DocuSigned by: Jun Liu E0CE171C3CDE419... Date: 05/14/2024
Print Name: Jun Liu

Committee Member: DocuSigned by: Bhaskar Rimal 911AA4B61574EF... Date: 05/15/2024
Print Name: Bhaskar Rimal

Committee Member: _____ Date: _____
Print Name: _____

Submit Form Through Docusign Only
or to Office of Graduate Studies
Dakota State University

ACKNOWLEDGMENTS

I would like to express my thanks to Dakota State University for giving me the opportunity to study at this esteemed institution.

My special sincere gratitude to Dr. Yong Wang for his valued guidance, support, and encouragement throughout the dissertation. I am grateful for the opportunity to work with him since the beginning of my PhD program as his graduate assistant. I would like to thank my committee members, Dr. David Zeng, Dr. Jun Liu, and Dr. Bhaskar Rimal, for all their questions, suggestions, and feedback on improving the dissertation. I express my thanks to the Office of Graduate Studies, Dr. Mark Hawkes, and Dr. Cherie Noteboom, for their support during the PhD program. I acknowledge all faculty and my student colleagues with whom I had the opportunity to work on various research projects.

I want to thank my grandparents, parents, brother, uncles and aunts, cousins, and friends for their wholehearted support and confidence in me.

Above all, I thank the Almighty for his blessings.

ABSTRACT

Internet of Things (IoT) is commonly utilized in domestic and industrial environments to automate various tasks. Due to this, an enormous amount of data is being generated and transmitted through IoT networks. These data may contain sensitive information depending on the context. Access control is one of the frontline security measures that any information system should adopt. The dynamic nature of the IoT requires access control policies should be able to adapt to their environments. However, it is very challenging to specify access control policies manually because of their dynamic nature. Current literature suggests the need for automating the process of policy generation. Machine Learning and Deep Learning techniques can enable the required automation. The main objective of this dissertation is to answer the following research questions: 1) How can we self-generate contextual access control policies for the Internet of Things during unforeseen situations? 2) What are the existing challenges while specifying dynamic policies for access control in IoT? 3) How realistic are the generated access control policies to be used in real-time situations? In this research, we proposed a mixed-method approach where we implemented and evaluated two baseline Tabular Generative Adversarial Network models. We evaluated the performance of the solution using two datasets, namely the CAV Policies and Amazon Access Logs datasets. We obtained different perspectives based on our experiments. The common findings that our results demonstrate are that the models were able to generate synthetic access control policies by training from the datasets, and the models were able to learn the background knowledge specified during training to generate policies without any constraint violation.

DECLARATION

I hereby certify that this dissertation constitutes my own product, that where the language of others is set forth, quotation marks so indicate, and that appropriate credit is given where I have used the language, ideas, expressions or writings of another. I declare that the dissertation describes original work that has not previously been presented for the award of any other degree of any institution.

Signed,

Kaushik Nagarajan Muthusamy Ragothaman

Kaushik Nagarajan Muthusamy Ragothaman

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

1.1 Internet of Things and IAM

Internet of Things (IoT) provides many conveniences to users in domestic and industrial environments. Statista estimated that 75 billion devices would be connected to the Internet worldwide by 2025 [1]. Security and privacy are two major concerns revolving around the IoT. IoT device manufacturers and service providers are required by regulations to ensure the security of the devices, thereby protecting the users' privacy.

Identity and Access Management (IAM) is one of the most important services in security and a prime module in implementing security for any IoT application [2], [3]. Imagine a car manufacturer that sells smart cars to its customers. The manufacturer must design the vehicle's system in a way that it constantly collects and processes data from its surrounding environment through sensors embedded in the vehicle. The system might occasionally transmit the collected data to the manufacturer via the Internet. The data may contain the driver's personal information and sensitive information such as locations. The data that is being sent and shared with the manufacturer should be accessed by authorized users only. Furthermore, appropriate controls must be placed on the shared data if the manufacturer facilitates remote start functions for vehicle owners.

Access control provides the desired service to protect against unauthorized use of accessible resources. Traditional access control techniques are being adopted or extended for

access provisioning and management for the IoT. However, the design and implementation of access control for the IoT are complicated. IoT networks include devices with different hardware and software configurations. Their heterogeneous nature raises a considerable challenge for any access control solution. In addition, IoT devices are resource-constrained devices with limited memory, computation power, and battery [4]. The constrained resources on the IoT devices limit the use of complex algorithms when designing an access control solution. Further, IoT networks have encountered major attacks in recent years on a global scale [5]–[7]. Governments of many countries have already initiated to formulate policies for IoT devices. Hence, an appropriate access control solution is required for any IoT network.

1.2 Statement of the Problem

Access control policies provide the specifications at a higher level as to which subject has access to what resources in a given context. Traditional approaches such as Discretionary Access Control (DAC), Role-Based Access Control (RBAC) with static policies are not suitable for such dynamic and heterogenous environments. Attribute-Based Access Control (ABAC) is considered as one of the most suitable models due to its expressiveness by supporting the use of multiple attributes. However, ABAC is still not considered as a completely dynamic model due to static policies.

Access control policies should be able to adapt to its environment for IoT systems. Calo et. al. [8] suggests the need for dynamic access control in which the system should be able to generate policies by itself according to the varying contexts. Machine Learning (ML) and Deep Learning (DL) models are being utilized in order to realize the required automation at policy level.

Consider a smart city platform where various types of IoT devices are connected to a network and operated at scale. In a case of a smart street light system, it is expected

that multiple players, such as, device manufacturers, municipality, smart city solution providers, all interoperate with the devices at varying situations. Situations shall be the access to maintenance data by the manufacturer, access to statistical data by the municipality, collection, and storage of real time sensor data by solution provider, access to device information by the devices gateway etc. For an autonomous vehicle, varying situations such as, driver requesting autonomous mode for a particular time period, driving system requesting access to location information, driving system is unable to access the policy repository etc. In addition to those discussed above, there may be use cases where a policy may not exist for a given situation. In such dynamic situations, it will be expected that the system generates its own policies. Automated policy generation could be considered to overcome the challenges. In automated policy generation, the high-level policies required to administer a system are generated by a set of processes without the need for manual input. In IoT systems, automated policy generation has benefits in terms of scalability and flexibility by eliminating the need for a human administrator to specify the policies. Calo and Cunnington introduced various scenarios where dynamic generation of policies are required [8], [9]. They used Answer Set Grammar and inductive logic programming to generate policies.

To sum up, the traditional access control models mentioned above utilize static policies and are not suitable for IoT environments. The challenge in creating and enforcing adaptive and contextual access control policies for IoT, particularly in unforeseen situations still exists, and the scope to propose novel solutions to self-generate the policies is wide open.

On the other hand, we posit that the generated policies should adhere to the system and environmental requirements. IoT devices and systems are used to automate various physical processes. Hence, the access control policies should not only conform to the technical security requirements, but also consider the physical safety of the environment. Let us consider a situation where a policy states to turn water sprinkler on when a smoke

is detected. At the same time, if there is another policy to turn the water valve off when moisture is detected, then these two rules are not only contradicting, but they will also cause violation to physical safety. There are many IoT platforms nowadays such as IFTTT, OpenHAB, Zapier takes policies as input in the form of natural languages and then convert them into their respective symbolic representations. Also, [9] states a future scenario where a driver in an autonomous vehicle can communicate with a driving system through a natural language interface. With this, there are many tools proposed in the academic literature that analyze these natural language policies. Recently, [10] performed a survey of tools that performed security and safety verification of IoT systems. In another work, [11] discusses the challenges related to the quality of access control policies. However, the models that generate access control policies should be able to learn the requirements and generate policies that conform to these requirements.

1.3 Objective of this Dissertation

To overcome the challenges presented in the previous section, the main research questions this dissertation will address is,

- **RQ1: What are the existing challenges while specifying dynamic policies for access control in IoT?**
- **RQ2: How can we self-generate contextual access control policies for the internet of things during unforeseen situations?**
- **RQ3: How realistic are the generated access control policies to be used in real-time situations?**

To this end, we propose that Tabular Generative Adversarial Networks shall be used for the automated generation of access control policies. We will utilize two datasets

that are publicly available to experiment and evaluate our approach. Hence, the specific objectives of this dissertation are as follows:

- Answer RQ1 by conducting an extensive survey of the literature to identify the challenges while specifying dynamic policies for access control in IoT.
- Explore the potential of Tabular Generative Adversarial Networks (GANs) to generate access control policies.
- Answer RQ2 by implementing the baseline Tabular GANs and compare them with other existing solutions in the literature.
- Answer RQ3 by evaluating the generated access control policy data against the requirements specified during the model learning process.

1.4 Outline of this Dissertation

This dissertation is organized as follows. Chapter 1 provides a general introduction about Internet of Things and Identity & Access Management, statement of the problem, and objective of this dissertation. In Chapter 2, we performed a comprehensive survey of existing literature on access control in IoT. Chapter 3 discusses the methodology we adopt for this research, the datasets we use, and our experimentation strategy. Chapter 4 describes our results and analysis. Chapter 5 provides a discussion on the significance of our approach and the limitations of the research. Chapter 6 concludes the dissertation and discusses the scope for future work.

Chapter 2

Literature Review

This chapter conducts a comprehensive survey of access control in the IoT, including access control requirements, authorization architecture, access control models, access control policies, access control research challenges, and future directions. A few survey articles discuss IoT access control [12]–[14]. However, none of them discuss the issues regarding IoT access control policies. The key contributions of the chapter are summarized as follows.

- The latest development of the access control in the IoT is provided to understand the recent progress on the access control.
- Access control requirements are discussed to help design and implement access control solutions for effective IoT IAM.
- Three major access control authorization architectures, namely, policy-based eXtensible Access Control Markup Language (XACML), token-based Open Authorization (OAuth), and hybrid User-Managed Access (UMA) architectures are discussed and their essential components are briefly summarized.
- We have compared different IoT access control models, including discretionary access control, role-based access control, organization-based access control, usage-based access control, capability-based access control, attribute-based access control,

blockchain-based access control, and relationship-based access control to facilitate the adoption of access control solutions.

- Access control policies such as dynamic policies specification are thoroughly discussed and challenges faced by the current solutions are highlighted.

- To guide future research in access control, we have summarized the research challenges in access control and also pointed out future research directions in the IoT.

The remainder of this chapter is organized as follows: Section 2.1 presents an overview of access control in IoT. Section 2.2 summarizes the requirements for access control in IoT. Section 2.3 focuses on authorization issues in access control, followed by discussions of access control models and policies in Section 2.4 and 2.5, respectively. Section 2.6 discusses the challenges in access control in IoT.

2.1 Access Control in IoT

IoT interconnects computing devices embedded in everyday objects. IoT has been widely adopted in consumer and business environments to bring convenience and facilitate business processes. Due to the large amount of data IoT collects and the sensitivity of the data, IoT security is critical.

2.1.1 Identity and Access Management

IAM is at the core of security in the IoT and coming up with an effective IAM solution designed and developed for the IoT is an important step [15], [16]. Unfortunately, there is no well-defined and established standard for the IAM in the IoT. Different manufacturers have developed their solutions to identify, provision, authenticate, and authorize their devices. As a result, IoT systems from different manufacturers are not able to communicate with each other and thus are not able to take advantage of capabilities already

available in other devices. Even if the manufacturers agree to use the same IAM system to manage their devices for better interoperability, there are challenges to identifying such an IAM system. The current IAM systems focus on managing users, not the devices. The solutions for user authentication are very mature. However, the solutions for device authentication are still in development. Further, the scale of the devices to be supported in the IoT is also far more than the number of users an IAM system supports in a typical enterprise network.

2.1.2 Communication Models

The Internet Architecture Board (IAB), which oversees the development and growth of the Internet, published RFC 7452 in March of 2015. RFC 7452 outlined four communication models for IoT devices: (1) device-to-device model; (2) device-to-gateway; (3) device-to-cloud; and (4) backend data sharing.

In a device-to-device communication model, two devices directly connect and communicate with each other rather than through an intermediary. Communications are often conducted through a wireless network using protocols like Bluetooth or ZigBee. The device-to-device communication model is often suitable for home automation systems such as thermostats, smartwatches, and light bulbs where the packet size is small, and the data throughput is relatively low.

In a device-to-gateway model, IoT devices access cloud services using intermediate gateways. Although gateway devices may come in many different forms, they serve the same purposes, i.e., aggregating received data, bridging the gaps between devices using different communication protocols, implementing security, and forwarding data to the cloud. In some cases, a gateway device may relay the data to another gateway which will then forward the data to the cloud.

In a device-to-cloud model, customers build and devise rules to filter data generated from IoT devices and take actions accordingly through cloud computing services. For

example, Microsoft Power BI is a cloud service that gives non-technical users the ability to analyze and visualize data generated by IoT devices. As the IoT devices continuously generate data and transmit the data to the cloud, network congestion may occur. The network traffic should be monitored constantly, and appropriate settings and actions should be taken to avoid lagging data throughput.

IoT devices often upload data to a particular cloud service provider, leading to data silos where data is isolated from other applications. A backend data-sharing model extends the device-to-cloud communication model. It prevents data silos by allowing the generated data to be shared among trusted parties. The backend data-sharing model also allows a user to export, aggregate, and analyze data generated from IoT devices from other applications.

2.1.3 Identities of Things

IAM systems focus on managing the identities of users. The IoT requires extending identity management to include devices. Identities of Things (IDoT), a general term describing the IoT entities (e.g., users and devices), has been adopted.

The IDoT includes the identities of both users and devices. Identities of users have been studied extensively. Three primary authentication factors could be used to identify users: something you know (e.g., username and password), something you have (e.g., a physical token or a smartcard), and something you are (e.g., fingerprint or face recognition). Identities of devices are still in development. New schemes for uniquely identifying devices need to be further studied. Due to the vast number of devices available, the scalability of any new scheme is essential.

2.1.4 Access Control

Access control is primarily implemented within centralized and distributed architecture categories in the IoT. In a centralized architecture, a single node is used for policy ad-

ministration and management, i.e., access provisioning and revocation happening from a single entity [14]. One of the limitations in centralized architecture is the single point of failure. In a dynamic environment like IoT, the entity that administers access control decisions is expected to be available anytime.

Distributed architecture, in contrast, can handle multiple nodes for administration [14]. Although it is easier to facilitate delegation and scalability, a challenge in designing access control solutions for distributed architecture is coherence. A decision or a change made at one node should reflect in all the other managing nodes. Designing an appropriate access control solution depends on the architecture of the IoT network.

An access control system in IoT includes essential functions such as authentication function, access control function, audit function, managing policies function, and administration function as shown in Figure 2.1. The authentication function verifies the identity of a user, a process, or a device. Access control function grants and denies specific requests from a user, a process, or a device to access resources. Access control policies describe high-level requirements that specify how access is managed and who may access information under what circumstances. An access control system also includes an administration function to create, provision, and effectively manage different users, groups, roles, devices, and policies. The audit function provides an independent review and examination of records and activities to assess the adequacy of access control to ensure compliance with established policies and operational procedures. Authorization takes place after authentication is complete to determine which resources/services are available to a user or a device.

In this paper, we focus on IoT access control. Since authentication and policies are also essential components in an access control system as shown in Figure 2.1, this paper also reviews available IoT authorization architectures and policies specification.

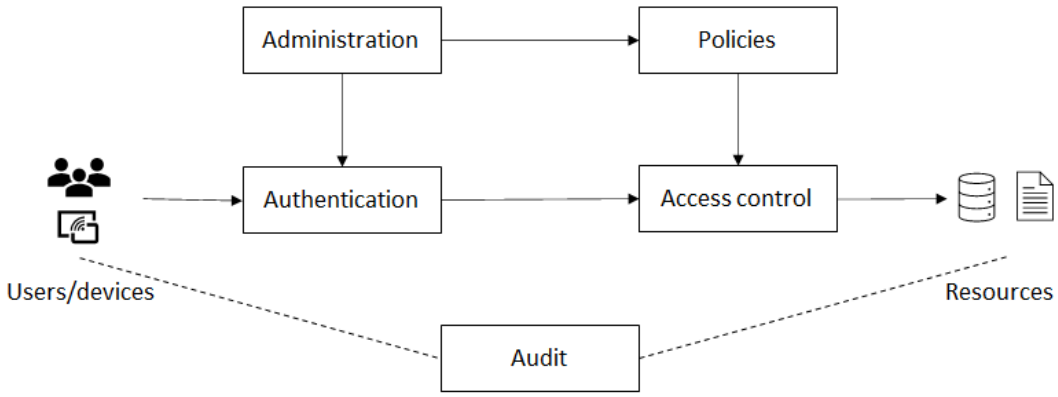

Figure 2.1: Access Control in IoT

2.2 Access Control Requirements

This section presents a summary of the concerns related to IoT access control.

1. **Granularity:** The fine-grained nature is the most important characteristic for any solution that is designed to manage access rights. Due to the heterogeneity property of the IoT networks and their dynamic nature, granularity is a major concern while designing the models [13], [17].

2. **Policy Specification:** The policies developed for the access control models should be able to handle dynamicity, allow and monitor delegation. An IoT network may contain a large number of devices presented in various forms and locations. Therefore, access control should consider the granularity and the policies specification to govern the network effectively [13].

3. **Handling Complexity:** IoT networks are heterogeneous networks which are characterized by resource-constrained devices, multiple hop links, unreliable communications, and limited physical security. Access control models shall be designed to handle the complex nature of the IoT networks [14].

4. **Interoperability:** Many device manufacturers provide a variety of IoT devices to customers. There is a high possibility that an IoT network may contain devices

from different manufacturers and must function together. Therefore, access control must support this interoperable nature in the IoT [18].

5. **Facilitation of Users:** IoT devices may be shared and accessed by multiple users. For example, virtual assistants and smart home products can be used by family members and guests at home. Access control must be able to allow users to delegate access to other users instead of handling all at a single administrative point [19], [20].

6. **Automation:** The complex nature of IoT environments and the number of access decisions to be made at a given time make it difficult to provision or make decisions individually. Hence, the processes of policy generation, decision, and evaluation should be automated in the IoT [17], [21].

7. **Resource Constraints:** IoT devices possess low memory and processing power when compared to regular computing machines. These constraints also raise challenges in developing access control solutions for the IoT [14], [22].

8. **Coherence:** In the case of multiple administrative points adopted in an access control model, all the administrative nodes should be coherent with management and provisioning of access control. The variant types of IoT networks create a challenge when ensuring coherence across multiple administrative domains in the IoT [13].

9. **Resolving Identities**: IoT devices can be characterized by attributes such as model number, serial number, IP address, physical address, location, etc. In turn, these devices are accessed by other devices and human users when connected to a network. Leveraging the combination of the device attributes to uniquely identify a device in a network poses a concern during the access control specification and implementation [17].

10. **Downtime:** The dynamic nature of the IoT environments tests the limits of any access control solutions. Since access decisions are made frequently, there should be no downtime [12], [13], [21]. The design of a centralized administrative point or a distributive model decides the downtime. In a centralized model, if the administrative node fails, it causes a single point of failure.

11. **Security:** The security of an access control model is another major concern. Access control solutions should be resistant to any cyber attack [14], [16], [22].

The concerns discussed above should all be considered while designing and implementing an access control solution for effective IoT IAM. Table 2.1 summarize these requirements.

Table 2.1: Access Control Requirements for IoT.

Requirement	**Description**
Granularity	Access control solution should be fine-grained.
Policy Specification	Access control policy should handle dynamicity and delegation.
Complexity	Access control can handle the complex nature of IoT networks.
Interoperability	Access control solution should support interoperability.
Facilitation of Users	Access control policy should allow users to configure and delegate controls.
Automation	Policy enforcement process shall be automated to support dynamicity and scalability.
Resource Constraints	Access control solution should consider the constrained nature of IoT devices.
Coherence	Access control solution should be coherent at administrative level.
Identity	Access control solution should possess attributes used for device identification.
Downtime	Access control solution should possess multiple administration points to avoid downtime.
Security	Access control solution should be secure and resistant to any cyber attack.

2.3 Access Control Authorization Architecture

There are different types of authorization architectures available. The common types are the policy-based eXtensible Access Control Markup Language architecture, the token-based Open Authorization architecture, and the hybrid User-Managed Access architecture [12]. Other customized architectures are either derived from those three or specific to the proposed applications.

2.3.1 Policy-based XACML Architecture

XACML is an access control language based on Extensible Markup Language (XML), which is standardized by the OASIS consortium [12]. XACML is a popular standard that provides fine-grained access control. XACML describes access control language, request/response language, and reference architecture. The architecture consists of components, namely the Policy Enforcement Point (PEP) to perform access control, Policy Decision Point (PDP) to offer authorization, Policy Information Point (PIP) as a source of attributes, Policy Administration Point (PAP) to create and administer the policy. XACML and attribute-based access control in combination can offer rich and fine-grained solutions. The interpretation of attributes and the language used to define the access control policies is complex making this standard a limitation in terms of usability [14].

The essential components present in an access control solution may include PAP, PEP, PDP, PIP, and Policy Refinement Point (PRP). Figure 2.2 demonstrates the policy-based XACML architecture. These essential components are briefly discussed below.

1. **Policy Administration Point:** The PAP, also known as a policy repository, is where all the policies required to grant or deny permissions are stored. Typically, the policies are stored in a specific format, for example, XACML. In addition, the PAP makes the complete access control policy available for the policy decision point to

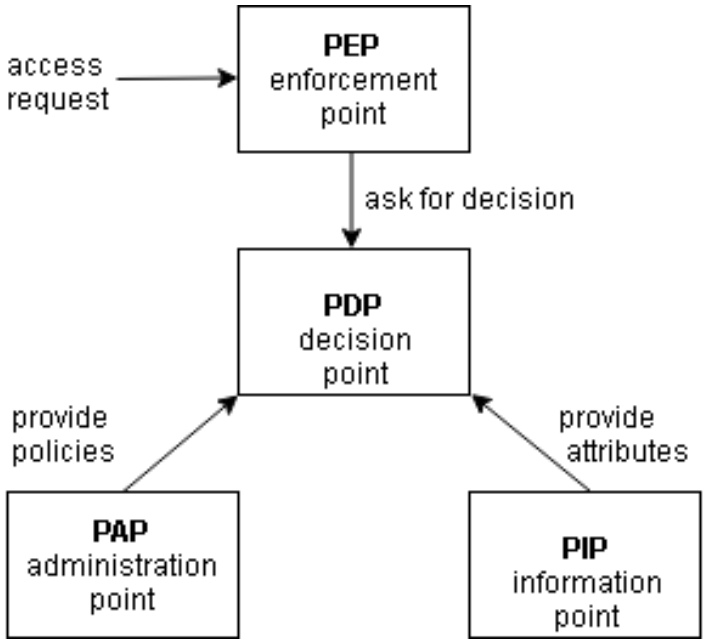

Figure 2.2: XACML Architecture [12].

grant or deny permissions [9], [23]. In an IoT environment, PAP should be designed so that policies can be added, removed, or modified at run time.

2. **Policy Enforcement Point:** The PEP acts as an intercept between the PDP and the requesting subjects. It forwards every request made by a subject along with the attribute values related to the subjects, the resources, the actions to be performed, and the environment to the PDP. Once the evaluation is performed at the PDP, the PEP retrieves the decision and forwards it to the subject that made the request. Moreover, based on the decision, the PEP is responsible for enforcing the actions that the subject can perform on a resource (e.g., read, write, or both) [23]–[25].

3. **Policy Decision Point:** The PDP evaluates the requests it receives based on the subject that makes the request, the resource that the subject is requesting to access, and the contextual (attributes) information. The PDP triggers the PIP to provide all the required contextual information, such as attribute values of the requester, the resources, the action that is being requested, and the environmental variables. Based on the information that is received, the PDP evaluates the decision

by verifying them against the policies [23]–[25].

4. **Policy Information Point:** The PIP is responsible for collecting and storing all the contextual information related to the system. In an IoT network, granting or denying permissions based on the context is one of the important requirements of access control. Hence, whenever the PDP requires the contextual information and the attribute information, the PIP sends them through the PEP to make an access decision [23]–[25].

5. **Policy Refinement Point:** The PRP is a component that is responsible for refining policies at run time and updating the policy repository. The refining process can be triggered for several reasons such as any change for the context in the environment or detection of an abnormal or unauthorized access behavior [9], [26]. Various techniques have been adopted in the literature for the policy refinement process. Most of these techniques are based on machine learning and deep learning. The PRP contributes to automating policies specification for access control which is essential in a dynamic environment like IoT.

2.3.2 Token-based OAuth Architecture

OAuth is an open-source authorization standard that is mainly used to provide access to web applications and services. With OAuth, users can access protected resources to third-party applications without disclosing their login credentials. Major OAuth service providers include Google, Microsoft, and Facebook [12]. These service providers are identity providers, who verify the users and provide external applications access to the users' information stored on the providers' domains with the users' consent. OAuth has several advantages in terms of scalability, interoperability, and flexibility. However, research finds that it lacks fine-grained property during implementation. Due to the requirement of the user registration, the client registration, and the nature of IoT networks, implementation

and configuration are challenging for service providers.

The Internet Engineering Task Force (IETF) has extended OAuth 2.0 for devices, and browserless clients under RFC8628 [27]. Figure 2.3 shows the OAuth device flow. As shown in the figure, the authorization flow is the sequence of steps (A) through (F). The client initially sends an access request along with its client identifier to the authorization server. Following the request, the authorization server responds with a device code, an end-user code, and end-user verification Uniform Resource Identifier (URI). Next, the client provides instruction to the end-user to use a user-agent on another device and visit the end-user verification URI. After the end-user is authenticated, the authorization server prompts the user to input the end-user code for validation. During this step, when the end-user reviews the client's request, the device client continually polls the authorization server to identify if the user has completed the authorization step. Finally, the authorization server validates the device code and issues the access token to the client if the access is granted, an error in case of denial, or notify the client to poll the authorization server continually.

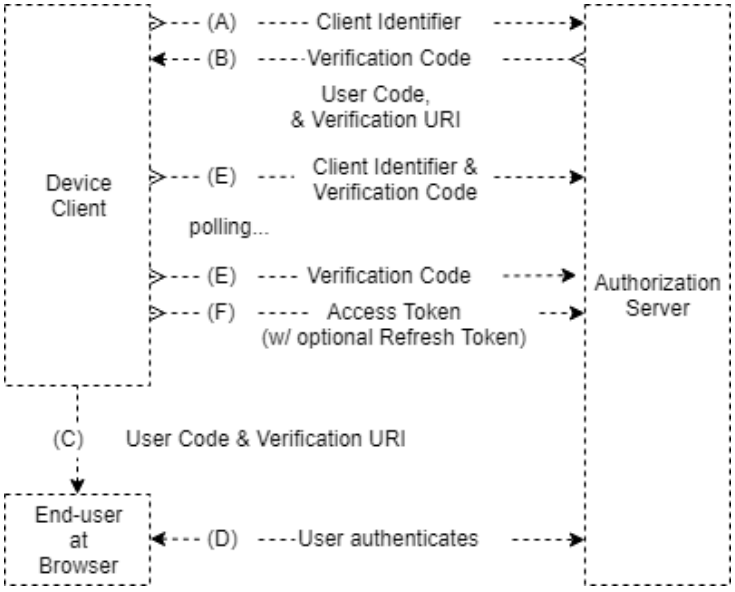

Figure 2.3: OAuth Device Flow [27].

Protocols such as Constrained Application Protocol (CoAP) and Message Queuing

Telemetry Transport (MQTT) are mainly used for resource-constrained devices, leveraging OAuth 2.0 tokens for authorization purposes.

- **Constrained Application Protocol:** CoAP is a protocol specially designed for interaction between endpoints and networks that are resource-constrained [28]. Specifically, this protocol is designed for machine-to-machine applications. The structure of CoAP is logically divided into two layers [29]. The first layer is used for requests and responses. CoAP uses a Representational State Transfer Constraints approach, allowing the clients to use HTTP methods to send requests [29]. The second layer, called the message layer, is used for retransmitting lost packets [29]. CoAP uses Datagram Transport Layer Security (DTLS) protocol for security.

- **Message Queuing Telemetry Transport:** MQTT is a messaging protocol for the IoT standardized by the OASIS consortium. MQTT offers bidirectional communication and supports scalability and reliability. MQTT is considered a great communication protocol for the IoT due to its simple, lightweight, and easy deployment properties [30]. Moreover, the use of MQTT has advantages on the ability to work with low-end devices [31], implementing machine learning algorithms in the cloud by interfacing the device with the Internet [32], and easy integration of new devices [32]. MQTT also comes with limitations. The default plain-text data exchange mechanism is a significant threat to data security [30]. Several security attacks on IoT communication protocols were analyzed in [29].

2.3.3 Hybrid User-Managed Access Architecture

User-Managed Access is developed as part of the Kantara Initiative [33]. UMA is an OAuth-based protocol. Unlike OAuth, access to third-party applications for resources is granted regardless of where those resources reside. Hence, UMA follows a capability-based approach, in which an entity with a defined capability and an access token will have access

to a resource [12], [14]. UMA is a user-oriented standard and is evolving to be adopted in IoT environments.

Figure 2.4 shows an example of UMA architecture. A resource owner manages all the resources stored in a resource server. The function of the authorization server is to protect the resource server. The resource server registers the resources that need to be protected with the authorization server and then configures them with appropriate policies for the registered resources. The client first sends a request to the resource server to receive an authorization grant. On the first attempt, the resource server registers the permission with the authorization server and issues a permission ticket to the client. The client presents the ticket to the authorization server. If the permission is granted, the authorization server issues a requesting party token (RPT) to the client. The client uses the RPT to access the requested resources.

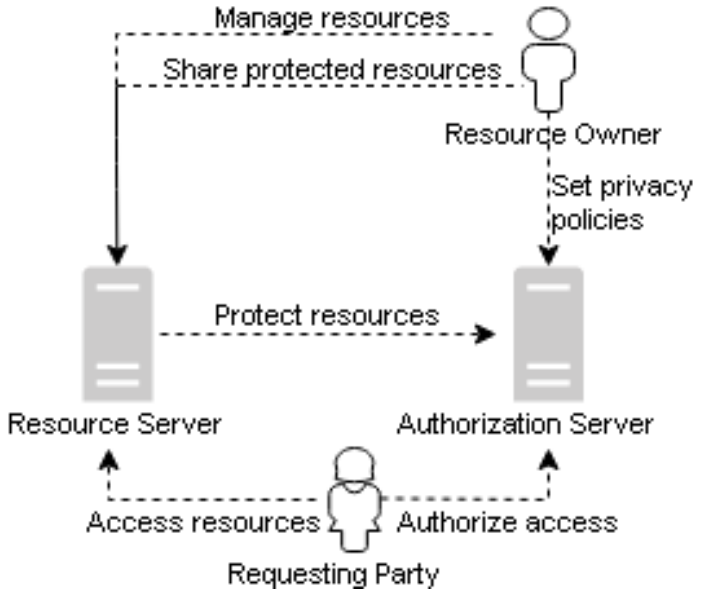

Figure 2.4: UMA Architecture [14].

UMA provides a unified control point for authorization [34]. There are many use cases where UMA can be utilized for IoT networks. However, research has identified four major challenges including availability, transparency, traceability, and maintainability [35]. UMA follows a centralized architecture and therefore is at risk of a single point of

failure [35]. In a third-party access control service utilizing UMA, it can be a difficult task to trace authorization history [35]. In terms of maintenance, it is challenging to upgrade in a centralized solution [35]. The blockchain technology that follows distributed architecture is being researched to assist UMA for the IoT [35], [36].

2.4 Access Control Models

There are many access control models for traditional computing and networking environments. An overview of such models implemented for the IoT and their issues are discussed below.

1. **Discretionary Access Control (DAC):** DAC is one of the primary access control techniques introduced in computing. It grants access by managing an access control matrix or an Access-Control List (ACL) [37]. Once access is granted in DAC, it remains forever until the administrator revokes access. In IoT, access should be continually monitored and evaluated for timely revocation. As new devices are being added or when existing devices are removed, access control must be updated automatically. Access decisions should be made based on the various criterion in different situations. DAC is a static model and the ACL must be manually updated by an administrator. For a dynamic environment like IoT, DAC is not suitable.

2. **Role-Based Access Control (RBAC):** In RBAC, a user is granted access based on roles which are in turn assigned with appropriate permissions to access resources [38]. Although it is easy to assign permissions to roles, many users may fall under a single role. RBAC is suitable for access rights in regular computing environments but not in the IoT. As IoT devices come with a variety of functionalities and offer a wide range of services, the administrator must create a new role whenever a device with new functionality is added to a network. In a large enterprise network, this may lead to role explosion. In addition, RBAC does not support dynamicity.

3. **Attribute-Based Access Control (ABAC):** ABAC is considered by many as one of the suitable models for IoT to provision access rights because of its ability to support additional attributes with user roles. Using ABAC, different attributes of IoT such as device ID and location can be included for evaluation while providing access. Even though this model is being used in large-scale projects like smart grid, ABAC faces the issue of complexity due to its centralized architecture [39], [40].

4. **Organization-Based Access Control (OrBAC):** OrBAC is an extension of the role-based access control by including a new dimension called "organization" [41]. This additional attribute helps in granting access when multiple organizations play a role or when an organization has many subdivisions. However, other than the above-mentioned concept, this model is no different from its parent model RBAC and is considered unsuitable for heterogeneous and dynamic IoT environments.

5. **Usage-Based Access Control (UCON):** UCON was introduced as a framework to protect digital resources that come under digital rights management (DRM). This model comes with three main concepts: authorization, obligation, and condition [42]. The authorization represents evaluation as to whether a subject is eligible to be provided access. The obligation is a criterion that a subject must perform to be provided with or sustain access. The condition represents the criteria that a subject must satisfy. Due to the three evaluation categories, UCON provides high dynamicity where the access is continually monitored, thereby revoking access whenever required by policies. However, this model does not explain the delegation property and follows a centralized architecture.

6. **Capability-Based Access Control (CapBAC):** The concept of CapBAC was started as part of the IoT@Work project [43]. It is an initiative by the European Union to leverage IoT to automate various services in public sector entities [43]. CapBAC follows a distributed approach. It is implemented through various nodes by

using PDP and PEP [44]. In CapBAC, a resource requester must show a particular capability to request an access token. The PDP decides whether to issue the token to the requester. Once issued, the token is evaluated at the PEP for the requester to access the resource. Another advantage of CapBAC is the property of delegation, where nodes can be given the authority to provide access to other nodes. The level of delegation is determined while designing the model. Nevertheless, the model must depend on a central server for either identity verification or certificate whether to trust the requester or not. Access is issued based on the requester's capability. CapBAC does not consider context while provisioning access [12].

7. **Blockchain-Based Access Control (BBAC):** Blockchain technology has had explosive growth in security and privacy applications in recent years. The important characteristic of this technology is its distributed nature. The methods through which access control using blockchain is described in the literature can be further divided into transaction-based and smart contract-based access control [45]–[47]. Transactions can be used to grant, delegate, or revoke access rights. Smart contracts can evaluate access requests and make decisions based on the access policies defined by the resource owner. In either case, an access token is generated and passed on to the requester which signifies the right to access. The main disadvantage of the transaction-based approach is that access decisions must be made by a centralized node. In contrast, the smart contract-based approach may invoke large overheads due to the creation of contracts between nodes.

8. **Relationship-Based Access Control (ReBAC):** In ReBAC, permission is granted based on the relationship between a subject and a device. For example, if a subject is the owner of a device, the device can access a resource. The relationship as an 'owner' of the device grants the permission [48]. The ReBAC is one of the recent models and it is gaining more attention due to its dynamic nature [15].

Table 2.2 summarizes the discussed models and their concerns to fulfill the access control requirements. As shown in Table 2.2, all access control models have limitations when adopted in IoT. These limitations indicate that more research efforts are desired in the field.

In addition to the models discussed above, a few more access control models can also be found in the literature [12], [19], [49], [50]. In History-Based Access Control (HBAC), an access decision is made dynamically based on the context of access history in a given state. The model requires a centralized authorization system such as a certificate authority in place [49]. Two access control models, Risk Adaptive and Proximity-Based Access Controls [50], are available for implantable medical devices. In the risk adaptive model, a decision is made by considering the risk factor evaluated by policies. In the proximity-based model, a device's programmer must be in close proximity with a patient to generate the key to decrypt the communications from the device. This model has a potential physical security issue that an adversary should not be near the patient [50]. The proximity-based model is used widely in implantable devices. Trust-based models allow devices to be attached to use spaces within a short period [19]. In this model, the access permissions are assigned to users based on their levels of trust. However, it is difficult to define how trust and relationships are established between users and devices. Examples of trusted-based models include Billing-Based Access Control and Privilege-Based Access Control. The billing-based approach is a business-driven control where a service is provided to any user who receives an adequate reward [12]. Identity does not matter in this model. In the privilege-based model, a decision is made based on an organization's policies, and the access is restricted only to particular users [12]. Trust is one of the important criteria in a heterogeneous environment such as IoT. It enhances both security and privacy [51]. However, trust systems in the IoT face challenges such as heterogeneity, scalability, and integrity [51].

Table 2.2: Access Control Models and Feature Matrix

Features	**DAC**	**RBAC**	**ABAC**	**OrBAC**	**CapBAC**	**UCON**	**ReBAC**	**BBAC**
Granularity	Coarse	Coarse	Fine	Coarse	Coarse	Fine	Fine	Fine
Context-Aware	No	No	Yes	Yes	No	Yes	Yes	Yes
Dynamicity	No	No	Yes	No	Yes	Yes	Yes	Yes
Complexity	More	More	More	More	Less	More	More	More
Distributed Nature	No	No	No	No	Yes	No	No	Yes
Interoperability	No	No	Yes	Yes	Yes	No	Yes	Yes
Delegation	No	No	No	No	Yes	No	Yes	Yes
Revocation	No	No	No	No	Yes	Yes	Yes	Yes
Scalability	No	No	Yes	No	Yes	Yes	Yes	Yes

2.5 Access Control Policies

An access control policy defines access permissions when an IoT device connects to a network. Access control policies primarily administer and manage the entire solution. The process of formulating access control policies for IoT networks should meet several objectives [12]. The process should not be too complex for a device owner to understand, and the usability should be of primary importance to the policy [52]. Further, IoT devices that connect to the network should be flexible to conform to the network's policies, so that risk is not introduced into the network. Due to the nature of the IoT, framing access control policies is domain-specific. Policies must adapt to a particular environment and its characteristics. For example, smart home products available in the market today can facilitate users to generate policies that allow access delegation. However, the generated policies might not be as fine-grained as the users expect and may lead to over-privilege [53], [54]. For instance, the Nest thermostat allows a homeowner to add a family member. This will give the family member complete access to the device, although the homeowner might not intend to give the family member full access [53]. Many access control solutions define the properties of delegation and context which are required for dynamicity. However, this generally happens at a single node when it comes to access decisions or administration.

Additionally, various commercial IoT services such as AWS IoT and NiagaraAx support ACL and role-based policies [55]. ACL-based policies are administered manually. It becomes unsuitable for the creation of roles and permissions when devices are added at scale.

2.5.1 Dynamic Policies Specification

Specifying access control policies at run time is desired to fulfill many requirements as listed in Table 2.1. This section summarizes the techniques adopted for dynamic policies specification in the IoT.

1. **Traditional Access Control Model-Based Approaches:** Liu et al. proposed an access control model for resource sharing in [56]. The approach is based on RBAC. The authorization mechanism uses a planning graph-based technique to search for an optimal route. The policy encompasses user roles, permissions, and resources. This approach is completely static and based on the experiments performed. Its performance is not optimal. In [24], Alkhresheh et al. designed a dynamic access control framework based on ABAC. A novel algorithm, namely the automatic policy specification algorithm, in which the policy is generated based on the extraction of the attributes from the subject, the object, the operation to be performed, and evaluated against a set of primitive facts, was presented. In addition, the policy enforcement algorithm adjusts the policies continually and automatically. In [57], Gabillon et al. proposed an ABAC-based framework for the MQTT protocol to which sensors could subscribe for topics. In their approach, the policy language is based on Shapes Constraints Language introduced by the World Wide Web Consortium. Although the policies are expressive and contextual by means of attributes, the administration is still static. In [58], Riad et al. extended XACML from adaptive policies to suit the distributed IoT environment. Their architecture follows the ABAC model and allows the policies to be adjusted at run time [58]. The gen-

erated policies are validated using the MD5 message-digest algorithm checksums. The scheme protects the IoT network from two attacks, i.e., the masquerade attack and the man-in-the-middle attack. Similarly, a conceptual framework that enforces access control policies in a smart health environment was proposed in [59]. This framework follows a centralized architecture but can refine policies at run time to ensure dynamism. The policy language is based on XML. The XML was utilized due to its flexibility to exchange policies between domains. The framework in [59] is based on the ABAC model. In fact, many approaches utilize the ABAC model to enforce access control policies due to its support for multiple attributes. However, the ABAC model, due to multiple attributes used in access control, may also have performance issues compared to others as shown in [60].

2. **Artificial Intelligence-Based Approaches:** Bertino et al. conducted a case study on XACML policies to analyze their model developed based on symbolic learning in the Generative Policy Model (GPM) in [26]. A public dataset, including XACML policies requests and responses, was used to perform the study. Based on the dataset, they generated a set of examples that contain ABAC parameters that were based on answer-set-grammar. Cunnington et al. proposed a centralized architecture based on GPM for connected and autonomous vehicles in [9]. The adopted method is based on inductive logic programming. Their solution does not generate access control policies, but it refines and stores policies dynamically. Liu et al. proposed a risk prediction-based access control model for the Internet of Vehicles (IoV) in [61]. In their approach, they use a generative adversarial network model based on LSTM to improve the training dataset. The risk prediction model is determined by the risk. The vehicle can access the requested resource if the risk is below a predefined threshold. This approach follows a centralized architecture. Yu et al. proposed a learning-based approach that learns contextual access control policies from the behavior patterns of multiple smart home devices in [62]. This

approach uses a federated learning framework that incorporates temporal modeling. In [63], Chu et al. proposed a multi-access control technique based on battery prediction with energy harvesting in IoT. The proposed solution utilizes an LSTM based deep neural network. It is designed for a wireless network where sensor nodes are dispersed geographically. The nodes are granted access to the base station based on the sensor node's battery state. In the proposed two-layer LSTM network, the first layer predicts and generates the battery level of the sensor node, and the second layer uses the channel information and predicted values to generate the access control policies.

3. **Blockchain-Based Access Control Approaches:** Blockchain has been explored to make access control decisions in IoT due to its distributed nature. A smart contract-based access control system was proposed in [64]. In the blockchain-based approach, a policy created by a resource owner is stored in the blockchain as a transaction. The policy is written in XACML and is transformed into a smart contract. To update or delete a policy, the contract is replaced with a new smart contract. In [65], Liu et al. proposed a distributed ledger-based approach to protect the privacy of IoT data. In the approach, policy updates are done through the edge node by adding a new policy to the blockchain, thus enabling dynamic access control. In [66], a distributed blockchain-based access control system is proposed for the smart grid domain. The approach consists of three layers: the first layer is the network layer, the second layer consists of the raw RBAC and ABAC policies, and the third layer consists of the distributed ledger. Context information updates to the PDP are performed by virtual auditors. These updates assist the PDP in performing dynamic access control decision-making. In [67], Zhang et al. proposed a smart contract-based access control approach utilizing the ABAC paradigm. The policies are not hardcoded in the smart contracts, allowing the approach to have less overhead. This solution also contains predefined functions to add, delete and

update the policies, thus assisting the concept of dynamic access control.

4. **Policies that Carry Data:** With the contextual nature and the amount of sensitive data transmitted and processed, IoT devices can also embed policies within the data. This embedded data policy allows for constant monitoring and revocation of access. First introduced in [68], sticky policies provide a data owner-centric approach for the IoT and allow users to embed policies into data. This concept was applied in many approaches in the field of IoT. For example, an approach called policy-carrying data was proposed in [69]. In the approach, the policy can specify information regarding permissions, obligations, and restrictions of the data, which brings dynamism. The policy language is based on first-order logic. However, the language is considered complex, and there is a need for a centralized server to evaluate both data producers and consumers. In [70], Sicari et al. use a middleware architecture to handle policy requests and responses by utilizing ABAC. The approaches in [71], [72] use sticky policies by utilizing the edge computing architecture. JavaScript Object Notation format was used to define policies, and end-to-end communication was encrypted to preserve data privacy. Sticky policies allow intelligent control over the authorization of IoT resources. However, it comes with limitations too. There are no established languages for policies due to the pinning of the policies with the data. It may also increase the computational overhead on the devices due to the encryption that is being used during data transmission [73].

2.5.2 Dynamic Policies Specification Challenges

The challenges faced by the current dynamic policies specification solutions are discussed below.

- **Centralized Architecture vs. Distributed Architecture:** A number of solutions including [23], [24], [26] have adopted centralized architecture to specify dy-

namic policies. For a heterogeneous environment, centralized architecture is complex to design, and does not scale well. Many IoT networks depend on cloud platforms for management. Therefore, it is recommended to utilize the technologies such as edge computing which supports distributed architecture. The solution proposed in [62] utilizes an edge computing paradigm [74] to learn context-aware access control policies from multiple smart homes.

- **Policy Generation, Decision, and Evaluation at Run Time:** Many solutions in the literature proposed their own approaches for dynamic policies specification. For example, the solution in [23] uses supervised machine learning to classify device access behavior based on a real-life data set. Access control solutions should be able to make access decisions based on policies automatically, in case the connection to the backend system is lost [9] or in a large-scale project like a smart city where numerous devices are added at scale [75]. A solution should consider these scenarios when enabling dynamic policies specification.

- **Eliminating Policy Violation and Policy Conflict:** IoT devices are often used to automate physical processes such as detecting water leaks, adjusting temperature, controlling security cameras, and enabling autonomous driving. Hence, dynamic policy specification or policy automation should address policy conflict and policy violation identified from the generated policies. Policy validation provides the opportunity to invoke several issues related to the security of the devices and physical safety.

- **Selection of Required Features:** Almost all the discussed solutions use machine learning approaches for extraction or refining policies at run time. The machine learning-based solutions depend on a specific set of defined features for operations. The features used in machine learning in IoT include, but are not limited to, the contextual attributes of the subject that requests access, resources, and other en-

vironmental attributes. When implemented in real-time, frequent requests to the current state or attribute values may potentially reduce the performance of the devices. Therefore, the machine learning solutions must consider the memory and processing capabilities while performing feature selection.

- **Accuracy of Real-Time Classification:** Access control authorizes a user or a device's request to access a particular resource. Hence, in a classification scenario, a machine learning solution must predict a request with utmost accuracy. Otherwise, at times of misclassification, there is a chance that a legitimate subject is denied access. In generative models, it is believed that any policy that is being generated should not conflict with the existing policies in the repository and should not violate the security and privacy requirements of the network. Designing a solution to verify these issues automatically is a challenge.

- **Using Balanced Dataset:** Unfortunately, identifying a relevant publicly available dataset is a challenge for access control research in IoT. Various constraints such as security and privacy might be part of the reasons. However, the machine learning models should learn from a balanced dataset to provide accurate classification or policy generation. For example, Bertino et al. evaluated their proposed solution with the help of a noisy XACML dataset [26]. Their models led to issues such as overfitting. Hence, a well-balanced dataset is highly essential to realize the true potential of a machine learning solution.

- **Real-Time Implementation:** A number of the proposed solutions have not been evaluated in real-time. For example, studies such as [25], [76], [77] have been proposed as generalized frameworks that can be utilized for dynamic policy specification in the IoT. Solutions tend to behave differently in a test environment and a real-time environment. Consequently, when they are implemented in real-time, the actual issues and the challenges the solutions may face shall be captured enhancing the scope

for further research.

2.6 Access Control Research Challenges and Future Directions

This section summarizes the research challenges in access control and points out future research directions in the IoT.

2.6.1 Research Challenges

Table 2.2 reveals the research gaps in access control in the IoT. Many challenges exist in designing an ideal access control solution to fulfill the access control requirements identified in the IoT.

1. **Device Identification**: Access control assumes IoT devices can be uniquely identified and access control policies can be applied to network traffic. As users are identified in a digital network by their unique identities, IoT devices also require their unique identities when connecting to a network. While users are often identified by something the users know, something the users have, and something the users are IoT devices can only be identified by something they have. A common technique to identify a device in a network is using the device's MAC address. However, the MAC address is easy to be spoofed. Given the heterogeneity and the need to protect the data that IoT devices collect, device identities need to be addressed before access control [78].

2. **Relationships and Access Control**: Relationships such as user-to-user, user-to-device, and device-to-device relationships can be utilized for identity management and access control. It is expected by many consumers that the IoT device manufacturers include the concept of relationships for access provisioning. Thus, Identity

Relationship Management (IRM) is gaining attention and has been identified as a promising IAM system for the IoT [79].

3. **Policies Specification and Automation**: A comprehensive review of the policies specification in the IoT reveals that the existing solutions lack the dynamicity in policy generation, decision, and evaluation [80]. Machine learning can be utilized for policies automation. With automation, there is no need to edit policies manually when devices are added at scale. Therefore, the use of machine learning will directly help in achieving dynamicity in an access control solution although machine learning-based approaches also face the challenges as discussed in Section 2.5.2.

4. **Interoperability Issues**: In an IoT network, not all the devices come from the same manufacturer. Devices should be interoperable when connected to a network, allowing access control to function as expected.

5. **Blockchain and Access Control**: The operation of blockchain for access control in IoT is still in its infancy. Due to its distributed nature and property of delegation, blockchain is well suited to IoT networks. Research on the blockchain is needed for access provisioning in IoT environments.

6. **Computational and Communication Overhead:** IoT devices come with a number of constraints, particularly in terms of memory and processing power. The solutions proposed should not invoke any overhead on the devices, which may reduce their performance. One way to overcome this challenge is to bring the computation and storage to the edge, which may enhance the network's performance.

7. **Access Control vs. Security and Privacy**: The security of access control models is also a concern. Access control is one of the important services in security. Therefore, the access control solution itself should be resistant to attacks. The security flaws in access control may occur in many places, including design, protocols,

implementations, and configurations. Although many access control models have been proposed for IoT, limited research has been conducted on access control security analysis [81]. Due to the importance of access control for any network, access control security analysis is desired. Moreover, an IoT network tends to consist of a large amount of sensitive information. Thus, the access control policies must comply with privacy regulations such as General Data Protection Regulation (GDPR) and Health Insurance Portability and Accountability Act (HIPAA).

2.6.2 Future Directions

Access control is essential to secure the IoT. Future research is desired in the following areas:

1. **New Scalable Schemes for Identities of Things**: IDoT includes identities of both users and devices. Identities of users have been studied extensively. Identities of devices are still developing. Due to the vast number of devices available, the scalability of the new schemes is essential. Most of the identification schemes are based on symmetric or asymmetric cryptography. However, both symmetric and asymmetric cryptography have limitations when used for identifications [82]. New schemes for uniquely identifying devices need to be further studied.

2. **Novel Multi-factor Authentication (MFA) Methods for the IoT**: MFA provides alternative approaches to verify a user's identity. It has been proven to be effective for many internet-based applications and services. MFA is essential and desired for the IoT. However, using MFA in IoT applications is challenging since many IoT devices do not come with screens and keyboards. Novel MFA methods are desired for small form factor IoT devices that do not have input devices.

3. **Utilizing Relationships for Authentication and Access Control**: Relationships among users, devices, applications, and services can be used to describe the

dynamic intelligence that an IRM system seeks. The dynamic intelligence can be embedded into the context when authentication and access control are requested. However, there are challenges on how to characterize and define relationships and how relationships can be used for authentication and access control.

4. **Standards and Interoperability**: Global standardization bodies need to define specifications for a borderless identity and access management system adopted by the communities. The system needs to be built in a modular and pluggable manner without requiring a single organization to maintain its system. The standardization of identity and access management systems will also help resolve the challenges when multiple information systems are adopted in an organization.

Chapter 3

Methodology

This chapter discusses the research methodology adopted in this dissertation. First, we describe the methodology we adopt for this research. Next, we briefly discuss Generative Adversarial Network (GAN) architecture, the GAN variant used for tabular data generation, and how it can be utilized to generate access control policy data. We then discuss the types of GAN models we plan to use for our experiments, the datasets, experimentation strategy, and evaluation strategy.

Our research addresses the challenge of generating contextual and usable access control policies for the Internet of Things. In Chapter 2, we identified the scope and need for further research in the domain of automated policy generation for access control in IoT. While architectures like the Generative Policy Model (GPM) have been proposed for automated policy generation, they follow a centralized approach. However, our literature review reveals that IoT networks are heterogeneous and distributed. We propose the use of Deep Generative Models (DGMs), specifically Generative Adversarial Networks (GANs), which can be effectively deployed in various architectures, including distributed environments. On the other hand, we also plan to identify how realistic are the generated policies and if they can be used for real-time scenarios.

We formulate our research as a case study to explore the feasibility and challenges of the proposed approach. As part of this, we implement two baseline tabular GAN models, which serve as the foundation for our proposed solution. This structured approach

allows us to systematically evaluate the effectiveness of our solution and compare it to existing approaches. We conduct several experiments using the GAN models and utilize quantitative methods to analyze and evaluate our approach. Finally, we also evaluate the quality of the generated policy using manual evaluation.

3.1 Theoretical Background

We consider the proposed automated policy generation as a synthetic data generation problem. We derive the concept of generative adversarial networks (GAN) [83]. GANs consist of two neural networks, namely a Generator and a Discriminator. The generator takes, as input, a random Noise and, through a number of iterations, learns to generate samples as realistic as the original data. Meanwhile, the role of the discriminator is to distinguish whether the generated data is real or fake. Thus, the two networks are trained alternatively together to play a Minimax game to compete with each other forming an adversarial learning process. This architecture is based on game theory. [85]. Hence, we provide the preprocessed policies as input data to the GAN model. The model trains with the help of neural network layers and produces synthetic policies. We then perform an evaluation of the obtained policies. GANs possess several advantages. They will be able to effectively generate synthetic data by training from limited real data. Additionally, IoT networks are distributed in nature. The literature suggests there may be situations where a device is not able to contact the policy repository. Then, devices should be able to generate policies themselves. GANs can also be implemented in a distributed architecture where local generators and discriminators shall be deployed in individual local nodes and a model globally available at the server.

According to [84], tabular data is the most commonly identified data type in businesses today and the second most common format in academia. The need for synthetic data has risen in the past few years, and several techniques have been introduced to gen-

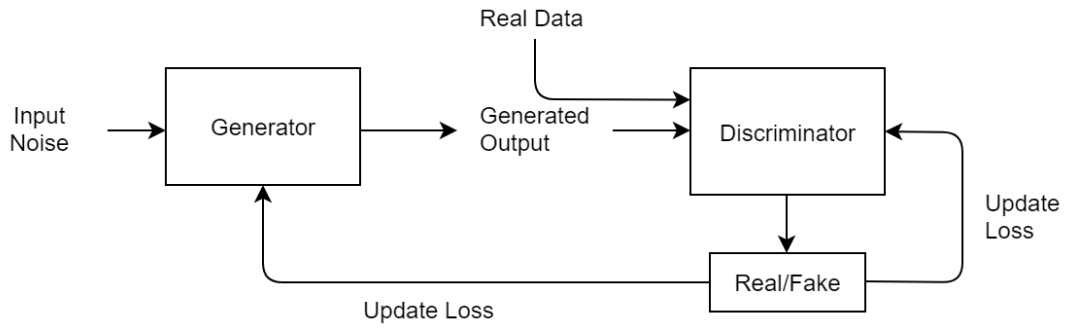

Figure 3.1: Basic Structure of a GAN

erate synthetic data from learned distributions, from modeling multivariate probability distributions, hidden Markov models, Gaussian copulas, and randomization-based methods [84]. Due to their success in generating realistic image data, neural network-based generative techniques, such as Generative Adversarial Networks (GANs), have also been utilized widely in language modeling and the synthetic generation of tabular data. Access control, especially Attribute-based access control data, is similar to tabular data. The feature set of tabular data may contain both continuous and discrete datatypes. Similarly, access control data takes both continuous and discrete values such as user attributes, environmental attributes, and operational attributes. We utilize two baseline tabular GAN models to experiment and analyze the efficacy of these models in generating access control policies. The figure below shows the structure of a tabular GAN with conditional inputs.

3.1.1 Conditional Tabular GAN

The Conditional Tabular Generative Adversarial Network (GAN) was introduced by [85]. It is an extension of the Tabular Generative Adversarial Network (TGAN) [84] that employs a conditional generator to produce synthetic data conditioned on one of the discrete columns present in real data. Both the generator and discriminator comprise two fully connected hidden layers. The generator utilizes Batch Normalization and Rectified Linear Unit (ReLU) for activation, while the discriminator employs Leaky ReLU activation and dropout on each hidden layer. Lastly, in the generated data, the scalar values are

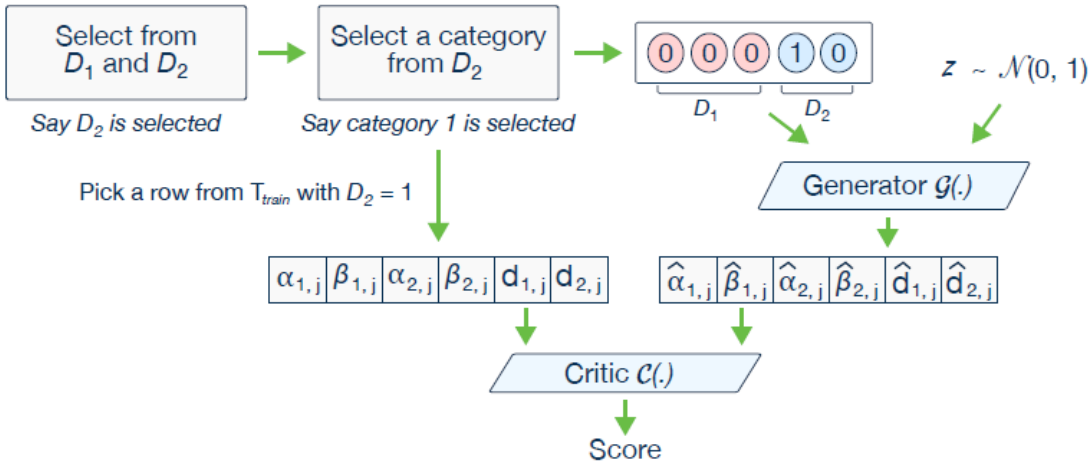

Figure 3.2: Structure of a Tabular GAN with Conditional Inputs [85]

produced using hyperbolic tangent (tanh), and the discrete values are generated through the gumbel-softmax function.

3.1.2 Copula GAN

The Copula GAN is a variant of CTGAN and has been introduced by [86]. This approach utilizes probability integral transforms and establishes relationships between uniform marginals and the GAN framework. This way, a probabilistic G model is implemented which estimates univariate marginals and multivariate copula density in the learning process. In Copula GAN, the data is transformed using the distribution that best achieves uniformity and their relationship is modeled after the transform. This architecture applies the Gaussian copula CDF-based transformation [87] to enhance the learning process. Synthetic data is generated by implementing inverse transformation sampling and randomly generating the samples from a uniform distribution.

3.2 Learning Background Knowledge

Over the years, GAN models have proven to be very efficient in generating high-quality synthetic data. In other words, generative models can perform a good approximation of the distribution of the learned data. However, it is critical that the models learn the background knowledge required to generate data and comply with the constraints [88]. This is essential, especially when generative models are deployed in real-time information systems like IoT networks. Verifying compliance is as equal as evaluating how realistic the generated data is. In access control for IoT, specifying constraints based on the underlying business rules is critical. Any policy generated and enforced that violates the business rules results unauthorized access. In real-time environments like IoT, this may impact physical security or safety. In this research, we plan to incorporate constraints in the GAN models we implement by considering any rules explicitly mentioned in the literature or by learning the background knowledge of the domain we plan to experiment.

3.3 Proposed Experiments and Evaluation

3.3.1 Datasets Description

We employ two datasets for our research.

CAV Policies: We use the CAV policies dataset published by [9]. It is created via policy learning tasks for connected and autonomous vehicles (CAVs). CAVs are an instantiation of IoT. The dataset was created to evaluate a Generative Policy Model where CAVs learn and generate high-level policies to operate autonomously during various unseen scenarios. Each policy includes information pertaining to who is responsible for controlling the vehicle, monitoring the driving environment, and who is the fallback responsibility for the driving task. Depending on varying contexts, the vehicle is controlled by the human driver, the system, or both. The monitoring and fallback responsibilities are held by either

the driver or the system. Three environmental conditions, such as weather, visibility, and traffic congestion, are considered, with values of each ranging from 0 to 10, both inclusive, and they are represented collectively by their weighted average. Eventually, a policy is either approved or rejected depending on the values pertaining to three facts: the level of autonomies (LOA) of the driving task, vehicle, and region. A policy is approved only if the driving task is lesser or equal to both vehicle LOA and region LOA. Otherwise, the policy is rejected. Although the data is not used for an access control system, the characteristics are fully consistent with an ABAC model. Hence, we choose to utilize the dataset to evaluate access control policy generation. The dataset contains 239,480 unique policies and eleven features by which a decision is computed.

Table 3.1: CAV Policies Dataset Description

Column	**Datatype**	**Description**
Driving Task Type	Categorical	adaptive speed control, high speed cruising, parking, performing turn, residential driving
Control	Categorical	human, system, human and system
Monitoring	Categorical	human, system
Fallback	Categorical	human, system
Weather	Numerical	0-10
Visibility	Numerical	0-10
Traffic Congestion	Numerical	0-10
Environmental Weighted Average	Numerical	Weighted average of weather, visibility, and traffic congestion
Driving Task LOA	Numerical	0-5
Vehicle LOA	Numerical	0-5
Region LOA	Numerical	0-5
Result	Categorical	approved, rejected

Amazon Access Logs Dataset:

The Amazon Access logs dataset was released in 2013 as part of a Kaggle competition [89]. The purpose of the competition is to automate the access provisioning and revocation process as an employee moves through different roles in a company. The dataset consists of 32769 records. Each record pertains to an access request to resource by a user and

its corresponding decision. There are about 9000 unique users and eight user-attribute combinations that are used to determine if a user has access to a specific resource. The user attributes include the Role Code, Role Title, Role Family, Role Family Description, Role Department Name, Role Rollup 1, Role Rollup 2, and the reporting Manager ID. The Resource ID specifies the resource an employee is requesting access to. According to [89], the employees in a given role will access the same or similar resources. and the Action column specifies the access decision. If the value is 1, then the access is approved, and if the value is 0, then access is denied. All the features in the dataset are of type Categorical.

Table 3.2: Amazon Access Logs Dataset Column Description

Column	**Description**	**No. of Categories**
ACTION	Access decision. 1 if the request is approved. 0 if the request is denied	2
Resource ID	ID assigned for each resource	7517
Manager ID	Employee ID of the manager of a given employee	4243
Role Rollup 1	Company role grouping category 1	128
Role Rollup 2	Company role grouping category 2	177
Role Department Name	Company role department description	449
Role Title	Company role title description	343
Role Family Description	Company role family extended description	2358
Role Family	Company role family description	67
Role Code	Company role code. This value is unique to each role.	343

3.3.2 Data Processing

Our data pre-processing involved converting the metadata into the appropriate datatypes to be consistent with the data. The SDV requires extracting the metadata and specifying it explicitly as a parameter to the model before training. For the CAV Policies dataset, the metadata API recognized the driving_task_type as a Vehicle Identification Number

(VIN) and a primary key. Similarly, due to the lesser range of values in driving_task_loa, vehicle_loa, and region_loa, the API recognized them as categorical types. Hence, we transformed all the features into their appropriate types as categorical and numerical respectively and updated the metadata. For the Amazon Access Logs dataset, except for the ACTION column, the metadata API identified all other features as numerical types. Hence, we converted all these features into categorical. The RESOURCE_ID column contains more than 7000 categories. Hence, we used the Reversible Data Transform (RDT) library to encode the feature using the label encoder. For CTGAN, all other features categorical features are one-hot encoded. For Copula GAN, the gaussian copula transformer was used on both numerical and categorical variables and then the RDT transformer was used for appropriate transformation.

3.3.3 Constraints

We created custom constraints for the training of both datasets. For the CAV Policies dataset, the primary constraint is that a policy is approved only if the driving_task_loa value is lesser than or equal to both the vehicle_loa and region_loa values. Otherwise, the policy should be rejected. This is explicitly mentioned in [9]. We found it necessary to include the constraint for the environmental weighted average. The authors provided the weights for all three features: weather, visibility, and traffic congestion. For the Amazon Access Logs dataset, the constraints are not provided explicitly, but based on analyzing the data, we found one that can be a potential constraint. For any given ROLE_CODE category, the values of ROLE_TITLE and ROLE_FAMILY is the same. The aforementioned constraints are coded using the SDV custom constraints class for both CTGAN and CopulaGAN.

3.3.4 Training Strategy

We follow different strategies to train both datasets. As mentioned previously, the CAV Policies dataset consists of 239,480 unique policies. We decided to train the models on 20 percent of the entire data and 40 percent of the entire data to analyze the difference in the distribution a model is able to learn and generate. We consider the Amazon Access Logs dataset as is. The reason is there is a separate test set that we could compare if the model is able to generate similar distributions as the real data. We trained all the datasets without applying constraints first and then with constraints.

3.3.5 Evaluation Strategy

Once the models are trained, the first evaluation we perform is whether the synthetic data generated are completely new or whether and how many rows correlate with the training data. Next, an important evaluation we consider is how many rows in the synthetic data are in compliance with the defined constraints and whether and how many rows have violated the constraints. The third evaluation we perform is the efficacy of the supervised machine learning algorithms when trained on synthetic data and tested on real data, and vice versa. This is called the Train Synthetic Test Real strategy used in various publications such as [88], [90]–[92].

3.3.6 System Configuration

We performed all our experiments in an Ubuntu box (v22.04.1) hosted on VMWare. All model training was performed on the Nvidia Tesla T4 GPU implemented in Jupyter Notebook on the Anaconda Navigator platform. All versions of trained models are saved in the pickle format utilizing the CPU. The CPU is an Intel Xeon Gold 6242 @ 2.80 GHz x 4. The available system memory was 32 Gigabytes.

Chapter 4

Results and Analysis

This chapter details the results we obtained performing the experiments described in the previous chapter and our analysis of the results.

4.1 Data Generation

We obtained results for different dataset sizes mentioned in the previous chapter for the GAN models, CTGAN, and CopulaGAN. The CAV Policies dataset contains both numerical and categorical data. However, the number of categories is significantly less in every corresponding feature. Hence, we did not perform any transformation on the data. We tried training the model for varying batch sizes, learning rates, and the number of epochs. We found that the models were able to learn the data distributions significantly well around 300 epochs, and after that, there was no further improvement in the training. Hence, we decided to train the models for both dataset sizes for 300 epochs, with the generator and the discriminator learning rate of 2e-4 and a batch size of 700.

4.2 Analyzing Synthetic Data Characteristics

Initially, our focus was to verify the quality of the generated data. That is if the records in the generated data are different than the ones in the real data. the continuous features in the CAV policies dataset to see how well the model has learned the data distribution

and whether it can adhere to the boundary from the real data to generate the values within that range. Next, we also considered the number of records based on the values in the "result" column to see how many records resulted in an 'approved' decision and how many records resulted in a 'reject' decision. An important consideration is to find the number of violations when the constraints were not specified. The Amazon Access Logs dataset is different, with all the features being categorical and eight features representing the user attributes. For this dataset, we focused on identifying how many unique users were present in the generated data, how many records resulted in '0', and how many records resulted in '1' in the ACTION column.

To compare the generated data with the corresponding data on which the model was trained, we generated an equal number of samples based on the corresponding training dataset sizes. Table 4.1 shows the count for the rows that resulted in 'Approved' and 'Rejected'. As the results show, there is an increase in the difference between the "Approved' and 'Reject' decisions in all the scenarios, and 'Rejected' dominates in all the cases. We recorded the result count only for our experiments specifying the constraints because approved decisions that violate the constraints are not valid.

Table 4.1: CAV Policies - No. of Approved and Rejected Policies

Model	Training Set Size	Result Count	
		Approved	Rejected
CTGAN	20 percent	21081	28919
	40 percent	37376	58456
CopulaGAN	20 percent	22486	27514
	40 percent	37565	52435

Table 4.2 shows the synthesis score for all experiments performed on the CAV policies dataset. The row synthesis is a measure that identifies whether each record in the synthetic data is new, or it exactly matches the training data. The score ranges between 0 and 1, with 0 being the worst and 1 being the best. The scores obtained on all our experiments are almost close to perfect. The models trained on dataset size of 20 percent obtained a

score of approximately 0.99 each and the models trained on the dataset size of 40 percent obtained a score of approximately 0.98 each. For the Amazon Access Logs dataset we obtained a similarity score of 1 on all our experiments.

Table 4.2: CAV Policies - Comparison of Synthesis Scores

Model	Training Set Size	Score-(unconstrained)	Score-constrained
CTGAN	20 percent	0.9944	0.9920
	40 percent	0.9859	0.9830
CopulaGAN	20 percent	0.9967	0.9940
	40 percent	0.9871	0.9837

Next, we analyzed if and how many violations exist on the generated data. Table 4.3 shows the results obtained for our experiments on different dataset sizes without applying constraints and with constraints applied while fitting the model. Models trained based on both CTGAN and CopulaGAN have resulted in a significant number of violations. When the models were trained on the dataset size of 20 percent, there were about 17 percent and 19 percent violations on the generated data for CTGAN and CopulaGAN, respectively. Similarly, the same models trained on the dataset size of 40 percent resulted in about 18 percent violations for CTGAN and CopulaGAN respectively. However, when these models were trained after applying the constraints, all models were able to generate data that was in compliance with the constraints.

Table 4.3: CAV Policies - Violations Comparison

Model	Training Set Size	Violations	
		No Constraints	With Constraints
CTGAN	20 percent	8313	0
	40 percent	17447	0
CopulaGAN	20 percent	9178	0
	40 percent	17700	0

The Amazon access logs dataset contains all categorical features. Also, it is highly imbalanced with more records containing access approvals. One more challenge is that

some features contain thousands of categories. Hence, one-hot encoding is not an optimal processing technique for those features, and the dimension of the dataset would become large. We trained the model by transforming the data into two types of encoders, one-hot encoder and label encoder. While analyzing the generated synthetic data, we identified that the models generated the synthetic data with the imbalance, although CTGAN uses the training by sampling technique. We trained the models with and without constraints. When we trained the model without constraints, we identified that almost the entire generated data violated the constraint requirement. We wrote a custom logic for the models to comply with the fact that for a given ROLE_CODE, the corresponding ROLE_TITLE and ROLE_FAMILY must be the same for all instances of that role code. However, the custom logic did not fit well with both the models and generated data again violated the constraints. Researching further, we identified a fixed combination constraint available as part of the SDV library. Both models were able to comply with this constraint.

In the tables 4.4 and 4.5 respectively, we report the number of users, and the number of rows generated for each action for both the CTGAN and CopulaGAN models. As mentioned earlier, the number of records containing approved decisions is significantly higher than the number of records with denied decisions. In Table 4.5 we report the number of unique resources generated from both the models.

Table 4.4: Amazon Access Logs - Access Decisions by Model

Model	No. of Users	Action Count
CTGAN	32765	Approved – 30010, Denied – 2759
CopulaGAN	32754	Approved – 30780, Denied - 1989

Table 4.5: Amazon Access Logs - Unique Users and Resources

Model	No. of Users	No. of Resources
CTGAN	32765	6507
CopulaGAN	32754	5403

4.3 Visual Evaluation

To understand the generated data visually, we plotted the correlation tables for both the real and generated datasets, and the difference between them using the visual evaluation tool available in the table evaluator library. For the CAV Policies, we generated correlation tables for all the conducted experiments. The correlation tables of our interest are the ones for data we generated by applying the constraints. All models we trained were able to generate data that is similar to real data. As seen in the figures, the differences in correlation are not high. For example, in the real data, driving_task_loa has high correlation with driving_task_type, control, monitoring, and fallback. The trained models generated data with almost the same amount of correlation between the features. This indicates that the models were able to capture the correlation between various features. However, considerable difference in correlation is identified with vehicle_loa and region_loa. Overall, CopulaGAN generated data contains more difference in correlation when compared to the data generated by CTGAN.

On the other hand, for the Amazon Access Logs dataset, we identified significant differences in correlation between the real and the generated data. The Amazon dataset provides a different perspective compared to the CAV policies. The real data contains more correlation between different features. However, when we examine the generated data by both the models, the correlation is significantly. This also indicates that the generated data may have a different distribution than the real data. We identified this difference when we treated all the features as categorical. When we trained the models by treating all features except the target, as type continuous, we identified very less difference in correlation. We present only the visualizations of our experiments when features were treated as type categorical.

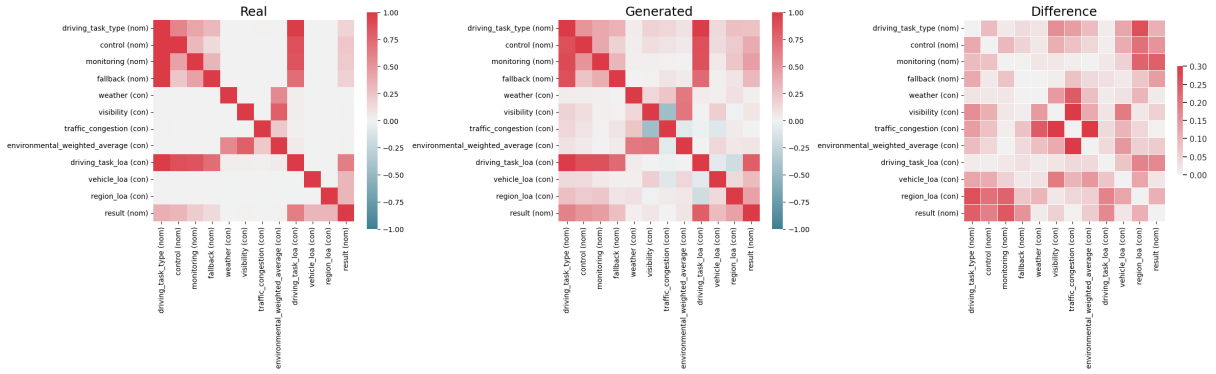

Figure 4.1: CAV Policies - Correlation Difference when Trained CTGAN on Dataset Size 20

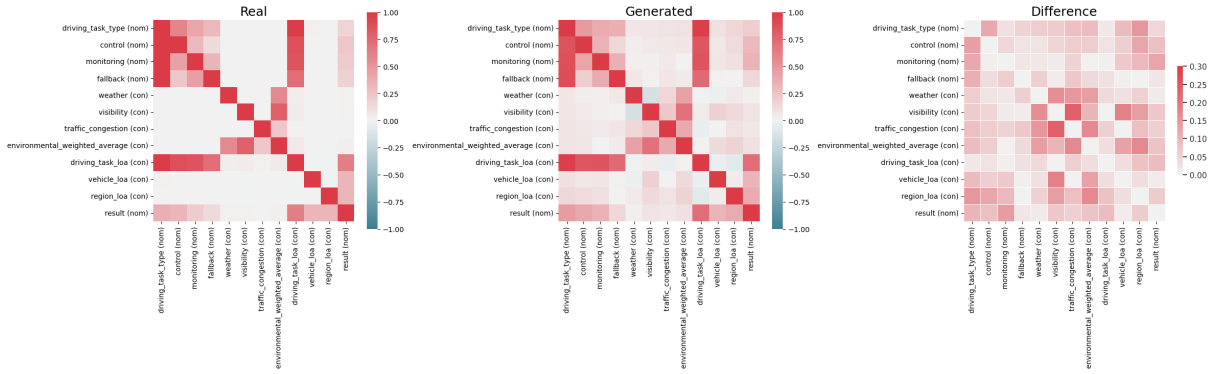

Figure 4.2: CAV Policies - Correlation Difference when Trained CTGAN on Dataset Size 40

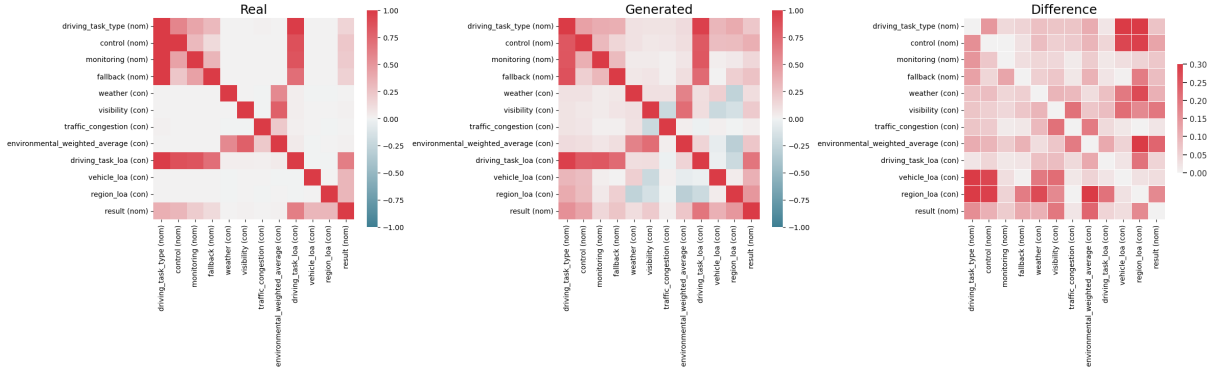

Figure 4.3: CAV Policies - Correlation Difference when Trained CopulaGAN on Dataset Size 20

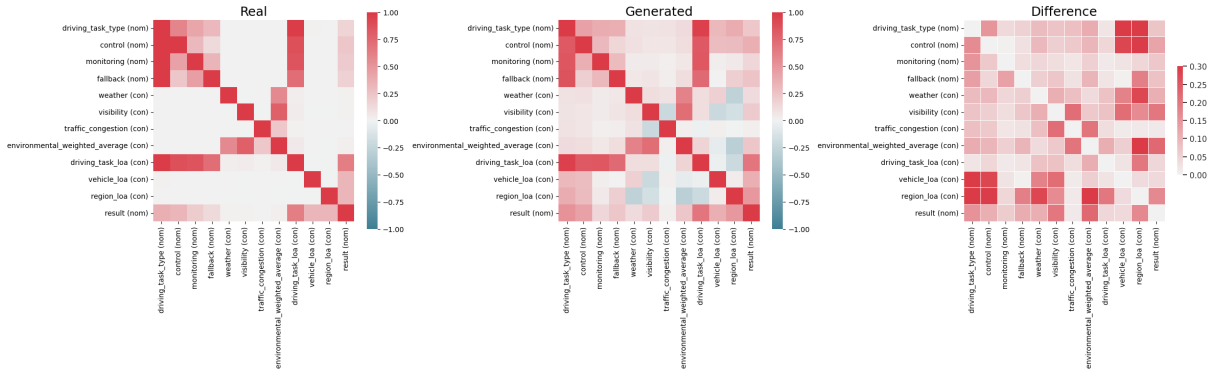

Figure 4.4: CAV Policies - Correlation Difference when Trained CopulaGAN on Dataset Size 40

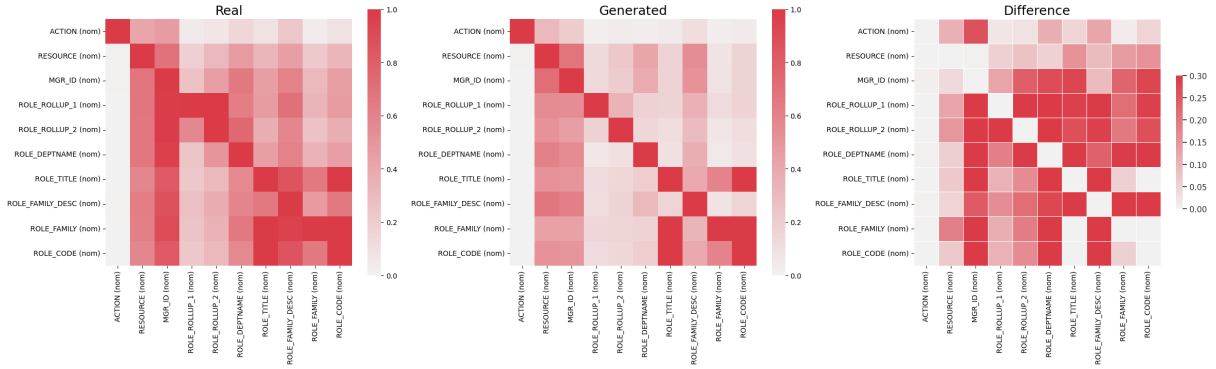

Figure 4.5: Amazon Access Logs - Correlation Difference when Trained on CTGAN

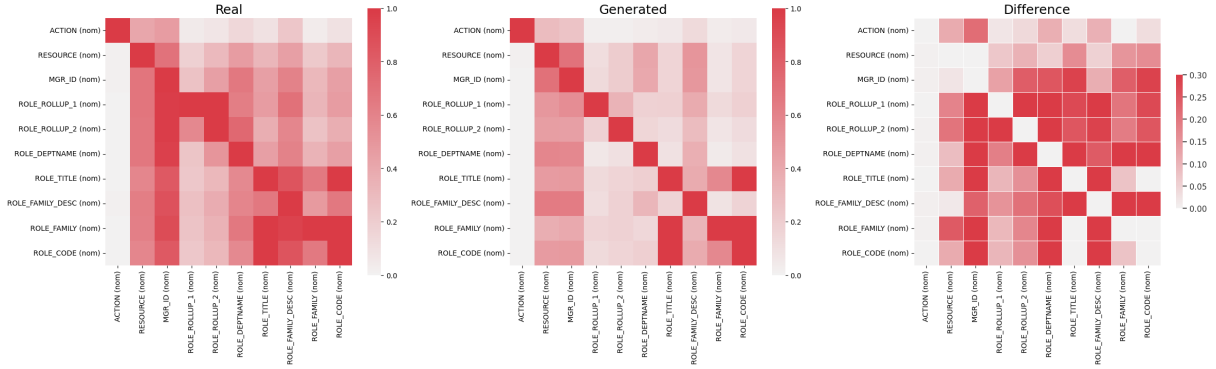

Figure 4.6: Amazon Access Logs - Correlation Difference when Trained on CopulaGAN

4.4 Qualitative Evaluation

We analyzed the generated data from both the datasets qualitatively. For analysis, we only consider the data generated by applying the constraints.

CAV Policies:

As mentioned before, we used 20 percent and 40 percent of the CAV policies dataset for training the models. For comparison, we first consider the attributes, driving_task_type of both real and generated data. There are five driving task types totally. The real datasets of both sizes contain equal distributions of every driving task type with approximately 9000 and 19000 for each type respectively. While analyzing the generated data, we identified varying counts for each driving task type, although the difference is not very significant. The generated data of both sizes from both CTGAN and CopulaGAN were able to capture the range of the attributes of continuous type and were able to generate data only within the learned range. Additionally, the environmental_weighted_average is also computed appropriately by adhering to the specified constraint.

In the GPM approach, there are two tasks. In the first task, the system should first determine who holds the responsibility for controlling the vehicle, monitoring the environment, and has the fallback. In the second task, the Driving Task LOA is assigned depending on the environmental conditions. According to the ground truth defined in [], if the weather conditions are poor, then the driving task is assigned for the system to perform all three functions in the first task. However, the policy is either approved or rejected depending on the constraint mentioned before.

Let us consider the following generated policies:

1. The requested driving task is adaptive cruise control. For this request, the system identifies that control should be jointly performed by the human driver and the system, and the driver must take on the monitoring and fallback responsibility. The weighted average of the environmental attributes is 70. Finally, the Driving Task LOA is less than

or equal to the Vehicle LOA and Region LOA, which satisfies the constraint.

2. In another policy, the requested driving task is residential driving. For the first task, CAV identifies that the system performs all three, controlling and monitoring the vehicle, and the fallback responsibility. The system identifies the LOA for the driving task, the LOA of the vehicle, and, depending on the environmental conditions, the minimum required LOA to operate in the region the vehicle is in. The weighted average is 2.1. However, this policy is rejected because both the vehicle LOA and the region LOA are less than the LOA of the driving task.

To be noted that the same policies were generated with decisions that violated the constraints when the models were trained without constraints. When we examined all our generated dataset, we did not identify any policy conflicts. This is due to the presence of environmental weighted average feature. It is a value calculated as the weighted average of weather visibility and traffic congestion. Remember that the initial task is to identify the level of autonomy of the driving task given the task type and the weather conditions.

As discussed in Chapter 3, the original CAV policies dataset contains 239, 480 policies. We used 20 percent and 40 percent of the dataset to train the GAN models. We used the remaining data as holdout data to compare if the generated policies match with them. In this case, we considered the holdout data as unseen scenarios. We were able to identify the generated policies are close to those present in the larger holdout data.

Amazon Access Logs:

The nature of the Amazon Access Logs dataset is different from the former. All given attributes are of type categorical. The features RESOURCE_ID, MGR_ID, and ROLE_FAMILY_DESCRIPTION contain thousands of categories while the other features contain more than hundred categories. CTGAN performs one-hot encoding of the categorical features. Performing one-hot encoding on large number of categories increases the training time. Also, this increases the dimensionality of the feature space, and the size of the trained model significantly becomes large. Using Label Encoding is easy to train,

but the categories are not in a given order. All of them are nominal. The closest work we found that used this dataset to generate ABAC data is [93]. In their approach, the data preprocessing steps were not detailed. Per the visualizations, all the features, except ACTION, were treated as numerical instead of categorical. We experimented treating the features as both numerical and categorical. When we considered all the features as categorical, the trained models generated data by adhering to the constraints. But applying the given constraint to continuous data is challenging. The only way we could apply constraint is by specifying a range until which the difference could be tolerable. However, this may result in challenges in a real-time scenario in identifying the category to which the corresponding value belongs.

For the Amazon Access Logs dataset, we were able to identify significant differences in the generated data with and without training constraints. As we described in Chapter 3, our initial data exploration revealed that the attributes ROLE_TITLE and ROLE_FAMILY contain the same values for all the instances for a given ROLE_CODE. Both CTGAN and CopulaGAN models generated data with more than 30000 violations for this pattern. When we trained the models by adding a fixed combination constraint, then all the records in the generated data were in compliance with this constraint. However, there was a significant difference in the data distribution. This could be due to the presence of only categorical features. Also, it is unknown how realistic are the generated access requests. For example, in a real-time organizational scenario, it is unknown whether the generated ROLE_CODE that belongs to a ROLE_FAMILY exists within a given department. The presence of more background knowledge regarding the data would be beneficial to model the relationship between the features.

Let us compare two policies in the Amazon Access Logs on how they were generated with and without applying constraints.

1 16619 56723 117961 118343 118301 307024 118568 6725 117898

1 16619 85475 117961 118266 118846 122290 136840 118453 117898

1 16495 21824 117993 120141 120144 124537 133986 118638 118705

1 16495 17326 117961 118386 6725 118278 296252 118638 118705

In the policies mentioned above, two role codes were selected, namely "117898" and "118705". The policies were generated by CTGAN without applying the constraints. We just considered a pair of each role codes for analyzing here. If we look at the role family and role title, both are different in both the instances of the two role codes, which is different than the real data. However, when we constrained the models, CTGAN and CopulaGAN, we obtained the policies like the pair mentioned below. The role family and role title for a given role code is the same. This is the case for all instances for all role codes in the generated data.

1 74818 2904 130684 126975 122215 117896 287351 117887 117898

1 74818 6227 118976 118085 118825 117896 278266 117887 117898

An important consideration on the Tabular GAN models is that, they can generate synthetic data only based on the sample space it encounters during the training process. It is not possible for generating entirely new data as with the GAN models that generate images. This is due to the discrete nature of the text data, and the inherent nature of Tabular GANs to learn and generate continuous samples within the range it learns during the training process.

4.5 Machine Learning Efficacy

We analyzed how efficient are the machine learning classifiers if the generated synthetic data are replaced and trained on machine learning classifiers for classification tasks. Both the datasets, CAV Policies, and Amazon Access Logs, possess a target column that a classifier could potentially predict after learning from a data distribution. We split both the real and generated datasets from all experiments into training and test sets, 80 percent and 20 percent, respectively. We followed the approach presented in [90], in which, the

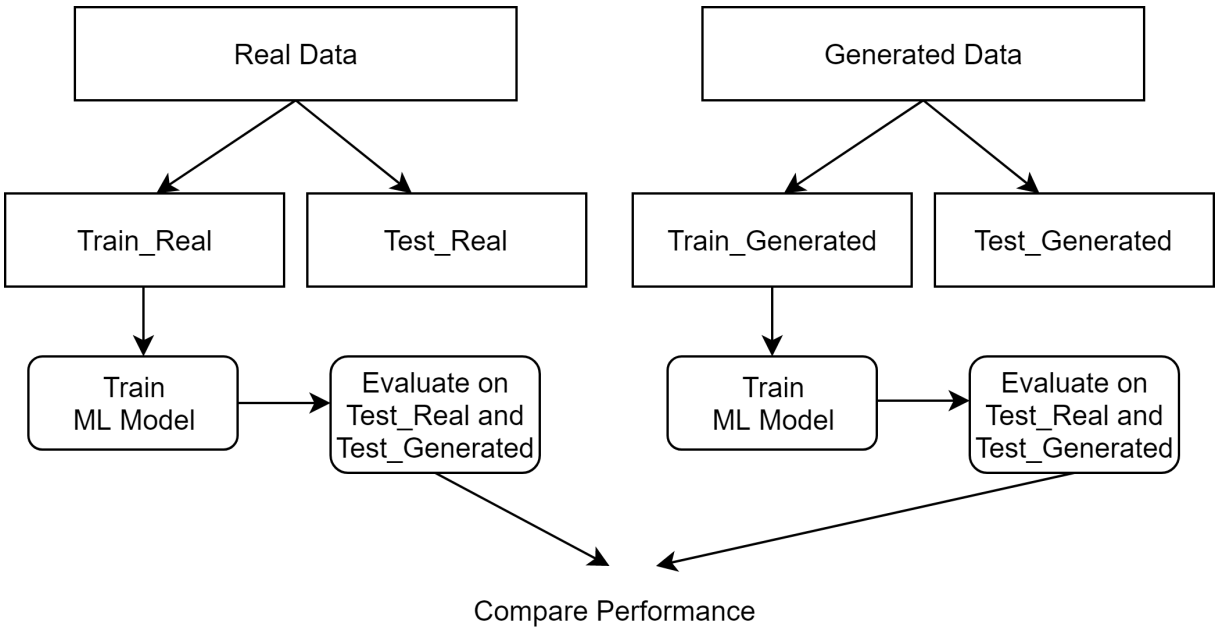

Figure 4.7: Workflow to Analyze Efficacy of Supervised ML Classifiers [90]

evaluation is performed by training a machine learning classifier on the synthetic dataset and testing on the real dataset and vice versa. The split data are named real_train, real_test, generated_train, and generated_test. The machine learning classifiers are trained on both real_train and generated_train sets, and the trained models are tested on both the real_test and generated_test sets. The classifiers we used are Decision Trees, Logistic Regression, Multi-Layer Perceptron, and Random Forests. We use the F1-score as a metric to evaluate the classifiers. TSTR is a well-known approach to evaluating the quality of synthetic data. We considered the hyperparameter space specified in [94]. We performed Grid Search and used the best hyperparameter configuration for testing. The Tables 4.6, 4.7, 4.8, 4.9, 4.10, 4.11, show the F-1 metrics of the real and generated data of the classifiers test on all the results we obtained using CTGAN and CopulaGAN.

We achieved close to 100 percent performance for the CAV Policies data generated by applying constraints. Although this result is in consistency with the experimental results discussed in Generative Policy Model approach [9], the main reason to achieving such high performance is due to the fact that the test data is very similar to the training data in both

real and generated datasets. Also, the data is less complex. When we trained the classifiers on the data generated without applying constraints, there is a significant decline in the performance. This is potentially due to the presence of so many violations in the data captured by the trained model. The Amazon Access Logs provides a different perspective. As discussed in the previous section, all features in the dataset are of type categorical. We did not identify a significant difference in the F-1 scores between the models trained on the data generated by applying constraints and without constraints. A primary reason could be the preprocessing performed before training the GAN models where all features were treated as categorical. This also resulted in a difference in correlation as shown in figures 4.5 and 4.6.

Table 4.6: CAV Policies with Constraints - ML Efficacy for Dataset Size 20

	CTGAN		CopulaGAN	
Classifiers	F1_Real	F1_Generated	F1_Real	F1_Generated
DecisionTree_generated	0.9980	1.000	0.9959	0.999
DecisionTree_Real	1.0000	1.000	1.000	1.000
LogisticRegression_generated	0.9612	0.9706	0.9597	0.9670
LogisticRegression_Real	0.9140	0.9138	0.9126	0.9036
MLP_generated	0.9960	1.0000	0.9947	1.000
MLP_Real	1.000	1.000	1.000	1.000
RandomForest_generated	0.9960	0.9987	0.9943	0.9990
RandomForest_Real	1.000	0.9995	0.9998	0.9998

Table 4.7: CAV Policies with Constraints - ML Efficacy for Dataset size 40

	CTGAN		CopulaGAN	
Classifiers	F1_Real	F1_Generated	F1_Real	F1_Generated
DecisionTree_generated	0.9961	1.000	0.9989	1.000
DecisionTree_Real	1.000	1.000	1.000	0.9991
LogisticRegression_generated	0.9506	0.9547	0.9516	0.9716
LogisticRegression_Real	0.9095	0.8951	0.9119	0.9024
MLP_generated	0.9981	1.000	0.9970	1.000
MLP_Real	1.000	1.000	1.000	1.000
RandomForest_generated	0.9943	1.000	0.9993	0.9999
RandomForest_Real	0.999	0.999	1.000	0.9991

Table 4.8: CAV Policies without Constraints - ML Efficacy for Dataset Size 20

	CTGAN		CopulaGAN	
Classifiers	F1_Real	F1_Generated	F1_Real	F1_Generated
DecisionTree_generated	0.8166	0.7732	0.8176	0.8294
DecisionTree_Real	1.0000	0.8039	1.000	0.7729
LogisticRegression_generated	0.8164	0.8386	0.8205	0.8754
LogisticRegression_Real	0.9124	0.8850	0.9139	0.8135
MLP_generated	0.8166	0.8395	0.8166	0.8759
MLP_Real	1.000	0.8910	1.000	0.8178
RandomForest_generated	0.8170	0.8138	0.8179	0.8664
RandomForest_Real	1.000	0.8577	1.000	0.7970

Table 4.9: CAV Policies without Constraints - ML Efficacy for Dataset Size 40

	CTGAN		CopulaGAN	
Classifiers	F1_Real	F1_Generated	F1_Real	F1_Generated
DecisionTree_generated	0.8153	0.7521	0.8186	0.7627
DecisionTree_Real	1.0000	0.7804	1.000	0.8078
LogisticRegression_generated	0.8142	0.8184	0.8038	0.8144
LogisticRegression_Real	0.9105	0.8927	0.9091	0.9012
MLP_generated	0.8172	0.8234	0.8197	0.8230
MLP_Real	1.000	0.9138	1.000	0.9299
RandomForest_generated	0.8161	0.7966	0.8208	0.8035
RandomForest_Real	0.9999	0.8663	1.000	0.8778

Table 4.10: Amazon Access Logs with Constraints

	CTGAN		CopulaGAN	
Classifiers	F1_Real	F1_Generated	F1_Real	F1_Generated
DecisionTree_generated	0.8355	0.8398	0.8578	0.8808
DecisionTree_Real	0.9319	0.8500	0.9281	0.8670
LogisticRegression_generated	0.9167	0.9167	0.9405	0.9405
LogisticRegression_Real	0.9446	0.9446	0.9451	0.9451
MLP_generated	0.9519	0.9162	0.9405	0.9405
MLP_Real	0.9448	0.9446	0.9452	0.9452
RandomForest_generated	0.9132	0.9095	0.9345	0.9364
RandomForest_Real	0.9480	0.9371	0.9484	0.9402

Table 4.11: Amazon Access Logs without Constraints

	CTGAN		CopulaGAN	
Classifiers	F1_Real	F1_Generated	F1_Real	F1_Generated
DecisionTree_generated	0.8656	0.8785	0.8012	0.7977
DecisionTree_Real	0.9323	0.8882	0.9310	0.8305
LogisticRegression_generated	0.9420	0.9420	0.8683	0.8682
LogisticRegression_Real	0.9397	0.9397	0.9439	0.9437
MLP_generated	0.9368	0.9164	0.8374	0.8460
MLP_Real	0.9391	0.9271	0.9220	0.9127
RandomForest_generated	0.9387	0.9396	0.8647	0.8602
RandomForest_Real	0.9458	0.9379	0.9500	0.9081

From our analysis, we identified both CTGAN and CopulaGAN were able to learn the data distribution and generate the synthetic data appropriately, for the CAV policies data. For the Amazon Access Logs, we were able to identify the difference between the real and generated data in in terms of correlation. The size of the training dataset has less impact on synthetic data generation, but the nature of training data has more impact on the generated data. The supervised machine learning classifiers trained using the synthetic data has reasonable performance on the classification tasks. The GAN models obtained almost perfect scores that the almost all generated data are new, and and unseen from the real data.

Chapter 5

Discussion

In the previous Chapter, we analyzed the results obtained by training CTGAN and CopulaGAN on two datasets, namely CAV Policies and Amazon Access Logs. Our results show that Generative Adversarial Networks can effectively generate access control policies for unforeseen contexts. In CAV policies, an unforeseen scenario is a request made by the human driver to the CAV to perform an action, for a specific driving task, under an environmental condition that the CAV did not encounter before. The environmental condition is defined by the weighted average of weather visibility, and traffic congestion. Likewise, in the Amazon Access Logs, an unforeseen scenario could be a request made by a user to access a specific resource that the access control system has never encountered before, or a request made new user to access a resource for which a policy does not exist. Our results achieved an almost perfect similarity score for all experiments using the CAV policies dataset, and a perfect score for the Amazon Access Dataset. As explained in our analysis, achieving a near perfect similarity score means almost all records in the generated data are different from the real data. Given the nature of the data, these new scenarios are dependent on the environmental_weighted_average attribute in CAV policies. However, for the Amazon Access Logs, the generated new policies are all several combinations of existing categories in the real dataset.

Our experiments and results answer RQ2. Research on policy generation in access control is being done for more than a decade. However, the use of GANs as a method

to automatically generate policies is unexplored. This method can potentially be deployed in various information systems for policy generation tasks. [9], [26], [95] discuss the generative policy architecture. However, the architecture require methods by which parties can generate and evolve their own policies. The National Institute of Standards and Technology, in its report on machine learning based access control policy verification mentioned, policy rules in some systems are automatically generated from previous access logs or by intelligent mechanisms [96]. [97], in their dissertation stresses the need for an automated method for ABAC policy data extraction. Automated policy generation is essential, especially in heterogeneous systems such as IoT.

We also demonstrated the importance of constraining the models to learn the background knowledge for the generated data to be in compliance with them. We discussed constrained generative models in Chapter 3. Policy generation approaches with learned constraints could prove beneficial and be a part of the defense in-depth approach in many information systems, especially IoT. A defense in-depth approach is a layered approach where an adversary should encounter several layers of security before exploiting an information system. Our experiments show that adding constraints during the training process enable the models to generate data without any violations. This answers our RQ3. Violations are nothing but over-permissions. Over-provisioning access poses a major risk to any information system. This may also lead to insider threats. Let us consider two real-time incidents. In the case of [98], a former employee of the company was able to use his credentials to access the personal information of customers, such as brokerage accounts, etc. In general, once an employee exits an organization, access to all systems pertaining to the employee should be revoked and the credentials should be disabled. In some scenarios, this could be overlooked. [99] says 1 in 4 employees still have access to data at their previous job. Machine Learning based access control approaches, including generative models are being adopted today in information systems security. Hence, constraining these models with business rules will add an additional layer of security.

5.1 Limitations

Our research comes with the following limitations:

- An important limitation we consider in this study is the data. As we mentioned in Chapter 2, there are not many publicly available labeled datasets for access control research in IoT. The CAV policies dataset we used in our research is well-balanced, containing an almost equal number of Approved and Rejected decisions. The number of categories (discrete features) present in the dataset is smaller. However, the authors mentioned a potential future direction for identifying more scenarios for further research. On the other hand, the Amazon Access Logs dataset has all categorical features and is highly imbalanced. Both datasets contain about ten features. The availability of more features potentially leads to more robust and expressive policies.

- Another limitation we posit is the constraints that we adopted in our experiments. All constraints we discussed in our experiments are statically coded and the models generated data by learning only those constraints. As mentioned in Chapter 2, IoT networks, in real-time, are dynamic in nature. Hence, the constraints within the network may change over time. Incorporating additional constraints requires retraining the model. Also, applying constraints for the Amazon dataset is also a challenge due to limited background information regarding the features.

- Limited work is performed using generative models for access control policy generation. The closest works we could compare our approach to are the generative policy model proposed in [9] and ABAC data generation using CTGAN [93]. The authors proposed a web-based tool where we can download a synthetic ABAC dataset by specifying the desired number of rows.

Chapter 6

Conclusion

In this chapter, we discuss an overview of the research, contributions, and scope for future work.

6.1 Research Summary

The purpose of this research is to implement and evaluate the efficacy of Generative Adversarial Networks (GANs) to generate contextual and usable access control policies during unforeseen scenarios. We formulated our research as a synthetic data generation problem and utilized mixed methods to evaluate the implemented models.

Our research study answered the following three questions.

RQ1: What are the existing challenges while specifying dynamic policies for access control in IoT?

To answer this question, we performed a literature review and arrived at the following findings:

- Centralized architecture is inefficient for a heterogeneous environment like IoT. Hence, it is recommended to adopt technologies such as edge computing that follows distributed architecture.

- The scope for research exists in the area of IoT for policy generation, decision, evaluation at runtime. Using balanced dataset, and the appropriate selection of

features for machine learning are among the key challenges in access control research

- Machine learning classifiers trained on the access control policy data for decision making should be as accurate as possible. Any misclassification constitutes either policy violation, or over-privileges.

RQ2: How can we self-generate contextual access control policies for the internet of things during unforeseen situations?

We considered the findings from RQ1 to formulate our research study. We proposed to utilize GANs to generate access control policies. We chose two baseline models, namely CTGAN and CopulaGAN for experimentation. We chose two datasets namely the CAV policies dataset and the Amazon Access Logs dataset. We followed a mixed methods approach to quantitatively evaluate the implemented models and evaluate the quality of generated policies using manual evaluation. We utilized 20 percent and 40 percent of the CAV policies for our experimentation. We utilized the Amazon Access Logs as is. The results demonstrated that the GAN models were to generate data that is similar to the distribution of the real data.

RQ3: How realistic are the generated access control policies to be used in real-time situations?

Our initial results identified a significant number of policy violations. Hence, we trained the models by applying constraints on both datasets. Our results demonstrate that the models were able to generate policies without any violation of the specified constraints. However, although the Amazon Access Logs generated data is in compliance with the constraints, the generated data distribution is not similar to that of the real data. As we discussed earlier, this could be due to the presence of only categorical features in the dataset.

Our results answer RQ2 and RQ3, stating that GANs shall be used to generate contextual and usable access control policies for IoT. GANs can learn constraints applied to the model and generate policies that comply with the constraints.

6.2 Research Contributions

Our research contributes to the field of access control for the Internet of Things. The primary objective of research is to dynamically generate access control policies for IoT. Our research suggests that GANs can be used for automated policy generation in access control for IoT. GAN models shall be deployed in various architectures including distributed and federated architectures which are most suitable for IoT networks. Next, our research demonstrates the importance of the background knowledge of the domain the models will be trained and deployed. Training with constraints is very essential for a model to generate policies without any violations. Existing studies related to GANs in cybersecurity did not discuss training the models with constraints. Hence, our research provides a novel perspective to generating constrained access control policies, and stresses the need for further research in this area.

6.3 Future Work

Our research study has provided scoping for further research in the following areas:

- Addition of more features. As we described earlier, both the datasets we utilized for our experiments contained less features. Adding new features paves the way for generating more expressive access control policies.

- Zero-Trust Architecture (ZTA). ZTA is widely adopted in organizational information systems today. Future research can focus on how automated access control policy generation can be utilized within the context of ZTA.

- Other GAN models. It will be another research direction to experiment other GAN models like Wasserstein GAN (WGAN), WGAN with gradient penalty in conjunction with how these models can ensure the privacy of the generated policies.

- Dynamic Constraints. Although the policies were generated dynamically, the constraints were manually coded before training the model. IoT networks are dynamic in nature and it is possible that the background constraints may change over time. Currently the only way to adapt to the constraints is to retrain the model. Future research can focus on adaptability of the models to varying constraints in the environment.

References

[1] Statista Research Department, *Internet of Things (IoT) Connected Devices Installed Base Worldwide from 2015 to 2025*, 2016. [Online]. Available: https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/.

[2] T. Macaulay, "Chapter 9 - Identity and Access Control Requirements in the IoT," in T. Macaulay, Ed. Boston: Morgan Kaufmann, 2017, pp. 157–176, ISBN: 978-0-12-419971-2. DOI: https://doi.org/10.1016/B978-0-12-419971-2.00009-1. [Online]. Available: http://www.sciencedirect.com/science/article/pii/B9780124199712000091.

[3] A. Sharma, S. Sharma, and M. Dave, "Identity and Access management- A Comprehensive Study," in *2015 International Conference on Green Computing and Internet of Things (ICGCIoT)*, 2015, pp. 1481–1485. DOI: 10.1109/ICGCIoT.2015.7380701.

[4] Y. Wang, G. Attebury, and B. Ramamurthy, "A Survey of Security Issues in Wireless Sensor Networks," *IEEE Communications Surveys Tutorials*, vol. 8, no. 2, pp. 2–23, 2006. DOI: 10.1109/COMST.2006.315852.

[5] S. Hilton, *Dyn Analysis Summary Of Friday October 21 Attack*, 2016. [Online]. Available: http://dyn.com/blog/dyn-analysis-summary-of-friday-october-21-attack/.

[6] Check Point Research, *IoTroop Botnet: The Full Investigation*, 2017. [Online]. Available: https://research.checkpoint.com/iotroop-botnet-full-investigation/.

[7] C. Kolias, G. Kambourakis, A. Stavrou, and J. Voas, "DDoS in the IoT: Mirai and Other Botnets," *Computer*, vol. 50, no. 7, pp. 80–84, 2017. DOI: 10.1109/MC.2017.201.

[8] S. Calo, D. Verma, S. Chakraborty, E. Bertino, E. Lupu, and G. Cirincione, "Self-generation of Access Control Policies," in *Proceedings of the 23nd ACM on Symposium on Access Control Models and Technologies*, 2018, pp. 39–47.

[9] D. Cunnington, I. Manotas, M. Law, *et al.*, "A Generative Policy Model for Connected and Autonomous Vehicles," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, 2019, pp. 1558–1565. DOI: 10.1109/ITSC.2019.8916782.

[10] L. Abuserrieh and M. H. Alalfi, *A survey of analysis methods for security and safety verification in iot systems*, 2022. DOI: 10.48550/ARXIV.2203.01464. [Online]. Available: https://arxiv.org/abs/2203.01464.

[11] E. Bertino, A. A. Jabal, S. Calo, D. Verma, and C. Williams, "The challenge of access control policies quality," *J. Data and Information Quality*, vol. 10, no. 2, 2018, ISSN: 1936-1955. DOI: 10.1145/3209668. [Online]. Available: https://doi.org/10.1145/3209668.

[12] A. Ouaddah, H. Mousannif, A. A. Elkalam, and A. A. Ouahman, "Access control in the Internet of Things: Big challenges and new opportunities," *Computer Networks*, vol. 112, pp. 237–262, 2017. DOI: https://doi.org/10.1016/j.comnet.2016.11.007. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1389128616303735.

[13] S. Ravidas, A. Lekidis, F. Paci, and N. Zannone, "Access Control in Internet-of-Things: A Survey," *Journal of Network and Computer Applications*, vol. 144, pp. 79–101, 2019. DOI: https://doi.org/10.1016/j.jnca.2019.06.017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S108480451930222X.

[14] E. Bertin, D. Hussein, C. Sengul, and V. Frey, "Access Control in the Internet of Things: A Survey of Existing Approaches and Open Research Questions," *Annals of Telecommunications*, vol. 74, no. 7-8, pp. 375–388, 2019. DOI: https://doi.org/10.1007/s12243-019-00709-7.

[15] M. Nur and Y. Wang, "Identity Relationship Management for Internet of Things: A Case Study," in *2022 IEEE International Conference on Consumer Electronics (ICCE)*, 2022.

[16] S. Bhattarai and Y. Wang, "End-to-End Trust and Security for Internet of Things Applications," *Computer*, vol. 51, no. 04, pp. 20–27, 2018, ISSN: 1558-0814. DOI: 10.1109/MC.2018.2141038.

[17] S. Pal, "Limitations and Approaches in Access Control and Identity Management for Constrained IoT Resources," in *2019 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, 2019, pp. 431–432. DOI: 10.1109/PERCOMW.2019.8730651.

[18] K. Cheung, M. Huth, L. Kirk, L.-N. Lundbæk, R. Marques, and J. Petsche, "Owner-Centric Sharing of Physical Resources, Data, and Data-Driven Insights in Digital Ecosystems," in *Proceedings of the 24th ACM Symposium on Access Control Models and Technologies*, ser. SACMAT '19, Toronto ON, Canada: Association for Computing Machinery, 2019, 73–81, ISBN: 9781450367530. DOI: 10.1145/3322431.3326326. [Online]. Available: https://doi.org/10.1145/3322431.3326326.

[19] Y. Al-Halabi, N. Raeq, and F. Abu-Dabaseh, "Study on Access Control Approaches in the Context of Internet of Things: A Survey," in *2017 International Conference on*

Engineering and Technology (ICET), 2017, pp. 1–7. DOI: 10.1109/ICEngTechnol.2017.8308153.

[20] M. Nguyen, M. O. Gani, and V. Raychoudhury, "Yours Truly? Survey on Accessibility of Our Personal Data in the Connected World," in *2019 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, 2019, pp. 292–297. DOI: 10.1109/PERCOMW.2019.8730880.

[21] I. Ray, R. Abdunabi, and R. Basnet, "Access Control for Internet of Things Applications," in *Proceedings of the 5th on Cyber-Physical System Security Workshop*, ser. CPSS '19, Association for Computing Machinery, 2019, pp. 35–36, ISBN: 9781450367875. DOI: 10.1145/3327961.3329533. [Online]. Available: https://doi.org/10.1145/3327961.3329533.

[22] A. Kaur, Isha, G. Rai, and A. Malik, "Authentication and Context Awareness Access Control in Internet of Things: A Review," in *2018 8th International Conference on Cloud Computing, Data Science Engineering (Confluence)*, 2018, pp. 630–635. DOI: 10.1109/CONFLUENCE.2018.8443067.

[23] A. Alkhresheh, K. Elgazzar, and H. S. Hassanein, "Adaptive Access Control Policies for IoT Deployments," in *2020 International Wireless Communications and Mobile Computing (IWCMC)*, IEEE, 2020, pp. 377–383. DOI: 10.1109/IWCMC48107.2020.9148090.

[24] A. Alkhresheh, K. Elgazzar, and H. S. Hassanein, "DACIoT: Dynamic Access Control Framework for IoT Deployments," *IEEE Internet of Things Journal*, vol. 7, no. 12, pp. 11 401–11 419, 2020. DOI: 10.1109/JIOT.2020.3002709.

[25] A. A. El Kalam, A. Outchakoucht, and H. Es-Samaali, "Emergence-Based Access Control: New Approach to Secure the Internet of Things," in *Proceedings of the 1st International Conference on Digital Tools & Uses Congress*, ser. DTUC '18, Paris, France: Association for Computing Machinery, 2018, ISBN: 9781450364515. DOI: 10.1145/3240117.3240136. [Online]. Available: https://doi.org/10.1145/3240117.3240136.

[26] E. Bertino, A. Russo, M. Law, *et al.*, "Generative Policies for Coalition Systems-A Symbolic Learning Framework," in *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, IEEE, 2019, pp. 1590–1600. DOI: 10.1109/ICDCS.2019.00158.

[27] W. Denniss, J.Bradley, M. Jones, and H. Tschofenig, *RFC8628: OAuth 2.0 Device Authorization Grant*, 2019. [Online]. Available: https://datatracker.ietf.org/doc/html/rfc8628.

[28] Z. Shelby, K. Hartke, and C. Bormann, *RFC7252: The Constrained Application Protocol (CoAP)*, 2014. [Online]. Available: https://datatracker.ietf.org/doc/html/rfc7252.

[29] J. Dizdarević, F. Carpio, A. Jukan, and X. Masip-Bruin, "A Survey of Communication Protocols for Internet of Things and Related Challenges of Fog and Cloud Computing Integration," *ACM Comput. Surv.*, vol. 51, no. 6, 2019, ISSN: 0360-0300. DOI: 10.1145/3292674. [Online]. Available: https://doi.org/10.1145/3292674.

[30] B. Mishra and A. Kertesz, "The Use of MQTT in M2M and IoT Systems: A Survey," *IEEE Access*, vol. 8, pp. 201 071–201 086, 2020. DOI: 10.1109/ACCESS.2020.3035849.

[31] M. A. Prada, P. Reguera, S. Alonso, A. Morán, J. J. Fuertes, and M. Domínguez, "Communication with Resource-Constrained Devices through MQTT for Control Education," *IFAC-PapersOnLine*, vol. 49, no. 6, pp. 150–155, 2016, 11th IFAC Symposium on Advances in Control Education ACE 2016, ISSN: 2405-8963. DOI: https://doi.org/10.1016/j.ifacol.2016.07.169. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2405896316303743.

[32] S. Wagle, "Semantic Data Extraction over MQTT for IoTcentric Wireless Sensor Networks," in *2016 International Conference on Internet of Things and Applications (IOTA)*, 2016, pp. 227–232. DOI: 10.1109/IOTA.2016.7562727.

[33] M. Machulak and J. Richer, *User-Managed Access (UMA) 2.0 Grant for OAuth 2.0 Authorization*, 2018. [Online]. Available: https://docs.kantarainitiative.org/uma/wg/rec-oauth-uma-grant-2.0.html.

[34] X. Su, J. Hyysalo, M. Rautiainen, *et al.*, "Privacy as a Service: Protecting the Individual in Healthcare Data Processing," *Computer*, vol. 49, no. 11, pp. 49–59, 2016. DOI: 10.1109/MC.2016.337.

[35] C.-A. Lin and C.-F. Liao, "User-Managed Access Delegation for Blockchain-driven IoT Services," in *2020 International Computer Symposium (ICS)*, 2020, pp. 462–467. DOI: 10.1109/ICS51289.2020.00097.

[36] V. A. Siris, D. Dimopoulos, N. Fotiou, S. Voulgaris, and G. C. Polyzos, "OAuth 2.0 Meets Blockchain for Authorization in Constrained IoT Environments," in *2019 IEEE 5th World Forum on Internet of Things (WF-IoT)*, 2019, pp. 364–367. DOI: 10.1109/WF-IoT.2019.8767223.

[37] K. Albulayhi, A. Abuhussein, F. Alsubaei, and F. T. Sheldon, "Fine-Grained Access Control in the Era of Cloud Computing: An Analytical Review," in *2020 10th Annual Computing and Communication Workshop and Conference (CCWC)*, 2020, pp. 0748–0755. DOI: 10.1109/CCWC47524.2020.9031179.

[38] T. L. Shan, S. A. Ismail, and A. Azizan, "Access Control Models for Cloud Computing: A Review," in *2018 2nd International Conference on Telematics and Future Generation Networks (TAFGEN)*, 2018, pp. 155–158. DOI: 10.1109/TAFGEN.2018.8580489.

[39] A. Ouaddah, H. Mousannif, A. Abou Elkalam, and A. Ait Ouahman, "Access Control in IoT: Survey & state of the art," in *2016 5th International Conference on Multimedia Computing and Systems (ICMCS)*, 2016, pp. 272–277. DOI: 10.1109/ICMCS.2016.7905662.

[40] D. Servos and S. L. Osborn, "Current Research and Open Problems in Attribute-Based Access Control," *ACM Computing Surveys (CSUR)*, vol. 49, no. 4, pp. 1–45, 2017. DOI: 10.1145/3007204.

[41] A. A. E. Kalam, R. E. Baida, P. Balbiani, *et al.*, "Organization Based Access Control," in *Proceedings POLICY 2003. IEEE 4th International Workshop on Policies for Distributed Systems and Networks*, 2003, pp. 120–131. DOI: 10.1109/POLICY.2003.1206966.

[42] J. Qiu, Z. Tian, C. Du, Q. Zuo, S. Su, and B. Fang, "A Survey on Access Control in the Age of Internet of Things," *IEEE Internet of Things Journal*, vol. 7, no. 6, pp. 4682–4696, 2020. DOI: 10.1109/JIOT.2020.2969326.

[43] S. Gusmeroli, S. Piccione, and D. Rotondi, "A Capability-Based Security Approach to Manage Access Control in the Internet of Things," *Mathematical and Computer Modelling*, vol. 58, no. 5, pp. 1189 –1205, 2013, ISSN: 0895-7177. DOI: https://doi.org/10.1016/j.mcm.2013.02.006. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S089571771300054X.

[44] Y. Andaloussi, M. D. E. Ouadghiri, Y. Maurel, J. M. Bonnin, and H. Chaoui, "Access Control in IoT Environments: Feasible Scenarios," *Procedia Computer Science*, vol. 130, pp. 1031–1036, 2018. DOI: https://doi.org/10.1016/j.procs.2018.04.144. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1877050918305064.

[45] I. Riabi, H. K. B. Ayed, and L. A. Saidane, "A Survey on Blockchain Based Access Control for Internet of Things," in *2019 15th International Wireless Communications Mobile Computing Conference (IWCMC)*, 2019, pp. 502–507. DOI: 10.1109/IWCMC.2019.8766453.

[46] X. Zhu and Y. Badr, "A Survey on Blockchain-Based Identity Management Systems for the Internet of Things," in *2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, 2018, pp. 1568–1573. DOI: 10.1109/Cybermatics_2018.2018.00263.

[47] P. E. Sedgewick and R. de Lemos, "Self-Adaptation Made Easy with Blockchains," in *2018 IEEE/ACM 13th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*, 2018, pp. 192–193.

[48] M. Nur and Y. Wang, "An Overview of Identity Relationship Management in the Internet of Things," in *2021 IEEE International Conference on Consumer Electronics (ICCE)*, 2021, pp. 1–5. DOI: 10.1109/ICCE50685.2021.9427723.

[49] L. Tandon, P. W. L. Fong, and R. Safavi-Naini, "HCAP: A History-Based Capability System for IoT Devices," in *Proceedings of the 23nd ACM on Symposium on Access Control Models and Technologies*, ser. SACMAT '18, Indianapolis, Indiana, USA: Association for Computing Machinery, 2018, 247–258, ISBN: 9781450356664. DOI: 10.1145/3205977.3205978. [Online]. Available: https://doi.org/10.1145/3205977.3205978.

[50] L. Wu, X. Du, M. Guizani, and A. Mohamed, "Access Control Schemes for Implantable Medical Devices: A Survey," *IEEE Internet of Things Journal*, vol. 4, no. 5, pp. 1272–1283, 2017. DOI: 10.1109/JIOT.2017.2708042.

[51] R. Thirukkumaran and P. Muthu Kannan, "Survey: Security and Trust Management in Internet of Things," in *2018 IEEE Global Conference on Wireless Computing and Networking (GCWCN)*, 2018, pp. 131–134. DOI: 10.1109/GCWCN.2018.8668640.

[52] K. Kafle, K. Moran, S. Manandhar, A. Nadkarni, and D. Poshyvanyk, "A Study of Data Store-Based Home Automation," in *Proceedings of the Ninth ACM Conference on Data and Application Security and Privacy*, ser. CODASPY '19, Richardson, Texas, USA: Association for Computing Machinery, 2019, 73–84, ISBN: 9781450360999. DOI: 10.1145/3292006.3300031. [Online]. Available: https://doi.org/10.1145/3292006.3300031.

[53] M. Tabassum, J. Kropczynski, P. Wisniewski, and H. R. Lipford, "Smart Home Beyond the Home: A Case for Community-Based Access Control," in *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, ser. CHI '20, Honolulu, HI, USA: Association for Computing Machinery, 2020, 1–12, ISBN: 9781450367080. DOI: 10.1145/3313831.3376255. [Online]. Available: https://doi.org/10.1145/3313831.3376255.

[54] W. Jang, A. Chhabra, and A. Prasad, "Enabling Multi-User Controls in Smart Home Devices," in *Proceedings of the 2017 Workshop on Internet of Things Security and Privacy*, ser. IoTS&P '17, Dallas, Texas, USA: Association for Computing Machinery, 2017, 49–54, ISBN: 9781450353960. DOI: 10.1145/3139937.3139941. [Online]. Available: https://doi.org/10.1145/3139937.3139941.

[55] J. Koh, D. Hong, S. Nagare, S. Boovaraghavan, Y. Agarwal, and R. Gupta, "Who Can Access What, and When? Understanding Minimal Access Requirements of

Building Applications," in *Proceedings of the 6th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*, ser. BuildSys '19, New York, NY, USA: Association for Computing Machinery, 2019, 121–124, ISBN: 9781450370059. DOI: 10.1145/3360322.3360868. [Online]. Available: https://doi.org/10.1145/3360322.3360868.

[56] Q. Liu, H. Zhang, J. Wan, and X. Chen, "An access control model for resource sharing based on the role-based access control intended for multi-domain manufacturing internet of things," *IEEE Access*, vol. 5, pp. 7001–7011, 2017. DOI: 10.1109/ACCESS.2017.2693380.

[57] A. Gabillon, R. Gallier, and E. Bruno, "Access Controls for IoT Networks," *SN Computer Science*, vol. 1, no. 1, pp. 1–13, 2020. DOI: 10.1007/s42979-019-0022-z.

[58] K. Riad and J. Cheng, "Adaptive XACML Access Policies for Heterogeneous Distributed IoT Environments," *Information Sciences*, vol. 548, pp. 135–152, 2021, ISSN: 0020-0255. DOI: https://doi.org/10.1016/j.ins.2020.09.051. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0020025520309634.

[59] S. Sicari, A. Rizzardi, L. Grieco, G. Piro, and A. Coen-Porisini, "A Policy Enforcement Framework for Internet of Things Applications in the Smart Health," *Smart Health*, vol. 3-4, pp. 39–74, 2017, ISSN: 2352-6483. DOI: https://doi.org/10.1016/j.smhl.2017.06.001. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352648316300435.

[60] Q Zhang, H Zhong, J Cui, L Ren, and W Shi, "AC4AV: A Flexible and Dynamic Access Control Framework for Connected and Autonomous Vehicles," *IEEE Internet of Things Journal*, vol. 8, no. 3, pp. 1946–1958, 2021. DOI: 10.1109/JIOT.2020.3016961.

[61] Y. Liu, M. Xiao, Y. Zhou, *et al.*, "An Access Control Mechanism Based on Risk Prediction for the IoV," in *2020 IEEE 91st Vehicular Technology Conference (VTC2020-Spring)*, 2020, pp. 1–5. DOI: 10.1109/VTC2020-Spring48590.2020.9129056.

[62] T. Yu, T. Li, Y. Sun, *et al.*, "Learning Context-Aware Policies from Multiple Smart Homes via Federated Multi-Task Learning," in *2020 IEEE/ACM Fifth International Conference on Internet-of-Things Design and Implementation (IoTDI)*, 2020, pp. 104–115. DOI: 10.1109/IoTDI49375.2020.00017.

[63] M. Chu, H. Li, X. Liao, and S. Cui, "Reinforcement Learning-Based Multiaccess Control and Battery Prediction With Energy Harvesting in IoT Systems," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 2009–2020, 2019, ISSN: 2372-2541. DOI: 10.1109/JIOT.2018.2872440.

[64] D. Di Francesco Maesa, P. Mori, and L. Ricci, "A Blockchain Based Approach for the Definition of Auditable Access Control Systems," *Computers & Security*, vol. 84, pp. 93–119, 2019, ISSN: 0167-4048. DOI: https://doi.org/10.1016/j.cose.2019.03.016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167404818309398.

[65] Y. Liu, J. Zhang, and J. Zhan, "Privacy Protection for Fog Computing and the Internet of Things Data Based on Blockchain," *Cluster Computing*, vol. 24, no. 2, pp. 1331–1345, 2021. DOI: https://doi.org/10.1007/s10586-020-03190-3.

[66] C. Alcaraz, J. E. Rubio, and J. Lopez, "Blockchain-assisted Access for Federated Smart Grid Domains: Coupling and Features," *Journal of Parallel and Distributed Computing*, vol. 144, pp. 124–135, 2020, ISSN: 0743-7315. DOI: https://doi.org/10.1016/j.jpdc.2020.05.012. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0743731520302914.

[67] Y. Zhang, M. Yutaka, M. Sasabe, and S. Kasahara, "Attribute-Based Access Control for Smart Cities: A Smart-Contract-Driven Framework," *IEEE Internet of Things Journal*, vol. 8, no. 8, pp. 6372–6384, 2021. DOI: 10.1109/JIOT.2020.3033434.

[68] M. Mont, S. Pearson, and P. Bramhall, "Towards Accountable Management of Identity and Privacy: Sticky Policies and Enforceable Tracing Services," in *14th International Workshop on Database and Expert Systems Applications, 2003. Proceedings.*, 2003, pp. 377–382. DOI: 10.1109/DEXA.2003.1232051.

[69] J. A. Padget and W. W. Vasconcelos, "Fine-Grained Access Control via Policy-Carrying Data," *ACM Trans. Internet Technol.*, vol. 18, no. 3, Feb. 2018, ISSN: 1533-5399. DOI: 10.1145/3133324. [Online]. Available: https://doi.org/10.1145/3133324.

[70] S. Sicari, A. Rizzardi, D. Miorandi, and A. Coen-Porisini, "Dynamic Policies in Internet of Things: Enforcement and Synchronization," *IEEE Internet of Things Journal*, vol. 4, no. 6, pp. 2228–2238, 2017. DOI: 10.1109/JIOT.2017.2749604.

[71] G. Sagirlar, B. Carminati, and E. Ferrari, "Decentralizing Privacy Enforcement for Internet of Things Smart Objects," *Computer Networks*, vol. 143, pp. 112–125, 2018, ISSN: 1389-1286. DOI: https://doi.org/10.1016/j.comnet.2018.07.019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1389128618305322.

[72] S. Sicari, A. Rizzardi, D. Miorandi, and A. Coen-Porisini, "Security Towards the Edge: Sticky Policy Enforcement for Networked Smart Objects," *Information Systems*, vol. 71, pp. 78–89, 2017, ISSN: 0306-4379. DOI: https://doi.org/10.1016/j.is.2017.07.006. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0306437917303770.

[73] D. Miorandi, A. Rizzardi, S. Sicari, and A. Coen-Porisini, "Sticky Policies: A Survey," *IEEE Transactions on Knowledge and Data Engineering*, vol. 32, no. 12, pp. 2481–2499, 2020. DOI: 10.1109/TKDE.2019.2936353.

[74] B. P. Rimal, M. Maier, and M. Satyanarayanan, "Experimental Testbed for Edge Computing in Fiber-Wireless Broadband Access Networks," *IEEE Communications Magazine*, vol. 56, no. 8, pp. 160–167, 2018.

[75] N. B. Akhuseyinoglu and J. Joshi, "Access Control Approaches for Smart Cities," *IoT Technologies in Smart Cities: From sensors to big data, security and trust*, pp. 1–40, Mar. 2020. DOI: 10.1049/PBCE128E_ch1. [Online]. Available: https://digital-library.theiet.org/content/books/10.1049/pbce128e_ch1.

[76] A. Outchakoucht, E. S. Hamza, and J. P. Leroy, "Dynamic Access Control Policy Based on Blockchain and Machine Learning for the internet of Things," *Int. J. Adv. Comput. Sci. Appl*, vol. 8, no. 7, pp. 417–424, 2017. DOI: 10.14569/IJACSA.2017.080757.

[77] A. Outchakoucht, A. Abou El Kalam, H. Es-Samaali, and S. Benhadou, "Machine Learning based Access Control Framework for the Internet of Things," *International Journal of Advanced Computer Science and Applications*, vol. 11, no. 2, 2020. DOI: 10.14569/IJACSA.2020.0110243.

[78] S. Wachter, "Normative Challenges of Identification in the Internet of Things: Privacy, Profiling, Discrimination, and the GDPR," *Computer Law & Security Review*, vol. 34, no. 3, pp. 436–449, 2018. DOI: https://doi.org/10.1016/j.clsr.2018.02.002. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0267364917303904.

[79] N. W. Hardy, "The Internet of Things Ecosystem: Survey of the Current Landscape, Identity Relationship Management, Multifactor Authentication Mechanisms, and Underlying Protocols," *International Journal of Computer and Information Engineering*, vol. 10, no. 6, pp. 1202–1206, 2016. DOI: https://doi.org/10.5281/zenodo.1125575.

[80] S. Calo, D. Verma, S. Chakraborty, E. Bertino, E. Lupu, and G. Cirincione, "Self-Generation of Access Control Policies," in *Proceedings of the 23nd ACM on Symposium on Access Control Models and Technologies*, ser. SACMAT '18, Indianapolis, Indiana, USA: Association for Computing Machinery, 2018, 39–47, ISBN: 9781450356664. DOI: 10.1145/3205977.3205995. [Online]. Available: https://doi.org/10.1145/3205977.3205995.

[81] K. N. Muthusamy Ragothaman and Y. Wang, "A Systematic Mapping Study of Access Control in the Internet of Things," in *Proceedings of the 54th Hawaii Inter-*

national Conference on System Sciences, 2021, pp. 7090–7099. DOI: hdl.handle.net/10125/71474.

[82] Y. Wang and J. Nikolai, "Key Management in CPSs," pp. 117–136, 2017. DOI: https://doi.org/10.1002/9781119226079.ch6. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/9781119226079.ch6. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/9781119226079.ch6.

[83] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, *et al.*, "Generative Adversarial Nets," in *Advances in Neural Information Processing Systems*, vol. 3, 2014, pp. 2672–2680. DOI: 10.3156/jsoft.29.5_177_2. [Online]. Available: http://www.github.com/goodfeli/adversarial.

[84] L. Xu and K. Veeramachaneni, *Synthesizing tabular data using generative adversarial networks*, 2018. arXiv: 1811.11264 [cs.LG].

[85] L. Xu, M. Skoularidou, A. Cuesta-Infante, and K. Veeramachaneni, "Modeling tabular data using conditional gan," in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds., vol. 32, Curran Associates, Inc., 2019. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2019/file/254ed7d2de3b23ab10936522dd547b
Paper.pdf.

[86] Synthetic Data Vault, *CopulaGANSynthesizer*. [Online]. Available: https://docs.sdv.dev/sdv/single-table-data/modeling/synthesizers/copulagansynthesizer.

[87] S. Kamthe, S. Assefa, and M. Deisenroth, *Copula flows for synthetic data generation*, 2021. arXiv: 2101.00598 [stat.ML].

[88] M. C. Stoian, S. Dyrmishi, M. Cordy, T. Lukasiewicz, and E. Giunchiglia, *How realistic is your synthetic data? constraining deep generative models for tabular data*, 2024. arXiv: 2402.04823 [cs.LG].

[89] Kaggle, *Amazon Employee Access Challenge*. [Online]. Available: https://www.kaggle.com/c/amazon-employee-access-challenge/.

[90] B. Brenninkmeijer, A de Vries, E Marchiori, and Y. Hille, "On the generation and evaluation of tabular data using gans," *Doctoral dissertation*, 2019.

[91] J. Yoon, J. Jordon, and M. van der Schaar, "PATE-GAN: Generating synthetic data with differential privacy guarantees," in *International Conference on Learning Representations*, 2019. [Online]. Available: https://openreview.net/forum?id=S1zk9iRqF7.

[92] C. Esteban, S. L. Hyland, and G. Rätsch, *Real-valued (medical) time series generation with recurrent conditional gans*, 2017. arXiv: 1706.02633 [stat.ML].

[93] R. Rai and S. Sural, "Tool/dataset paper: Realistic abac data generation using conditional tabular gan," in *Proceedings of the Thirteenth ACM Conference on Data and Application Security and Privacy*, ser. CODASPY '23, Charlotte, NC, USA: Association for Computing Machinery, 2023, 273–278, ISBN: 9798400700675. DOI: 10.1145/3577923.3583635. [Online]. Available: https://doi.org/10.1145/3577923.3583635.

[94] J. Kim, C. Lee, and N. Park, *Stasy: Score-based tabular data synthesis*, 2023. arXiv: 2210.04018 [cs.LG].

[95] S. Calo, I. Manotas, G. de Mel, *et al.*, "Agenp: An Asgrammar-Based Generative Policy Framework," in *Policy-Based Autonomic Data Governance*, Springer, 2019, pp. 3–20.

[96] NIST, *Machine Learning for Access Control Policy Verification*, 2024. [Online]. Available: https://doi.org/10.6028/NIST.IR.8360.

[97] L. Karimi, "Towards automatic attribute-based access control policy design and management for highly dynamic environments," Ph.D. dissertation, University of Pittsburgh, 2022. [Online]. Available: http://d-scholarship.pitt.edu/43487/13/Leila_Karimi_Dissertation_08-19-2022.pdf.

[98] Mendoza, Jordan, *Over 8 million Cash App users possibly affected by data breach from a former employee*, 2022. [Online]. Available: https://www.usatoday.com/story/money/2022/04/06/cash-app-data-breach/9490327002/.

[99] Pickard-Whitehead, Gabrielle, *1 in 4 Former Employees Still Has Access to Files at Old Job*, 2021. [Online]. Available: https://smallbiztrends.com/employees-access-files-former-employer/.

Appendix A

Publications from this Dissertation

1. Muthusamy Ragothaman, Kaushik Nagarajan, Yong Wang, Bhaskar Rimal, and Mark Lawrence. 2023. "Access Control for IoT: A Survey of Existing Research, Dynamic Policies and Future Directions" Sensors 23, no. 4: 1805. https://doi.org/10.3390/s23041805.

2. Muthusamy Ragothaman, Kaushik Nagarajan, Yong Wang, and Srinivasulu Vuggumudi, "Towards Automated Policy Generation for Dynamic Access Control in the Internet of Things" (2021). AMCIS 2021 Proceedings. 25. https://aisel.aisnet.org/amcis2021/info_security/info_security/25.

3. Muthusamy Ragothaman, Kaushik Nagarajan, and Yong Wang. "A Systematic Mapping Study of Access Control in the Internet of Things." Proceedings of the 54th Hawaii International Conference on System Sciences. 2021. http://hdl.handle.net/10125/71474.