SQuBA: Social Quorum Based Access Control for Open IoT Environments

Yixuan Wang
University of Minnesota, Twin Cities
yixua003@umn.edu

Abhishek Chandra
University of Minnesota, Twin Cities
chandra@umn.edu

Jon Weissman
University of Minnesota, Twin Cities
jon@cs.umn.edu

***Abstract*—Internet of things (IoT) devices have been ubiquitous in recent years. An emerging model for IoT deployment is an open edge-based infrastructure. Edge resources are commonly used to coordinate capabilities and manage access due to IoT device resource limitations and IoT vendor heterogeneity. The open IoT environment often exists in a multi-user setting, where multiple users interact with a single IoT device. In this setting, we assume that none of the users or the edges are fully trusted, thus IoT data privacy may be compromised. Limited attention has been paid to authorization and auditing in this environment. However, exploiting inter-user relationships gives us leverage. In this work, we propose a social quorum based architecture, SQuBA, as an access control mechanism for IoT which provides relationship-driven authorization and auditing. We present a tiered approach to support access control rules and relationship-based trustworthiness. We implemented a prototype and carried out experiments using a real-world dataset under various scenarios and configurations. The results demonstrate both SQuBA's promising near real-time response latency that is in the order of milliseconds, and good resilience to different edge faulty models. We also compare with various baselines and SQuBA is able to improve end-to-end latency by up to 10X and tolerate the number of faulty edges by up to 2X.**

***Index Terms*—Edge Computing, Internet of Things, Privacy, Access Control, Distributed Ledger Technologies**

I. Introduction

Emerging IoT devices and applications have been applied to various domains: healthcare [1], transportation [2], [3], agriculture [4], and others [5]. As a result, IoT devices collect and process large amounts of private and confidential information. This makes privacy a key concern in the context of open IoT systems in which IoT devices are open to multiple users and data consumers. Today, IoT privacy research is focused on software practices [6]–[8], human interaction [9], vendor heterogeneity [10], [11], insecure data flow [8], [12]–[15], and authentication [17], [18]. Less attention has been given to authorization and auditing. Access control consists of three components: authentication (verifying user identity), authorization (enforcing access control rules and preventing illegal actions), and auditing (tracking down and verifying activities).

Access control for IoT devices is a challenging task. IoT devices are unable to directly authorize users and audit activities because of their limited resources [9]. Thus, the traditional computer access control mechanisms [35], [36] are not applicable. In addition, vendor and device heterogeneity make it more challenging to build and enforce access control rules in a consistent way [11]. Meanwhile, IoT applications like surveillance and visual analytics usually require real-time performance [19], so it's critical that access control be done in a timely manner.

Complicating matters, authorization and auditing for IoT devices are fundamentally different from traditional devices [9]. Traditional edge devices like personal computers and phones usually interact only with a single user/owner. Once user authentication is completed, further authorization and auditing can be easily accomplished in the same device [20]. However, numerous users may simultaneously share one device in an IoT environment and this leads to different requirements of data ownership and management [21], such as a smart home's shared voice assistant [9] or surveillance in an Airbnb room [22]. Such IoT devices with limited resources are not capable of authenticating and auditing all users' activities. Furthermore, users may have complex social relationships with each other, which complicates the problem [9]. For instance, mischievous children [23], abusive romantic partners [24], and parents who keep their teens under surveillance [25] are internal threats to the home IoT environment [9]. In such scenarios, individual users intimately sharing IoT devices may occasionally abuse private data. As another example, smart cities may suffer from malicious service providers that corrupt or leak sensitive data and cause financial or privacy-related damage [26], [27]. To provide smart environment services, like weather and noise monitoring [27], many service providers need to collaborate and share data. Such collaboration relationships are similar to social relationships among individual users. Internal malicious providers are also significant threats in smart cities [26] and none of them are fully trusted to control data access. Thus, conventional role-based access control [35] does not fit in the above scenarios.

On the other hand, the challenges of multi-user settings and complex user relationships can provide leverage for the IoT access control problem. With multi-user settings, we assume that no single user or edge device is trusted, thus we require a trusted quorum to perform secure authorization and auditing. Furthermore, real-world relationships can be used to emulate the trust relationships between edge devices to build quorums and assist in access control.

In this paper, we propose a new IoT access control archi-

This work is supported in part by NSF grant NSF CNS-1908566.

tecture that supports a multi-user IoT ecosystem. We consider a common IoT infrastructure [9], [10], [28], [29] in which users use their edge devices such as personal computers and smartphones, to interact with IoT devices and perform activities. This architecture supports consistent access control rules and fits multi-user scenarios. We adopt a trust model based on localized IoT device usage and real-world user relationships to establish local quorums and validate access requests. The key observation is that real-world relationships can be utilized to develop quorums for access control. Specifically, devices owned by the same user shall trust each other. And users who are in primary relationships that enjoy direct and intimate connections, like commercial partners and spouses, may be included to assist in the authorization process. In addition, the users who are in an indirect relationship, such as the users' parents of parents, maybe less trustworthy. We define such trustworthiness as a tiered structure and use it to develop a local quorum per device. The local quorums conduct authorization decisions through a lightweight consensus algorithm based on Federated Byzantine Agreement (FBA). In addition, quorum edges maintain distributed logs to audit IoT device activities. Specifically, we make the following contributions:

- **Analysis of the IoT multi-user setting in edge-based infrastructure.** We observed that the unique challenges and threat models require a new access control method that is significantly different from that of traditional computer systems.
- **Propose a social relationship driven method for IoT access control.** We develop local hierarchical quorums using real-world relationships between individuals for collaborative authentication of access requests to address these unique challenges.
- **Adopt a lightweight local consensus algorithm.** We adopt a lightweight consensus algorithm suitable for distributed and hierarchical local quorums in the authentication process.

We implement a prototype of our system called SQuBA in AWS platform and evaluate its performance with a real-world IoT dataset, SIoT [30], which includes social information and proximity of an IoT network that consists of 14600 devices. Based on this dataset, we construct a trust model in a multi-user environment. We then study the configurations, assess end-to-end latency and resilience to faulty edges, and compare our architecture with other consensus-based authorization methods. The experimental results show that our architecture provides promising real-time performance and good privacy guarantees. SQuBA is able to reduce end-to-end latency by up to 10X and tolerate the number of faulty edges by up to 2X when compared to baselines.

II. Background

A. Edge-based IoT Infrastructure

We assume an edge-based IoT infrastructure [10], [12], [28], [29] as shown in Fig. 1. In this deployment, the network consists of two types of entities: IoT devices and edges. The IoT devices, such as surveillance devices and smart TV, interface with one or multiple edges that undertake the tasks of various applications. The edge infrastructure comprises multiple types of edges such as gateways, controllers, edge servers, and edge devices. We assume that the consumers of IoT data, such as end-users and service providers, are authenticated by the edge devices [9], such as smartphones and personal computers, and use it to interact with IoT devices. In the following sections, the terms "edge" and "edge device" are used interchangeably. For example, the household can turn on the smart TV through a smartphone app; an intelligent healthcare provider may access the patient's heartbeat rate through a web app. In this setting, the users and service providers may have complex real-world relationships and their edge devices are further interconnected to form a peer-to-peer network, which can accomplish more complicated tasks in more distributed settings [10].

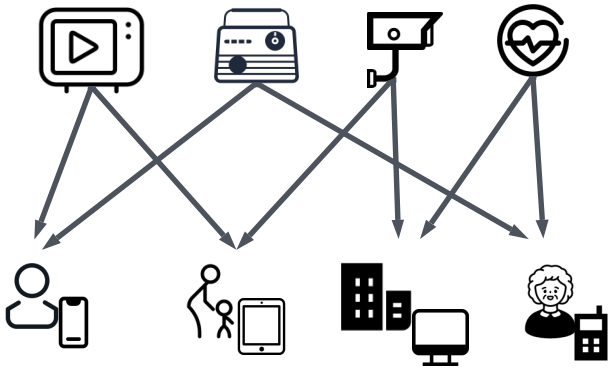


Fig. 1: Edge-based Infrastructure. IoT devices interface with edges and users.

B. IoT Multi-User Settings

The multi-user setting refers to the scenario in which multiple entities interact with the same IoT device. They can be end-users in smart homes or service providers in smart cities.

IoT privacy is a rising concern. Studies [22], [31], [32] show that the multi-user setting is a key challenge in IoT privacy. In the era of emergent IoT, it is very often that complete strangers may share personal services, such as surveillance in an Airbnb room. These sharing schemes raise significant concerns about IoT data privacy [22], [33]. Smart cities that require collaboration between service providers [27] is also a typical multi-user scenario.

In addition, some prior work [9] has identified social ties between users in home IoT environments, such as roommates, neighbors, and children. Given the importance of social relationships, privileged-abused users, such as abusive romantic partners, are internal threats that are magnified in the IoT environment. Considering that IoT devices are able to perform various operations, such as reading heart-rate and monitoring noise level, an access control mechanism should exploit both

social relationships and IoT device operations together [9] to provide resilience to internal threats.

C. Threat Model

The two major classes of threats are external third parties and internal privilege-abused entities. The former class refers to the attacks from external adversaries that exploit software vulnerabilities attempting to span malicious edges in the network. Examples are discussed in [7], [8], [12]. The latter class includes those internal entities that abuse the privilege. In this case, a user might be compromised by attackers or motivated to subvert a smart-home system's access control rules for disobedience, such as a child attempting to take actions forbidden by their parents [9]. It also includes users that attempt to compromise the data privacy of IoT devices that are shared with other users. For instance, abusive romantic partners [24] may secretly download household surveillance videos. Also, it considers the local threat that is in proximity and misuse of IoT devices, such as a maintenance worker might unlock the door through a smartphone app. Additionally, the service providers who attempt to corrupt and leak data also belong to this class. Overall, in the multi-user scenario, no single user, service provider, or edge is eligible to manage an IoT device alone. We aim for an architecture that fits the IoT context and supports the multi-user setting.

D. Challenges and Opportunities

The open IoT access control problem presents two main challenges. First, limited resources prevent IoT devices from completing the access control tasks. As a result, delegating these tasks to edge devices becomes a common solution. However, multi-user settings in the IoT environment complicate data ownership and management, which creates another unique challenge. Second, none of the edge devices are trusted. On the one hand, the multi-user setting is vulnerable to internal threats such as privilege-abuse users. On the other hand, the intricate relationship between the users provides leverage. We observe the opportunity to utilize users' relationships to guarantee privacy by collaboratively enforcing access control rules.

III. SQuBA Design and Implementation

We next present our consensus-based architecture, SQuBA, to provide access control for the open IoT environment.

A. System Overview

SQuBA is a consensus-based authentication approach in a edge-based IoT infrastructure where the end-users interact with IoT devices through edges. Each IoT and edge device are authorized and identified as uniquely identifiable by a set of secure, non-replicable credentials. In this work, certificate equivalent signatures generated by ECDSA algorithm [16] are used as the credentials. After authorization, IoT devices will own their own local quorums based on their real-world relationship (More details are discussed in the following sections). Quorum edges shall enforce the access control rules and vote to either deny or approve access requests. The

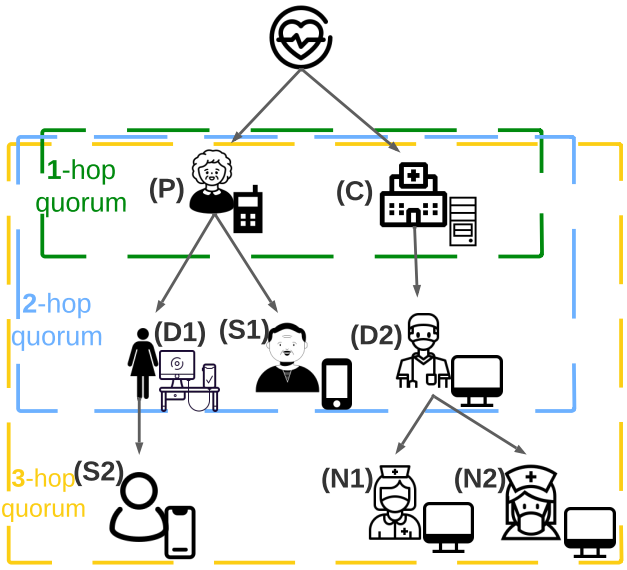


Fig. 2: An example of a 3-hop tiered quorum.

local quorums are responsible for enforcing access control rules, authenticating future access requests, and auditing IoT activities. Specifically, our architecture supports operations-based access control rules considering its fine-granularity [9], [17]. Each rule is a triplet that includes [Operation, User, Contextual Information]. In addition, the quorum edges keep distributed, consistent, and append-only ledgers of realized activities for auditing.

B. Relationship Driven Quorum

To adapt to the multi-user scenario, each IoT device owns a local quorum that is a union of end-users' quorums. The key insight is that we consider real-world relationships to construct the IoT devices' trust assumptions. First, all users of the given IoT device should participate in the request validation for that IoT device. Further, the entities might have intimate connections with others and rely on them to make decisions. Thus, we can also employ social ties to construct the local quorum. In the real world, people who are in primary relationships that enjoy direct and intimate connections, like commercial partners and parents, usually trust each other, while people who are in indirect relationships may be regarded as less trustworthy. This trust model results in tiered quorums.

The tiered structure consists of a succession of edges of decreasing trustworthiness. The primary trusted edges of a given IoT device are called 1-hop edges. They are edges that belong to the users of the IoT devices. Those edges participate in the daily usage of IoT devices and are most relevant to IoT device privacy, and are mostly likely the device and edge owner. Thus, 1-hop edges are the most trustworthy from the IoT device's perspective. Then, the edges that are primarily connected to 1-hop edges are called 2-hop edges. The 2-hop edges are closely connected with the 1-hop edges but they may be less trustworthy in the IoT device's view.

The tiered structure is needed because 1-hop edges can be compromised by an adversary, thus multi-level consensus is required. However, the trustworthiness attenuates with more hops away from the given IoT device. A hop threshold can be set up to constrain the trustworthiness and quorum size. As a consequence, the local quorum can be built in a tiered structure, and Fig. 2 presents an example of a tiered quorum with a hop threshold as 3. In this example, a sensor monitors a patient's heart rate and synchronizes data to the user's smartphone (P) and the clinic database (C). These two edges are the 1-hop quorum of the sensor. We can expand the 1-hop quorum based on users' social ties and obtain the 2-hop quorum, where the patient depends on her spouse (S1) and daughter (D1) and the clinic depends on the doctor (D2). Similarly, the 3-hop quorum further includes the daughter's spouse (S2) and nurses (N1 and N2) who work with the doctor.

C. Local Consensus

The quorum edges enforce access control rules and make authorization decisions through a lightweight consensus algorithm, tiered federated voting (TFV), which is based on federated voting in the Federated Byzantine Agreement (FBA) [48].

First, all quorum edges will synchronize the access control rules of the given device. Then they vote to approve the request only when all three factors of the access control rules are satisfied. Otherwise, they should vote for rejection. Specifically, the edges will check the client identity of the request and the capability of the target IoT device.

In addition, contextual information, such as the time of the day and the location of the client, will also be considered. For example, if the access control rule is [Turn on SmartTV, child#1, 9 am-9 pm], the client whose identity is not child#1 should be rejected. Also, the child#1 will not get permission from the edges to turn on the TV at 10 pm because of a contextual violation. Furthermore, each edge will multi-cast its voting ballot to all quorum edges to avoid byzantine votes. Similarly, the final decision must be propagated to the quorum and visible to everyone for future auditing.

As for the consensus algorithm, federated voting [48] is designed for a tiered structure quorum. The key insight behind federated voting is that edges can be convinced by their own local quorums. Specifically, it employs a three-phase protocol in which edges first vote for a statement (access request in this architecture), then accept/reject it, and finally confirm the decision [48]. Fig. 3 shows the steps of federated voting: A node $\boldsymbol{v}$ may vote for any valid statement $\mathbf{a}$ and only accepts $\mathbf{a}$ when $\boldsymbol{v}$ is a member of a quorum in which every node either votes for $\mathbf{a}$ or accepts $\mathbf{a}$. Even if $\boldsymbol{v}$ did not vote for $\mathbf{a}$, if $\boldsymbol{v}$'s quorum contains a node accepting $\mathbf{a}$, then $\boldsymbol{v}$ also accepts $\mathbf{a}$. Finally, when $\boldsymbol{v}$ is a member of a quorum in which every node accepts $\mathbf{a}$, then $\boldsymbol{v}$ confirms $\mathbf{a}$ [48]. However, the most important nodes in federated voting are the leaf-level nodes [49] because the leaves can persuade the top nodes. This is incompatible with the attenuating trustworthiness of social ties in our architecture and requires modification.

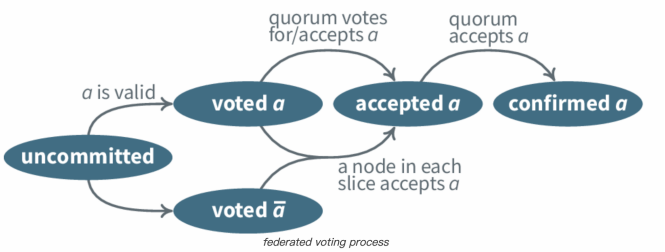


Fig. 3: Federated Voting [48].

To fit the trust relationship that the top-tiered nodes are the most trustworthy from the IoT device's perspective, we propose tiered federated voting (TFV). The principle is that the top-tiered edges are more important because they are directly and closely related to IoT device usage. In other words, they have more influence when making authorization decisions. However, due to software vulnerabilities and privilege misuse, any edges may turn out to be faulty regardless of their position in the quorums. In fact, internal threats from an intimate connection are a significant consideration. Faulty edges may violate access control rules and intend to cause physical, financial, and privacy-related damage. Hence, the voting shall expand quorum hops and involve more edges when some of the top-tiered edges turn out to be faulty and fail to reach a consensus. In this case, the adversary must make more efforts to either investigate the quorum structure or compromise more edges to control the majority and final decisions. Based on these insights, we modified federated voting as follows: The voting process contains multiple iterations. The first voting iteration only happens among the 1-hop edges, it terminates once consensus is reached among the top-tiered nodes. Otherwise, it starts the second voting iteration and initiates federated voting among the 2-hop quorum. Overall, it gradually expands the voting to a lower level of edges until the hop threshold or consensus is reached.

As for the agreement threshold, each quorum can have its own threshold depending on the quorum quality and trust level [51]. It can be 100% for critical security or simply 51% majority if the quorum edges are fully trusted. Moreover, the hierarchy of quorums provides more flexibility to define the agreement threshold. In particular, each level of a quorum can have an individual agreement threshold. It is an interesting question to weigh the quality of quorums and set up an agreement threshold accordingly. For instance, the threshold of the majority can be simply 51% among the 1-hop quorum, and the 2-hop quorum threshold might be 0.67 since they are less trustworthy. Similar to access control rules, setting the agreement threshold is outside our scope.

Fig. 4 shows an example of tiered federated voting applied to the healthcare example. When a new access request is received, the patient edge (P) and the clinic database (C) shall vote to either approve or reject the request. This 1-hop quorum can draw a decision once a consensus is reached. Otherwise, they should initiate the second voting iteration among the

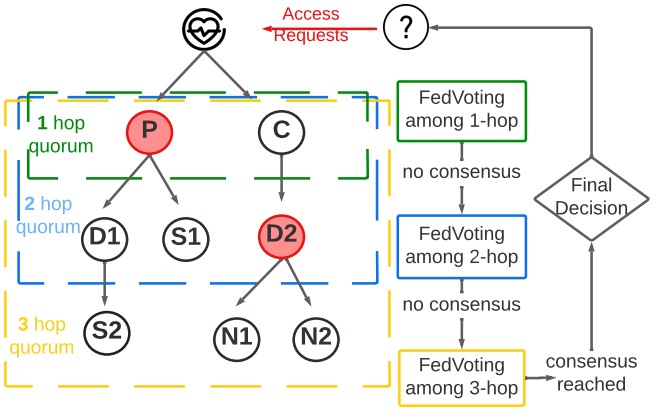


Fig. 4: A Tiered Federated Voting process.

Iteration	P	C	D1	S1	D2	S2	N1	N
1	✗	✓						
2	✓	–	✓	✓	✗			
3	✓	✓	✓	✓	✓	✓	✓	✓

Fig. 5: Ballots of the above voting process.

2-hop quorum. In this case, the patient edge (P) and the clinic database (C) might be convinced by their own quorums and finally reach a consensus. Similarly, if no consensus is achieved among the 2-hop quorum, the tier 3 edges, the spouse of the daughter (S2), and nurses (N1 and N2), will participate in the third iteration of federated voting.

There are three outcomes of the tiered federated voting. First, when non-faulty edges control the majority, they can reach a decision that respects the access control rules and the IoT device is secure. In the healthcare example with a simple 51% agreement threshold, the IoT device is protected when the 1-hop quorum edges, the patient (P) and the clinic (C), are honest. Even if the higher-level edges become faulty, the non-faulty lower-level edges can still persuade the top-tiered edges and reach a correct decision. This is demonstrated in Fig. 4. The red shaded nodes P and D2 turned out to be faulty and this led to a lack of consensus in the first two iterations as shown in Fig. 5. However, the third iteration involves S2, N1, and N2. Those honest nodes will finally persuade the faulty nodes and achieve proper consensus. The second possible outcome is a lack of majority and consensus. Given the healthcare example, this happens when P, D1, S1, and S2 are all faulty. Alternatively, there is no consensus when C, D2, and N1 are faulty. The third outcome is a faulty decision where the faulty edges outvote the non-faulty edges and control the majority. In this case, the protection of given IoT devices completely fails when the 1-hop quorum, P and C, are faulty. The same outcome is obtained at the second voting iteration when C, D1, and S1 are faulty.

When the local quorum reaches a consensus to approve the access request, all 1-hop edges of the IoT local quorum send the approved action to the IoT device which then responds accordingly. The IoT device will not respond until it receives approval messages from a majority of 1-hop edges to mitigate approval messages from rogue edges. In this case, choosing one single entity to enforce the consensus decision and route the request to the IoT device is insecure due to internal threats. Actually, avoiding faulty nodes when enforcing the consensus decision is still an open research problem in such open systems: round-robin [52] and random selection [50] are common techniques, however, they are not promised to eliminate the probability of selecting faulty nodes.

Local quorums and the tiered structure make our architecture more resilient to failure under the threat model. For example, if the adversary made an external attack by spanning a million malicious edges to the system, a typical system-wide agreement would fail because the majority of voting is controlled by the adversary. However, this does not affect SQuBA because of the distributed local quorums of IoT devices. As for internal threats from privilege-abused edges, the top-tier faulty edges will be persuaded by the honest edges in the lower level of quorums. Meanwhile, the leaf-tier faulty edges do not participate in the voting if the top-tier honest edges reach a consensus and dominate the decision. In addition, because of no global consensus, a failure of an individual quorum does not interfere with the others. Overall, the adversary must make a greater effort to deduce the structure of all quorums and then compromise a sufficient number of edges.

D. Journals

In addition to authorization, our architecture also provides a multi-user auditing mechanism based on a distributed ledger. The aim is that all users shall be able to audit the realized activities of the IoT devices. Constructing a log of activities is a common solution for auditing [42]. But a distributed ledger usually refers to whole system-wide records which contain information for all entities' activities. This consumes huge storage space and it may take hours to synchronize when new participants join [40]. In addition, the global public history reveals activity patterns and this may compromise data privacy [40]. Thus, we present device-oriented journals for audit activities.

Generally, a journal [40] is a public data structure of a group of nodes and contains only part of the records of the whole system. In this work, each IoT device owns a local quorum; each journal only maintains information of a single IoT device; and each edge in the quorum of that IoT device keeps the consistent, append-only, immutable journal. The journal is updated when access control decisions are made by the quorum. Thus, the distributed journal is compatible with local consensus and keeps it consistent per IoT device.

Each device-oriented journal consists of three categories of information:

- quorum relationship, the edges that are included in the IoT device's quorum;
- access control rules, the triplets to validate access requests;
- hashes of historical approved access.

The journal is implemented as a hashchain. A hashchain is an ordered set of blocks with a cryptographical hash to commit to every approved request of an IoT device in the ordered chain [42]. Each block has a unique header that contains a hash of current configurations that consists of access control rules and quorum relationships. The header also keeps a timestamp that proves the session of accesses happened after the block was created. Further, the quorum edges can reach a consensus about lease time for each block to guarantee liveness. A new block will be created when the lease time is expired. In addition, the hash of the previous block is maintained in the block header so that the chain is irreversible and append-only. When a new request is approved through the quorum, all quorum edges check it and ensure that the new access statement is a valid extension of the block it previously had from the IoT device. Specifically, each approved access statement contains the client ID hash, requested capability, and real-time contextual information. Each block only accommodates accesses that happen under the block header's configurations. Once the configuration is changed, a new block with an updated header will be created for future records. The journals can be used to audit the IoT device's activities for posterior analysis, such as looking up periodic requests and tracking down the potential cause when data leakage occurs.

The IoT devices synchronize and update the quorum relationship and access control rules by multicasting (sending data to multiple receivers simultaneously [41]) to its quorum edges. The historical hashes are synchronized during the voting process. When a new approval consensus is reached, all quorum edges distribute confirmation [48] thus they can automatically append the hashes and keep synchronized. The journal should be identical among all quorum edges during the life-cycle of the IoT device. In addition, the quorum edges can send a catchup message [52] to each other to check that their distributed journals are consistent.

Our design uses a hashchain with respect to each IoT device. This approach is compatible with our quorum and consensus architecture. Additionally, it does not require each edge to store a system-wide log of activities. This leads to less storage consumption and protects data access patterns.

There are other interesting directions and open questions about combining hashchains in IoT ecosystems. For instance, the hashchain can be capability-oriented so that one contains logs of temperature control and another one keeps lighting system history. In this case, the question of how to update and where to store the hashchain must be decided.

E. Skip-voting Mechanism

Access requests in IoT environments are more likely to be repeated due to periodic queries. We take advantage of the recorded access history to reduce latency for such requests by re-using prior decisions. We introduce a skip-voting mechanism to accelerate the voting process of repeated requests. When a new request is marked as repeated, the quorum edges will look up the journal history and skip the first voting stage among the three phases (voting, accepting, and confirming). In other words, the voting process of a repeated request consists of three phases: the edges of the corresponding 1-hop quorum 1) lookup the request if the historical access is approved, 2) accept the request if the real-time contextual information aligns with access control rules, 3) confirm the decision, update and synchronize the journal with the whole quorum accordingly. The skip-voting mechanism can save communication overhead of the first voting stage and following the tiered voting process. Thus, it can save end-to-end latency for periodic requests.

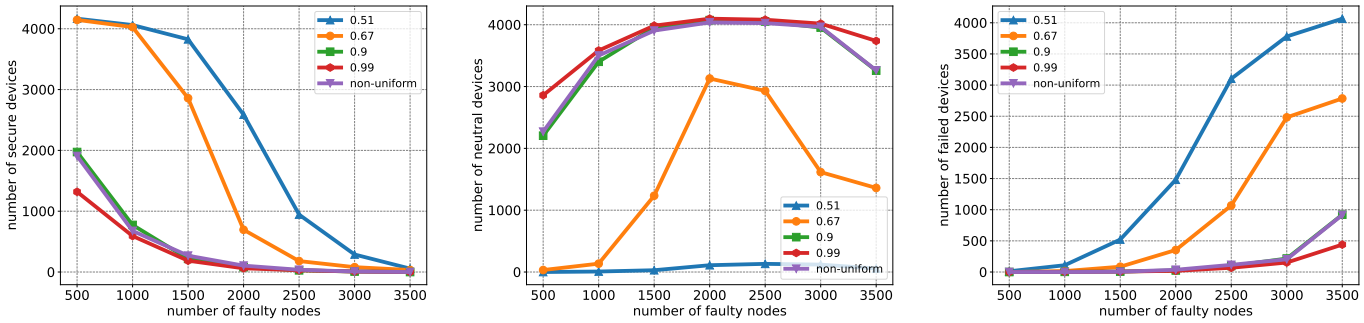
The journal guarantees the freshness of quorum configurations and prevents stale updates. In this case, the expired previous blocks will not be accessed. So an access history under stale configurations is not considered when applying skip-voting. Also, the quorum edges still account for contextual information for repeated requests. For instance, considering a smart TV policy, [Turn on TV, Child, 10am - 9pm], although historical access under the same configuration is approved at 4pm, a repeated request at 10pm shall not be accepted because of a contextual information violation.

IV. Evaluation

In this section, we evaluate SQuBA using a real-world dataset in terms of end-to-end latency to validate access requests and resilience to faulty edges. We analyze the system configurations and compare them with four baseline authorization methods.

The dataset, SIoT [30], contains information about IoT and edge devices that are installed in the city of Santander in Spain. A total of 14600 devices are owned by 4000 users. Among those, 7000 devices are edge devices, such as smartphones and PCs. 7600 devices are IoT devices such as smart fitness devices, printers, home devices, etc. The dataset also provides device relationships and we take advantage of ownership and social relationships to emulate multi-user scenarios. Ownership means devices are owned by the same owner. Social relationship means the devices "meet" (are in close proximity) with each other beyond a certain frequency (more than 3 times per 10 days) and duration (more than 15 minutes). Because proximity is an important usage factor in the IoT context [9], the users who are in "social relationships" with IoT devices are considered as users of the given IoT device to emulate the multi-user scenario. We also deploy social relationships as a primary relationship between edge users. Thus, each IoT device's 1-hop quorum contains the edges that are likely to be the users of the given IoT device.

Experiments are performed on 100 AWS EC2 machines, each machine contains 70 threads emulating edges. Based on the SIoT dataset, edges develop quorums for 7600 IoT devices and validate access requests based on their access control rules. All access control rules and access requests are synthetic.



(a) Number of secure devices (b) Number of devices lack consensus (c) Number of failed devices

Fig. 6: Effects of various agreement thresholds

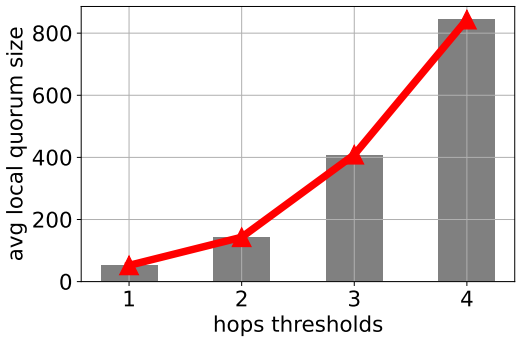


Fig. 7: Quorum sizes vs. hop thresholds

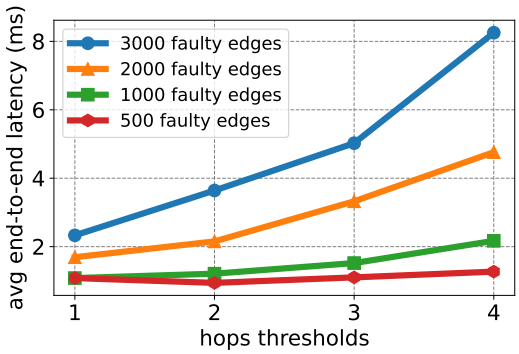


Fig. 8: Average end-to-end latency with various numbers of faulty nodes and hop thresholds.

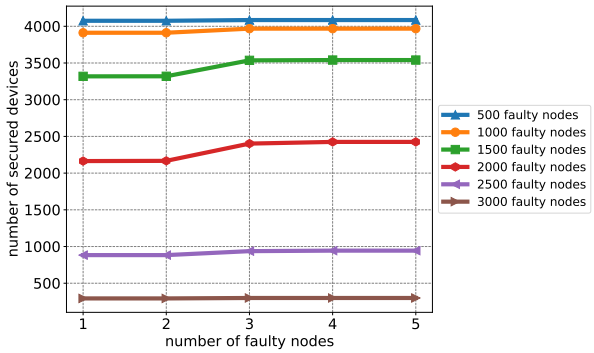


Fig. 9: Number of secured IoT devices with various numbers of faulty nodes and hop thresholds.

A. *Configuration Analysis*

1) Agreement Threshold: The agreement threshold is an important parameter. Each quorum can have its own agreement threshold depending on the quorum quality and trust level trade-off. The threshold can be 100% for critical security or simply 51% majority for high-quality quorums. Given the threat of software vulnerabilities and privileged-abused internal users, the edges can be faulty when they are compromised or misused. Consequently, the faulty edges attempt to allow illegal access without respecting IoT devices' access control rules. In this case, the quorum can not reach a consensus when faulty edges obstruct voting. In the worst case, if the faulty edges control the majority, then they can outvote the non-faulty edges. The quorum hierarchy complicates the analysis of the agreement threshold. In Fig. 6, we assess the effects of various agreement thresholds: 0.51, 0.67, 0.9, 0.99. Additionally, we also examine non-uniform agreement thresholds where the agreement threshold of the 1-hop quorum is 0.51, and the 2-hop quorum threshold is 0.6 since they are less trustworthy. Then the 3-hop quorum threshold is 0.7. Each quorum contains 3-hop and all faulty nodes are randomly selected. Because each IoT device owns its local quorum, the number of secure, failed, and lack consensus IoT devices are measured to present the effects of various agreement thresholds.

Obviously, more faulty edges lead to more unprotected IoT devices. However, higher agreement thresholds induce fewer failed IoT devices but increase the risk of failing to reach a consensus. Surprisingly, the non-uniform threshold performs similarly to the 0.9 threshold in the given dataset.

2) Hop Threshold: SQuBA constructs tiered quorums and the trustworthiness attenuates with more hops away from the given IoT device. A hop threshold can be set to constrain the trustworthiness and quorum size. Generally, a larger hop threshold leads to larger local quorums and deeper hierarchies. Intuitively, there is a trade-off between privacy guarantees and latency. Including more edges will increase the difficulty

in compromising the quorum but increase the latency to validate access requests. We present the average quorum sizes with various hop thresholds in Fig. 7. It demonstrates that expanding to more hops can involve more edges in the quorum.

We then evaluate the end-to-end latency with various hop thresholds. End-to-end latency is the time interval between sending a request and receiving a response. Fig. 8 is the average latency to process requests when the hop thresholds change and uses a uniform agreement threshold of 51% everywhere. In this case, the latency increases with a larger hop threshold. Generally, more hops lead to larger quorums. This results in longer end-to-end latency. However, because the voting is likely to converge among the top-tiered edges when there are few faulty edges, the latency does not grow as the hop threshold increases in this scenario.

The hop thresholds will also affect the privacy level of our architecture. Here, we assess the resilience to faulty edges with different hop thresholds in Fig. 9. The internal edges are faulty when they are compromised due to software vulnerabilities or privileged-abused users. The faulty edges attempt to allow illegal access without respecting IoT devices' access control rules. As a result, the IoT devices are not secure if the faulty nodes control the majority of the quorum. Due to local quorums and journals, authorization and auditing are granted per device. Thus, we consider the number of secured IoT devices to quantify the privacy level. In this case, all faulty nodes are randomly selected and a uniform agreement threshold is set as 51% which is a simple majority. The results indicate that arbitrarily increasing hop thresholds and increasing quorum size do not enhance privacy guarantees.

The Fig. 9 and Fig. 8 also demonstrate the trade-off between privacy guarantees and latency. Keeping a smaller quorum size enjoys better end-to-end latency. But it may be more vulnerable to internal faulty edges. In the SIoT dataset, choosing the hop threshold as 3 or 4 is reasonable.

B. Performance Comparisons

We now compare SQuBA performance with other authorization methods: a system-wide authorization protocol, a single host edge authorization, a 1-hop quorum, and a flat 3-hop quorum. For system-wide authorization, we deploy the two-phase commit protocol as the consensus algorithm [53] among all edge nodes of the network. For the single host edge authorization, one edge that is owned by the same user with the IoT device is assigned to be its authoritative entity. To assess the impact of quorum hierarchies, we also develop local quorums based on trust assumptions without hierarchy: the 1-hop and flat 3-hop quorums. They will perform a two-phase commit protocol among the local 1-hop and 3-hop quorums without hierarchy to reach a consensus. Generally, the system-wide quorum will conduct global consensus. The 1-hop, flat 3-hop, and SQuBA quorums conduct local consensus and the quorums are built on users' real-world relationships. Furthermore, SQuBA appreciates the trust attenuation along with hops expansion by deploying a hierarchy of quorums while the 1-hop and flat 3-hop do not consider such a trustworthiness

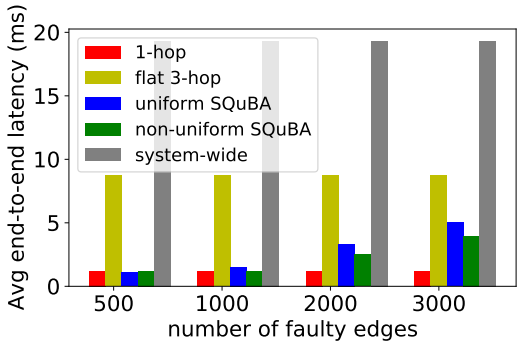
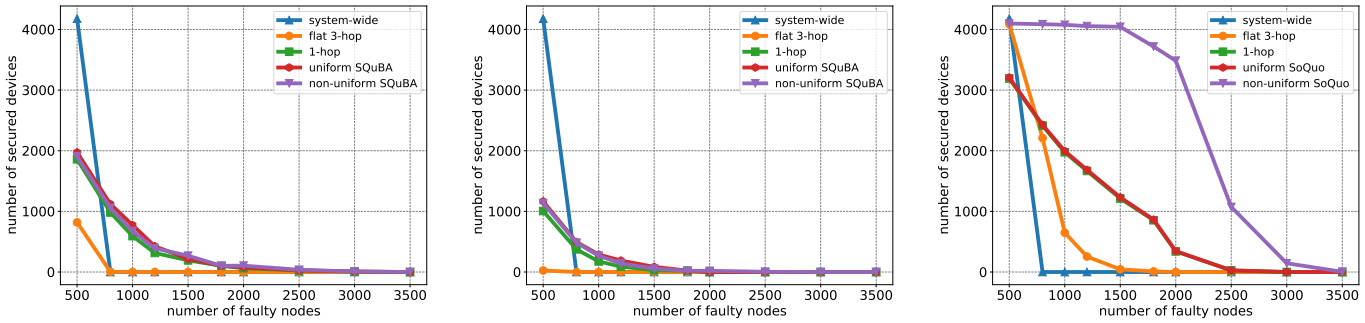


Fig. 10: Average end-to-end latency of various authorization methods.

setting. The uniform agreement threshold is set as 0.9 since it is for strict security. Additionally, the SQuBA quorums set the hop threshold as 3 so that the SQuBA will have exactly the same quorum elements as the flat 3-hop quorums.

1) Latency Performance: We compare the latency with other authorization methods in various configurations in Fig. 10. Obviously, a larger quorum size leads to higher end-to-end latency. The 1-hop quorum has the smallest latency since it involves a small number of edges. Meanwhile, SQuBA can outperform the system-wide and flat 3-hop quorum because SQuBA can early terminate the voting once consensus is reached among the top-tiered edges. Especially when comparing the system-wide authorization protocol with the two-phase commit protocol, SQuBA saved about 10X end-to-end latency. The non-uniform threshold setting can further save latency due to a smaller threshold among high-level edges. Thus, the tiered quorum and voting can save latency by exploiting real-world social relationships. Currently, both real-time query generation [18] and processing time [54] of IoT systems are typically in the range of milliseconds. Thus, our architecture is able to fit IoT applications.

2) Resilience to Internal Faulty Edges: In this part, we consider three faulty models regarding internal threats. The first faulty model is that internal edges turn out to be faulty in a random manner. The second model is hotspots-prior in which the edges that are more likely to be top-tiered became faulty first. This corresponds to the scenario that internal privilege-abuse users, such as romantic abusive partners [24], attempt to violate given access control rules. This may also happen when the adversary may observe the network traffic and hack the hotspots first. In the SIoT dataset, there are around 1800 edges that are involved in more than 23 IoT devices' 1-hop quorum. These edges are more likely to be top-tiered nodes of local quorums. Thus, they might be more participative and important when making decisions. Hence, they are regarded as hotspots and the rest are viewed as bystanders who are less participative in the authorization process. The third faulty model is bystanders-prior in which the bystanders are compromised first. This corresponds to the scenarios in which the less trusted participants are malicious.



(a) Under evenly distribution (b) Under hotspots-prior distribution (c) Under bystanders-prior distribution

Fig. 11: Resilience comparisons

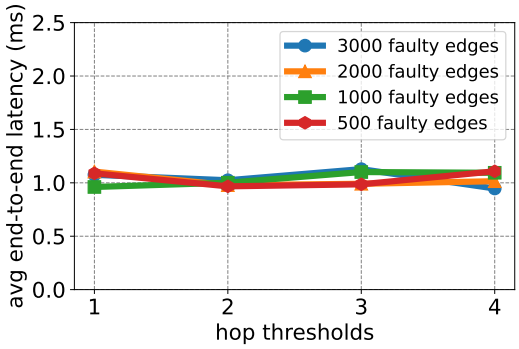


Fig. 12: Average end-to-end latency for repeated requests.

External attacks are not discussed because it never affects SQuBA's privacy due to distributed quorums.

The results are shown in Fig. 11. Generally, SQuBA is the most resilient to faulty edges under all of the above faulty models. First, injecting new faulty edges does not affect SQuBA's privacy because these new malicious edges are not included in IoT devices' local quorums. But the system-wide consensus fails when the number of new faulty edges grows. Similarly, as shown in Fig. 11a, the flat 3-hop and system-wide quorums completely fail to protect IoT privacy with a limited number of faulty edges, while the 1-hop and SQuBA can still protect many IoT devices. This result suggests that a local quorum based on users' social ties is beneficial in terms of privacy. Blindly expanding the quorum does not guarantee better privacy but increases the risk of involving malicious edges. In addition, because the local quorums do not interfere with each other, some IoT devices are still secure even if a significant number of edges turn out to be faulty.

Furthermore, the quorum hierarchy is also beneficial. Specifically, all types of local quorums perform better in the bystanders-prior model in Fig. 11c and are more sensitive to the hotspots-prior model in Fig. 11b. This is because the hotspots are more likely to be top-tiered edges of various quorums. Thus, they might be participants and important when voting. Once the top-tiered edges are compromised, the IoT device is more likely to be unprotected. However, the SQuBA outperforms flat 3-hop and 1-hop quorums in all models. This presents the benefits of the quorum hierarchy. The privilege-abused users' ballots are persuaded by the lower-level non-faulty edges. As for the faulty edges caused by software vulnerabilities, the adversaries cannot control the validation decisions of SQuBA unless they make a greater effort to discern the quorum structure or compromise more edges. Particularly, the non-uniform threshold is beneficial in the bystanders-prior model. This is because the smaller threshold among top-tiered edges allows them to dominate the voting and prevent faulty edges in the lower level. As a consequence, SQuBA can tolerate approximately 2X the number of faulty nodes vs. the system-wide authentication methods when securing the majority of the IoT devices.

C. Skip-Voting Performance

We proposed the skip-voting mechanism to more efficiently handle repeated requests. Fig. 12 is the average end-to-end latency to process repeated requests under various configurations with a uniform agreement threshold of 51%. The skip-voting mechanism significantly reduces latency up to 5X. In addition, the average end-to-end latency for validating repeated requests stays constant irrespective of environmental conditions (e.g. size of network, load, etc.).

V. Related Work

1) Traditional Access Control: There has been much work on privacy-aware and context-based access control [36], [37], which enhances access control models with specific components, such as roles and contexts. But they are designed for traditional computer systems. However, the multi-user setting and constrained resources of IoT devices make it non-applicable to the IoT access control problem. Also, collaborative access control [34], [35] does not consider the complex relationships among users and potential internal threats in the open IoT environments.

2) IoT Access Control: Much research has been conducted on access control architectures in the IoT environment. This work can be divided into three main categories: cloud service, single host edge, and system-wide consensus.

Using a cloud service to manage IoT access is a common solution in many IoT applications [38]. The disadvantage is that vendor and device heterogeneity can lead to complex interfaces that are confusing to the user [11]. As a consequence, users may fail to set up a proper access control rule. Cloud data leakage is also a significant concern [15].

The second class is a user-provided host edge. This host edge is fully responsible to enforce access control rules of IoT devices before releasing data to the vendors and other edges [11], [12], [39]. However, this approach relies on a single trusted host edge which is not suitable for open multi-user environments. In addition, authorization fails when the given edge is compromised and non-trusted.

Another approach is to conduct a system-wide consensus when validating access requests. This approach usually relies on blockchain platforms [22], [33], [40], [42]. Such architectures require independent edges and IoT devices to build a global consensus around a public, distributed, and append-only 'ledger' that is a chain of access hashes, without relying on a central coordinator to provide the authoritative version of the records. Furthermore, the current generation of blockchains, such as Ethereum [43] and Bitcoin [44], have introduced smart contracts [45], a programming logic residing in the blockchain as byte codes that are automatically executed when certain messages are triggered. Blockchain and smart contracts have been proposed by many [22], [33], [45], [46] as an access control mechanism. This authorization architecture fails when the majority of the entities are controlled by the adversary. Furthermore, blockchain platforms suffer from high overheads and require intensive computation and storage [47] that are unfavorable for IoT devices and applications. The execution of a smart contract may take up to hours [46] which may be unacceptable for many IoT applications. One study [47] proposed a blockchain optimization to fit in the IoT context. This approach keeps a trust rating system of entities to decrease certain mining process overheads. But it still requires a global ledger of all IoT devices in the given network and is less efficient when the set of users is changing.

3) Comparing with Prior Methods: We also contrast our architecture with other IoT access control methods. Specifically, cloud-based authorization suffers from vendor heterogeneity and fails to provide consistent access control rules [11]. And it does not guarantee real-time response due to network congestion and round-trip time [40]. In addition, Cloud data leakage can lead to a complete failure of access control [15]. For the blockchain-based approach, it is resource intensive and may be too expensive. Furthermore, it completely fails against 51% attack when the adversaries controlling more than half of the hashing power of a whole system [55]. In other words, all IoT devices of the system are insecure once the attacker controls the majority of hashing power. For the single-host edge method, it is hard to support multi-user scenarios because it can not prevent the host edge from abusing IoT data. Once the attacker controls the edge host, all IoT devices that are managed by the victim edge will be vulnerable. Also, it can not prevent the edge owner from misusing data because of the edge host completely dominates IoT access. However, our proposed architecture is able to support multi-user settings in the open IoT environment and provide real-time end-to-end latency. As for privacy, due to hierarchical structure of quorums, it only fails when the attacker learns the quorums' structures and compromises enough edges accordingly. Furthermore, because each IoT device owns a local quorum, failures do not interfere with each other. As the Fig .11b suggests, certain IoT devices stay safe under SQuBA even if a significant number of edges are compromised vs. all other baselines that fail completely.

VI. Discussion

SQuBA simultaneously achieves good resilience to faulty edges and near real-time end-to-end latency.To the best of our knowledge, this is the first work that deploys social relationships in IoT access control in an open untrusted environment. As a consequence, our work is able to accommodate real-world trustworthiness in the IoT environment.

1) Social Relationships in IoT: Social relationships brings more challenges into the IoT environment [31], [32]. On one side, it amplifies internal threats of privilege-abuse users and complicates the rules and vocabulary of IoT privacy [9]. On the other side, it creates opportunities to develop a new access control mechanism that takes advantage of social ties. Our work distinguishes the intimate and indirect social ties and constructs local quorums accordingly. However, there are other complex relationships that may also make a difference in the IoT context, like neighbors and visiting families [9]. Properly including more types of social ties remains an open challenge.

2) Local Consensus for Privacy: Our experimental results inspire us to rethink the consensus-based approaches to IoT privacy. In our experiments, all types of local quorums based on users' social ties perform better than the system-wide quorums. It suggests that aggressively expanding the consensus quorum does not guarantee better privacy but introduces the risk of involving malicious entities. Figuring out the trade-off between local quorum size and performance demands more effort.

VII. Conclusion

With the rapid development of IoT networks, this work focuses on access control for IoT devices in multi-user environments. We introduce SQuBA into an edge-based IoT infrastructure with the key insight that authorization should consider users' real-world social relationships. Specifically, we propose the tiered local quorum structure and authorization voting algorithm to validate access requests using quorum edges. We evaluated our architecture using a real-world IoT dataset. We assess configuration factors, latency, and resilience to faulty edges. Our experimental results show promising near the real-time performance and good privacy guarantees that perform better than state-of-the-art approaches.

Acknowledgment

This work is supported in part by NSF grant NSF CNS-1908566.

REFERENCES

[1] J. Leng, Z. Lin, and P. Wang, "An implementation of an internet of things system for smart hospitals," IEEE/ACM Fifth International Conference on Internet-of-Things Design and Implementation (IoTDI), pp. 254-255, 2020.

[2] S. Li, et al., "Mf-iot: A mobilityfirst-based internet of things architecture with global reach-ability and communication diversity," IEEE First International Conference on Internet-of-Things Design and Implementation (IoTDI), pp. 129-140, 2016.

[3] S. Jog, et al., "Enabling IoT self-localization using ambient 5G signals," 19th USENIX Symposium on Networked Systems Design and Implementation (NSDI), pp. 1011-1026, 2022.

[4] D. Vasisht, et al., "FarmBeats: An IoT platform for data-driven agriculture," 14th USENIX Symposium on Networked Systems Design and Implementation (NSDI), pp. 515-529, 2017.

[5] M. Symeonides, et al., "5g-slicer: An emulator for mobile IoT applications deployed over 5g network slices," IEEE Seventh International Conference on Internet-of-Things Design and Implementation (IoTDI), pp. 115-127, 2022

[6] E. Fernandes, A. Rahmati, K. Eykholt, and A. Prakash, "Internet of things security research: A rehash of old ideas or new intellectual challenges?" IEEE Symposium on Security and Privacy, vol. 15(4), pp. 79-84, 2017.

[7] M. Antonakakis, "Understanding the mirai botnet," 26th USENIX Security Symposium, pp. 1093-1110, 2017.

[8] E. Fernandes, J. Jung, and A. Prakash, "Security analysis of emerging smart home applications," IEEE Symposium on Security and Privacy, pp. 636-654, 2016.

[9] W. He, et al., "Rethinking access control and authentication for the home internet of things (IoT)," 27th USENIX Security Symposium, pp. 255-272, 2018.

[10] Z. Leidall, A. Chandra, and J. Weissman, "An edge-based framework for cooperation in internet of things applications," 2nd USENIX Workshop on Hot Topics in Edge Computing, 2019.

[11] G. Yuan, D. Mazières, and M. Zaharia, "Extricating IoT Devices from Vendor Infrastructure with Karl," arXiv:2204.13737, 2022.

[12] N. Davies, N. Taft, M. Satyanarayanan, S. Clinch, and B. Amos, "Privacy mediators: Helping iot cross the chasm," 17th International Workshop on Mobile Computing Systems and Applications, pp. 39-44, 2016.

[13] M. Novotny and F. Zavoral, "BubbleTrust: a reliable trust management for large P2P networks," Recent Trends in Network Security and Applications: Third International Conference, Springer Berlin Heidelberg, pp. 359-373, 2010.

[14] M. Surbatovich, J. Aljuraidan, L. Bauer, A. Das, and L. Jia, "Some recipes can do more than spoil your appetite: Analyzing the security and privacy risks of IFTTT recipes," 26th International Conference on World Wide Web, pp. 1501-1510, 2017.

[15] Q. Wang, W. Hassan, A. Bates, and C. Gunter, "Fear and logging in the internet of things," Network and Distributed Systems Symposium, 2018.

[16] D. Johnson, A. Menezes, and S. Vanstone. "The elliptic curve digital signature algorithm (ECDSA)." International journal of information security 1, pp. 36-63, 2001.

[17] Y. Tian, et al., "SmartAuth: User-centered authorization for the Internet of Things," 26th USENIX Security Symposium, pp. 361-378, 2017.

[18] M. Ali, et al., "Real-time data analytics and event detection for IoT-enabled communication systems," Journal of Web Semantics, Elsevier, vol. 42, pp. 19-37, 2017.

[19] K. Hsu, K. Bhardwaj, and A. Gavrilovska, "Couper: Dnn model slicing for visual analytics containers at the edge," 4th ACM/IEEE Symposium on Edge Computing, pp. 179-194, 2019.

[20] J. Bonneau, C. Herley, P. Oorschot, and F. Stajano, "The quest to replace Passwords: A framework for comparative evaluation of web authentication schemes," IEEE Symposium on Security and Privacy, pp. 553-567, 2012.

[21] J. Gubbi, et al. "Internet of Things (IoT): A vision, architectural elements, and future directions." Future generation computer systems, vol. 29(7),pp. 1645-1660, 2013

[22] M. Islam and S. Kundu, "Preserving IoT privacy in sharing economy via smart contract", IEEE/ACM Third International Conference on Internet-of-Things Design and Implementation (IoTDI), pp. 296-297, 2018.

[23] S. Schechter, "The user is the enemy, and (s) he keeps reaching for that bright shiny power button", Workshop on Home Usable Privacy and Security (HUPS), 2013.

[24] T. Matthews, et al., "Stories from survivors: Privacy & security practices when coping with intimate partner abuse," CHI Conference on Human Factors in Computing Systems, pp. 2189-2201, 2017.

[25] B. Ur, J. Jung, S. Schechter, "Intruders versus intrusiveness: teens' and parents' perspectives on home-entryway surveillance," ACM International Joint Conference on Pervasive and Ubiquitous Computing, pp.129-139, 2014.

[26] K. Zhang, et al., "Security and privacy in smart city applications: Challenges and solutions," IEEE Communications Magazine, vol. 55(1), pp. 122-129, 2017.

[27] D. Eckhoff and I. Wagner, "Privacy in the smart city—applications, technologies, challenges, and solutions," IEEE Communications Surveys & Tutorials, vol. 20(1), pp. 489-516, 2017.

[28] "HomeKit," Apple Developer. https://developer.apple.com/homekit/. [Accessed Feb. 15, 2023].

[29] "Google iot solutions," Google Developer. https://developers.google.com/iot. [Accessed Feb. 18, 2023].

[30] C. Marche, L. Atzori, M. Nitti, "A dataset for performance analysis of the social internet of things," IEEE 29th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC), pp. 1-5, 2018.

[31] P. Naeini, et al., "Privacy expectations and preferences in an IoT world," Thirteenth Symposium on Usable Privacy and Security (SOUPS), pp. 399-412, 2017.

[32] E. Zeng, S. Mare, and F. Roesner, "End user security and privacy concerns with smart homes," Thirteenth Symposium on Usable Privacy and Security (SOUPS), pp. 65-80, 2017.

[33] A. Pouraghily, M. Islam, S. Kundu, and T. Wolf, "Privacy in blockchain-enabled iot devices", IEEE/ACM Third International Conference on Internet-of-Things Design and Implementation (IoTDI), pp. 292-293, 2018.

[34] B. Carminati and E. Ferrari. "Collaborative access control in on-line social networks." 7th international conference on collaborative computing: Networking, applications and worksharing, 2011.

[35] Q. Ni, et al., "Privacy-aware role-based access control," ACM Transactions on Information and System Security (TISSEC), vol. 13(3), pp. 1-31, 2010.

[36] F. Paci, A. Squicciarini, and N. Zannone. "Survey on access control for community-centered collaborative systems," ACM Computing Surveys (CSUR), vol. 51(1), pp. 1-38, 2018.

[37] D. Kulkarni and A. Tripathi, "Context-aware role-based access control in pervasive computing systems," 13th ACM symposium on Access control models and technologies, pp. 113-122, 2008.

[38] B. Zhang, et al., "The cloud is not enough: Saving iot from the cloud," USENIX Workshop on Hot Topics in Cloud Computing (HotCloud), 2015

[39] G. Petracca, L. Marvel, A. Swami, and T. Jaeger, "Agility maneuvers to mitigate inference attacks on sensed location data," IEEE Military Communications Conference, pp.259-264, 2016.

[40] X. Liu, B. Farahani, and F. Firouzi. "Distributed ledger technology." Intelligent Internet of Things: From Device to Fog and Cloud, pp. 393-431, 2020.

[41] "Multicast", Wikipedia. https://en.wikipedia.org/wiki/Multicast. [Accessed Feb. 15, 2022]

[42] S. Basu and E. Sirer, "Trustless IoT: A logic-driven architecture for IoT hubs," USENIX Workshop on Hot Topics in Edge Computing, 2020.

[43] G. Wood, et al., "Ethereum: A secure decentralised generalised transaction ledger," Ethereum project yellow paper, vol. 151, pp. 1-32, 2014.

[44] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," Decentralized Business Review, pp. 21260, 2008.

[45] O. Novo, "Blockchain meets IoT: An architecture for scalable access management in IoT," IEEE Internet of Things Journal, vol. 5(2), pp. 1184-1195, 2018.

[46] Y. Zhang, S. Kasahara, Y. Shen, X. Jiang, and J. Wan, "Smart contract-based access control for the internet of things", vol. 6(2), pp. 1594-1605, 2018.

[47] A. Dorri, S. Kanhere, and R. Jurdak, "Towards an optimized blockchain for IoT," IEEE/ACM Second International Conference on Internet-of-Things Design and Implementation (IoTDI), pp. 173-178, 2017.

[48] D. Mazières, “The stellar consensus protocol: A federated model for internet-level consensus,” Stellar Development Foundation, vol. 32, 2015.
[49] G. Losa, E. Gafni, and D. Mazières, “Stellar consensus by instantiation,” International Symposium on Distributed Computing (DISC), 2019
[50] G. Losa and M. Dodds, “On the formal verification of the Stellar consensus protocol”, 2nd Workshop on Formal Methods for Blockchains, 2020.
[51] M. Lokhava, et al., “Fast and secure global payments with Stellar,” 27th ACM Symposium on Operating Systems Principles, pp. 80-96, 2019.
[52] A. Clement, et al. ”Making Byzantine fault tolerant systems tolerate Byzantine faults.” 6th USENIX symposium on Networked systems design and implementation. The USENIX Association, 2009.
[53] “Two-phase commit protocol”, Wikipedia. https://en.wikipedia.org/w/index.php?title=Two-phase_commit_protocol&oldid=1078983413. [Accessed Aug. 28, 2022]
[54] M. Bermudez-Edo, et al. ”IoT-Lite: a lightweight semantic model for the internet of things and its use with dynamic semantics.” Personal and Ubiquitous Computing, pp. 475-487, 2017.
[55] S. Sayeed, and H. Marco-Gisbert. ”Assessing blockchain consensus and security mechanisms against the 51% attack.” Applied sciences vol. 9(9), pp. 1788, 2019.
[56] “Echo,” Amazon. https://www.amazon.com/echo. [Accessed Feb. 15, 2023].
[57] “Hue,” PHILIPS. https://www.meethue.com. [Accessed Feb. 15, 2023].
[58] “SmartThings,” SAMSUNG. https://www.smartthings.com. [Accessed Feb. 15, 2023].