# A fast and scalable authentication scheme in IOT for smart living

Jianhua Li [a],[*], Jiong Jin [b], Lingjuan Lyu [c],[*], Dong Yuan [d], Yingying Yang [e], Longxiang Gao [a], Chao Shen [f]

[a] School of Info Technology, Deakin University, Melbourne, VIC 3125, Australia
[b] School of Software and Electrical Engineering, Swinburne University of Technology, Melbourne, VIC 3122, Australia
[c] The National University of Singapore, Singapore
[d] School of Electrical and Information Engineering, The University of Sydney, Sydney, NSW 2006, Australia
[e] School of Electrical and Data Engineering, The University of Technology Sydney, Sydney, NSW 2007, Australia
[f] School of Electronic and Information Engineering, Xi'an Jiaotong University Xi'an, SHX 710049, China

## ARTICLE INFO

## ABSTRACT

Numerous resource-limited smart objects (SOs) such as sensors and actuators have been widely deployed in smart environments, opening new attack surfaces to intruders. The severe security flaw discourages the adoption of the Internet of things in smart living. In this paper, we leverage fog computing and microservice to push certificate authority (CA) functions to the proximity of data sources. Through which, we can minimize attack surfaces and authentication latency, and result in a fast and scalable scheme in authenticating a large volume of resource-limited devices. Then, we design lightweight protocols to implement the scheme, where both a high level of security and low computation workloads on SO (no bilinear pairing requirement on the client-side) is accomplished. Evaluations demonstrate the efficiency and effectiveness of our scheme in handling authentication and registration for a large number of nodes, meanwhile protecting them against various threats to smart living. Finally, we showcase the success of computing intelligence movement towards data sources in handling complicated services.

© 2020 Published by Elsevier B.V.

## 1. Introduction

The Internet of things (IoT) has changed the world in many aspects of our living. For example, a user may install a Google smart speaker for less than $100, performing voice recognition, information retrieval, and environment control [1]. IoT-enabled applications such as healthcare and energy have already stepped into smart living [2], bringing a variety of benefits including automation, efficiency, and low cost. Unfortunately, such applications are forming isolated information silos without enough cross-application communication and inter-cloud collaboration. With significant advantages of low latency, situational awareness, low cost, mobility support, and widespread coverage [3], fog computing (fog) facilitates communication and collaboration among pervasive and ubiquitous things. As shown in Fig. 1, fog is deployed in multiple tiers along the cloud–fog–things continuum, delivering substantial intelligence to where it is needed. However, widespread cyberthreats discourage people from further adoption of IoT devices in smart living [4].

The current combination of cloud and IoT is a game-changer technology that addresses many problems, although it is associated with considerable security deficiencies. Without mature legislation in the IoT market, profit-driven vendors often prioritize functionality over security in designing a product to save time and money, opening numerous attack surfaces. According to the U.S. Department of Justice, there are more than 4000 ransomware attacks each day on average since 2016. With little or no effort, an attacker can hack into home cameras to watch anything within the scope and capture videos and pictures [5]. The deployment of such security-flaw devices will cause catastrophic disasters in many aspects.

It is compulsory to recognize all vulnerabilities and then implement reliable solutions to minimize potential risks and achieve end-to-end security [6]. Such solutions may involve computation-intensive workloads that are not feasible on many resource-limited SOs. Indeed, they have the same security requirements as powerful devices because any compromised device may potentially undermine the entire security pillar. Therefore, numerous efforts have been made in addressing the above challenges by using various technologies [7–13].
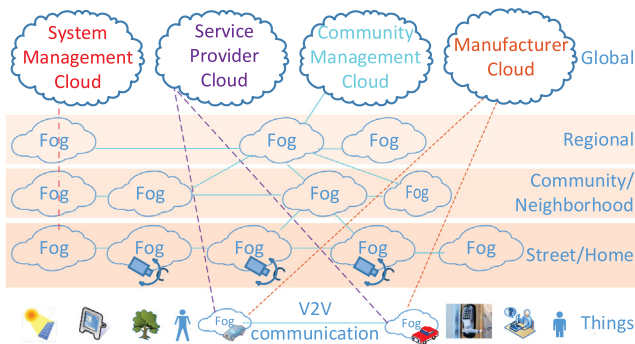
**Fig. 1.** The cloud–fog–things continuum for smart living.

It is worth noting that security starts with reliable authentication to ensure the identification legitimacy of participating parties, which is a prerequisite for other security services like authorization and access control [14]. In smart living, users (via an endpoint) and things must be mutually authenticated for secure communication, preventing it from executing a malicious process, data leakage, and privacy breach [15]. As reviewed in [16,17], people have developed a diversity of standards and protocols in this direction.

Due to a lack of a uniformed industrial standard for security among heterogeneous IoT nodes, the public-key-infrastructure (PKI) tends to be employed to provide authentication services [18], where an entity owns a pair of public and private keys. In this system, a public key is presented as a form of a digital certificate to integrate its identity, while a private key manifests the ownership of the public key. Hence, the key pair of each entity must be effectively generated, managed, stored, and accessed through PKI. In this regard, a certificate authority (CA) is a centralized and trusted component to issue, store, and revoke keys if required.

Admittedly, the CA-based PKI works on many conditions, including robust endpoints, stable connections, and centralized management, but heterogeneous and ubiquitous IoT endpoints [19] may invalidate such assumptions in many ways. First, asymmetric cryptography is impractical to use with numerous sensors due to resource limitations. Furthermore, centralized CA needs to maintain a big database of keys material for each entity, which causes long delays in the keys acquisition, authentication failure, and even more severe problems [20]. Even worse, the long-distance and big-volume data movements from ubiquitous things to the remote cloud provide excessive attack surfaces for crackers [21], deteriorating the security situation in all aspects [22]. For these reasons, we consider moving the security computation closer to IoT endpoints, thus making improvements to address such issues.

Therefore, we are motivated to design a novel scheme called "CA as a Service" (CAaaS) by following the microservice architecture [23] and the virtual fog framework proposed in [24]. In this scheme, we customize CA functionality for the IoT and push it closer to IoT to cope with the above concerns. The contributions are fivefold.

- Initially, we design a microservice-enabled and fog-based CA scheme addressing both strong security and low latency.
- Secondly, we develop lightweight protocols to validate our scheme, where there is no need to run complicated asymmetric cryptography algorithm on data-source devices.
- Thirdly, we facilitate the realizing of short-lived key pairs to reduce the burden of CA servers and enhance the device-to-device security.

- Meanwhile, we propose a situation-awareness approach to identify compromise in IoT nodes and then explore cognitive measures to minimize attack surfaces.
- Eventually, we showcase the success of the computation movement from centralized servers to network edges.

The road-map of this paper is as follows. In the next section, we present an overview of our proposed scheme. We elaborate on its implementation in Section 3, followed by the experimental evaluation in Section 4, which showcases low latency and high scalability. In Section 5, we discuss the protection intelligence against diverse attacks, accompanied by a formal verification with AVISPA to prove the effectiveness. In Section 6, we present the related work. Furthermore, Section 7 pinpoints limitations and future directions, while Section 8 concludes the paper. Table 1 presents the abbreviations and symbols used in this paper for better readability.

## 2. The mutual authentication scheme overview

In the first place, we introduce the virtual fog (VFOG) for IoT. Next, we inaugurate the scheme in the framework.

### 2.1. The VFOG framework for IoT

Fueled by virtualization technologies encompassing service virtualization, network function virtualization [25,26], and object virtualization, the VFOG enables the fog's realization as investigated in [24], underpinning the majority of beneficial features. This framework includes three layers, i.e., a service virtualization layer, a network function virtualization layer, and an object virtualization layer from top to bottom, along the cloud-to-things continuum.

In the service virtualization layer of the framework, through managing and coordinating *microservice blocks (MSB)*, the service virtualization manager (SVM) empowers service orchestration, service monitoring, and service optimization based on MSBs, running on a pervasive FN. The MSB is a micro-service fragmenting application previously found on large servers [27]. Section 2.4.2 reviews more details regarding MSB components.

In the middle, the network function virtualization layer decouples software network functions from vendor-specific equipment in the style of virtual network functions (VNFs). Such VNFs are immediately instantiated on an FN to facilitate networking functions among entities. Coupled with software-defined networking (SDN) technologies, the network function virtualization manager (NFVM) provisions, manages, coordinates, monitors VNFs, and its underlying infrastructure, enforcing resiliency and security.

At the bottom object virtualization layer, a virtual object (VO), hosted on an FN, seamlessly integrates an IP-enabled and sensor-specific network domain. On the one hand, it enhances the capability of an SO by taking over some resource-intensive workloads such as security. On the other hand, it manages and controls SOs, therefore further augmenting security with cognitive working intentions.

In the overarching VFOG framework, a lightweight middleware called foglets is responsible for the interplay between FN and clients. Apart from this, the VFOG employs foglets to manage the network infrastructure and empower user interface at endpoints, grounded with local situational awareness.

Most interestingly, the VFOG adds more benefits in the smart living transition, that is, transforming traditional devices and existing applications smoothly into a future environment. In consequence, the adoption of this framework makes smart living more finance-viable and affordable. To put it differently, the VFOG could protect investments made previously in an evolutionary process, although it is indeed a revolutionary reconstruction.

**Table 1**
Abbreviation and symbol used in this paper.

| Abbreviation/Symbol | Explanation |
| --- | --- |
| AI | Artificial intelligence |
| CA | Certificate authority |
| CAaaS | Certificate authority as a service |
| CRL | Certificate revocation list |
| ECC | Elliptic curve cryptosystem |
| FN | Fog node |
| MFA | Multi-factor authentication |
| ML | Machine learning |
| MSB | Microservice block |
| NFVM | Network function virtualization manager |
| PKI | Public key infrastructure |
| SDN | Software-defined networking |
| SO | Smart object |
| SVM | Service virtualization manager |
| VNF | Virtual network functions |
| VO | Virtual object |
| $\times$ | Multiply, encryption, decryption |
| $A_{ID_C}$ | An ID-based authentication key for client C |
| $C, C_i, C_j$ | A client or child |
| $ID_C$ | An ID of client C |
| $E_i, E_j, E_{k_{ij}}, E_N$ | An encrypted message |
| $IVV, IVV*$ | An integrity verification value |
| $k, k_{ij}, k_{iS}, k_{jS}$ | A symmetric session k |
| $K_S^R$ | A private key of CA server |
| $K_S^U$ | A public key of CA server |
| $M_C$ | A temporary message based on ECC for a client |
| $M_k, M'_k$ | A temporary message based on a session key |
| $M_S$ | A temporary message based on ECC for a client |
| $N, N_i, N_j, N'_j$ | A nonce |
| $P, Q$ | An initialized starting point on an ECC curve |
| $Q_{ID_C}$ | Calculated coordinates of point Q from P on an ECC curve |
| $R_C, R_C^*, R'_C, R''_C$ | The coordinates of client C on an ECC curve with a different timestamp |
| $R_S, R'_S$ | The coordinates of server S on an ECC curve with a different timestamp |
| $S$ | A server or a device offers CA services |
| $T_1, T'_1, t_1, T_2$ | A different timestamp for that moment when an event starts |
| $x_C, x'_C$ | The x coordinate of client C on the ECC curve |
| $x_S$ | The x coordinate of server S on the ECC curve |

Despite the above benefits, this framework brings numerous advantages like low latency and cost. Security is also enhanced grounded on at least two facts initiated by VO, namely, heavy-weight security workloads offloaded from an SO, and flexible management imposed on an SO with local awareness of working context. To this end, our scheme concentrates on security improvement in alignment with the two facts.

It is also worth noting the VFOG enables the realization of cloud-to-things continuum systematically with increased situational awareness along the continuum. Before reviewing our proposed authentication scheme, we first present the problem description and threat model as follows.

### 2.2. Problem description

Although the VFOG framework is a key to the fulfillment of the cloud-fog-things continuum [24], the IoT security is still a challenge before practical adoption. Therefore, our primary goal is to explore an effective (i.e., against various attacks) and efficient (i.e., low-latency and scalable) authentication scheme, particularly suitable for resource-constrained devices.

Manufacturer clouds store detailed data of their products, including serial numbers, firmware, ID, software, embedded security feature, and other detailed specifications. Vendors conduct their business based on such critical data, supplying devices and applications to the market. At this moment, users order devices and services provided along the cloud–fog–things continuum. To generalize our description, we define fog as the *Custodian* that has the responsibility to protect things (sensors, actuators, endpoints, etc.). Comparatively, a manufacturer cloud is the *Parent*, while things are the *Child*. Again, manufacturers have detailed

knowledge of their products, which is the *Parent–Child affinity*. Empowered by such affinities, a *Custodian* accomplishes situational protection for the IoT with local security awareness and working intention awareness.

As illustrated in Fig. 2, a *Child* is adopted in a cloud–fog–things continuum. From the perspective of security, it is feasible to achieve data security between *Parent* and *Custodian* (green flow) by collaboratively using advanced technologies. It is, nevertheless, inappropriate to a *Child* due to various constraints, including energy, memory, computational speed, and network capability. Therefore, we focus on the authentication service provisioning for things (red flow). To this end, there are two questions raised, i.e., what scheme is suitable for the resource-constrained environment and how to implement it. In this paper, we first explore mutual authentication and symmetric key agreement between *Child* and *Custodian*. Then, we study inter-*Child* authentication and key management via a trusted *Custodian*.

### 2.3. Threat model

As a general rule, a device in the cloud is securely protected rather than a fog node on-premises [28]. Due to actual circumstances (computing power, hardware, and software feature, deployment surroundings, business purpose, etc.), some FNs can take benefit from cutting-edge technological combinations for security. As such, it is acceptable to assume that some FNs are securely protected, while other FNs and IoT nodes are not in the continuum. In smart living, insecure nodes are susceptible targets to many cyber–physical attacks, including physical attacks (modifying a node with untrusted hardware/software components), passive attacks (eavesdropping and traffic analysis), replay attacks (pretending to be a legitimate user by capturing some valid
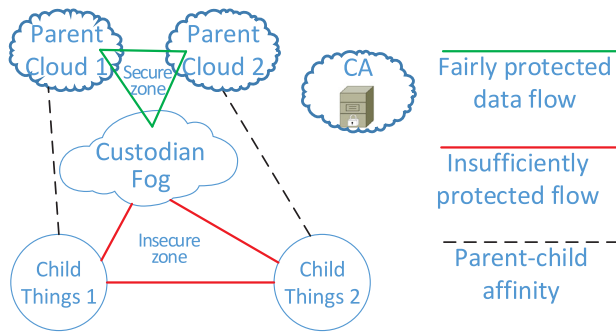
**Fig. 2.** The secure and insecure area: There are two concerns of authentication raised for the insecure zone, namely, what scheme performs well and how to implement it?

data), and impersonation attacks (assuming the legal identity of an alternative communication participant).

In theory, CA distributes key services from a centralized server to CA proxies in a *Custodian* device along with the cloud–fog–things continuum. A secure FN can be a *Custodian* node, which forms a secure black box with cloud servers, referring to the secure zone triangle (the green triangle) in Fig. 2. A *Child* needs to register to a *Custodian*. During this process, a *Custodian* performs an integrity check, authentication, and session key negotiation for a *Child*. After this, it serves as a trusted point for inter-*Child* authentication. We assume that full-fledged security technologies fairly guarantee data security on the *Custodian* device.

Conversely, a *Child* such as a non-*Custodian* FNs or an IoT endpoint locates in an insecure zone (the red triangle in Fig. 2), which is hackable in terms of software, firmware, hardware, and communication. An intruder can eavesdrop data between *Child* and *Custodian*, manipulate data as wanted (e.g., delete, inject, delay, duplicate, replay). It even can acquire legitimate session keys with a *Custodian* to launch further attacks against data-in-transit among other devices in this zone.

### 2.4. CA as a service (CAaaS)

Technically, the most critical role of a CA is to serve as a trust anchor in the IoT ecosystem through which certificates are issued, distributed, and revoked. In this section, we briefly explain the cloud–fog–things continuum to contextualize our scheme.

#### 2.4.1. Cloud–fog–things continuum

According to the OpenFog Consortium, fog distributes resources and services of computing, storage, control, and networking anywhere along the cloud–fog–things continuum [24]. As shown in Fig. 1, fog computing provides computing intelligence at a hierarchical location of regional, community/neighborhood, street, and home, with the cognition of working conditions. It is thus equitable to deploy such resources and services to where they are most in need. From the data's point of view, data are collected, filtered, aggregated, stored, and processed with appropriate resources and services within proximity of data source, which significantly reduces communication cost [29] and alleviates bandwidth bottleneck.

The VFOG framework makes use of the technological landscape of cloud computing to make up the cloud–fog–things continuum. Relevant technologies incorporate resource pooling, resource management, service provisioning, multitenancy, load balancing, distributed storage, middleware, API, web service, and virtualization. Recent technological advancements in SDN, big data [30], deep learning [31], artificial intelligence (AI), docker,

microservice are also instrumental in bringing benefits such as low latency, high efficiency, scalability, and security [32–34].

As we focus on authentication in this paper, a detailed survey on such a technological integration is beyond the scope. Next, we explore MSBs to scrutinize CA components in the scheme.

#### 2.4.2. Microservice block (MSB)

As the trust anchor, a CA plays critical roles in policy authority, certificate issuer, certificate generator, revocation manager, registration authority, repository, and authentication officer. A CA runs on a powerful server, traditionally, sitting in a well-protected cloud data center. However, it is hard for mobile devices to detect such a CA server in a dynamic IoT ecosystem. For example, mobile nodes like driving cars would likely incur quick network topology change, while a blind effort to increase the numbers of Child CAs easily enkindles a security breach.

As demonstrated in [35], it is achievable to break down large programs into small slices [36], based on which we can perform the same function, in distributed nodes. We, therefore, take the advantage to decompose previous large programs into current MSBs. An MSB is a segmented software instance bearing discrete, autonomous, and network-accessible services. Each MSB is loosely-coupled and highly cohesive, which independently deploys on a widespread fog hardware platform. Such an MSB is a single application in light of working intentions.

A CA server in a cloud is well-protected by cutting edge security implementations in all aspects (logically, electronically, physically, etc.). In our scheme, we employ MSBs to bear the functions of a traditional CA, including generation, storage, and distribution of keys, certificate revocation list (CRL) services, CA proxy services, and more. Most importantly, evaluation results demonstrate that our scheme strengthens the security of CA. Next, we scrutinize the proposed CAaaS scheme composed of MSBs along with the cloud–fog–things continuum.

#### 2.4.3. CAaaS scheme

In brief, our CAaaS scheme has outstanding performance, resulting from the deployment of corresponding MSBs with situational awareness. A *service virtualization manager (SVM)* manages such MSBs dynamically according to the actual authentication needs [24]. To clarify, an SVM initializes, monitors, coordinates, and controls MSBs in the continuum to facilitate the provisioning of authentication services. Note that an MSB may or may not necessarily exist permanently at a particular *Custodian* FN. Consequently, the local profile-based scheme provides unique benefits in authentication service provisioning.

As shown in Fig. 3, the VFOG distributes CA MSBs to capable FNs along the cloud–fog–things continuum. Such an FN is a *Custodian*, bearing CA services and protecting nearby IoT components. Under the coordination of an SVM, these MSBs make up needed functions at the proximity of data source in fog [37], which improves the availability, efficiency, and scalability of CA [38]. To demonstrate, $FN_1$ performs the authentication between $SO_{11}$, $SO_{12}$, and user $U_1$.

Provided that two entities are accessing the fog via different locations, for example, user $U_1$ and $U_2$, the positioning of authentications becomes non-trivial. Based on the latency estimation service provided by fog [39], a node with the least overall latency, such as $FN_2$, may fulfill the authentication between the two. From the perspective of FN's deployment, we dispatch keys services as well as CRL services on upper layers, keys distribution services in middle layers, and CA proxy services at lower tiers. In effect, the reduced distance diminishes latency and risk of data movements, fostering security, and scalability in the IoT ecosystem.

To further validate our scheme, We develop four protocols in the implementation. In the system configuration phase (detailed
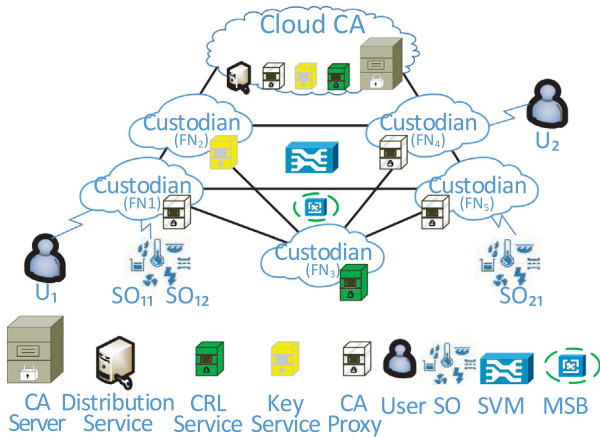
**Fig. 3.** An overview of our proposed CAaaS scheme.

in Protocol 1 in Section 3.3), the CA generates its private and public key pair and publishes its public key to all networked nodes, getting ready for *Child* registration. During the registration and authentication process (performed with Protocol 2), CA generates an ID-based authentication key and passes to the *Child* in a secure channel. Simultaneously, a *Custodian* completes device integrity verification (achieved in Section 3.2), taking advantage of the *Parent–Child affinity*. Then, the *Child* can use the authentication key for mutual authentication and key agreement with a *Custodian* (accomplished with Protocol 3). After *Custodian-Child* mutual-trust is established, the *Custodian* is further used as a trust point for inter-*Child* authentication and key management (fulfilled in Protocol 4).

It is also worth mentioning that some *Custodian* may be able to issue short-lived keys for fast-moving things, e.g., driving cars. Location-based and situation-awareness authentication may be required to get local service, while the temporary key expires when the vehicle leaves the area. In this scenario, the *CRL* service would be kept busy without the support of a short-lived key. In more detail, a CRL is an array of certificates that have been marked as "no longer trusted" before their scheduled expiration.

## 3. The scheme implementation

In this section, we review the preliminary of ECC cryptography, device integrity check, and proposed protocols.

### 3.1. The preliminary

An elliptic curve $E_p(a, b)$ over a Galois field $F_p$, where $p > 3$ and is prime, is the set of all $(x, y)$ $(x, y \in F_p)$ that satisfies the cubic equation:

$$y^2 = x^3 + ax + b \tag{1}$$

where $a, b \in F_p$, and $4a^3 + 27b^2 \neq 0$.

The points on the curve with an extra point $O$ called the point at infinity, form a group

$$G = \{(x, y) : x, y \in F_p, E_p(x, y) = 0\} \cup \{O\} \tag{2}$$

$G$ is a cyclic additive group defined as below:

Let $P, Q \in G$, a line $l$ containing $P$ and $Q$ (tangent line to $E$ if $P = Q$), meets $E$ at $R$. Let $l'$ be a line linking $R$ and $O$. Then, $P$ "+" $Q$ is the point where $l'$ intersects $E$ at $R$ and $O$, and $P$ "+" $Q$. Scalar multiplication over $E_p$ can be calculated as

$$s \times P = P + P + \cdots + P \text{ (s times)} \tag{3}$$

where $s \in F_p*$ is an integer, and a point $P \in E_p(a, b)$. In general, the security of ECC depends on the following difficulties:

1. Discrete logarithm problem (DLP): given $G$'s generator $P$ and $Q = x \times P$ to calculate x.
2. Computational Diffie–Hellman problem (CDHP): given $G$'s generator $P$, $a \times P$, and $b \times P$ to compute $ab \times P$.
3. Elliptic curve factorization problem (ECFP): given two points $P$ and $Q = s \times P + t \times P$ over $E_p(a, b)$ for $s, t \in F_p^*$, to locate two points $s \times P$ and $t \times P$ over $E_p(a, b)$.

### 3.2. Device integrity verification

Working in an insecure zone, a *Child* offers numerous attack surfaces, including processors, function-specific modules, operating systems, and software utilities. Thanks to fog enabling the situational awareness of manufacturers, service providers, users, and working surroundings where an SO operates, the VO performs vulnerability assessment and mitigation, based on available contexts. To illustrate, hackers can frequently scan and exploit system vulnerabilities to install backdoors, sniffers, data collection software, and even remotely control a victim to launch malicious attacks. Fortunately, a *Custodian* can determine if a *Child* integrity is compromised, through *Parent–Child affinity*, and take reasonable actions surveyed in Table 2, therefore minimizing threats.

Following the above action, the VO calculates the *integrity verification value (IVV)* to verify whether there is any modification on a *Child*. The parameter contributing to this value may include ID, firmware, OS, running software, used slots, unused slots, blacklisting services, and more. Meanwhile, the *Custodian* computes it ($IVV^*$) based on the parameter obtained from the affinity. If the two values match, there is no modification; otherwise, the *Custodian* may reset the device or conduct device quarantine and inform other nodes of related security risks. As such, the system maintains consistent local security awareness.

In more detail, a *Child* device (user devices, sensors, etc.) needs to register to fog before performing its business functions. During the registration process, the *Custodian* conducts verification and device hardening. A device may have preloaded security features such as encryption algorithms and pre-shared keys, through which establishing an exclusively confidential communication channel to transmit ID-based authentication keys. This isolation prevents an ID-based authentication key from being leaked.

It is equally important to point out that device update (namely, software patches, firmware update, components upgrade, etc.) is compatible with the device integrity verification. Fog actively collects such updates from various hardware manufacturers and software developers to maintain its currency in the IoT ecosystem. In case that a *Child* device is hard to upgrade directly from its *Parent* network due to diverse reasons (unstable connection, working surroundings, business requirement, etc.), fog may serve as an agent to perform such updates. Next, we investigate a series of protocols for the scheme implementation.

### 3.3. The protocol

Grounded in the device integrity verification and device hardening, our proposed protocol achieves end-to-end authentication for IoT devices in the continuum. In brief, Protocol 1 initializes the system; Protocol 2 performs registration to CA; Protocol 3 conducts mutual authentication, generating a session key between user/device and CA, while Protocol 4 carries out the endpoint to endpoint authentication. To simplify the description, we name CA

**Table 2**
Possible attacks and countermeasures.

| Attack surface | Possible attacks | Possible consequence | Counter-measures |
|---|---|---|---|
| Processors (CPU, GPU, etc.) | Swap processors with flawed ones | Device damage, backdoors | Serial number check, device quarantine, trustworthy downgrade |
| ROM, EPROM, functionality-specific module (e.g., sensing, crypto module) | Utilization of compromised firmware, default access | Device malfunction, unwanted functions, device ID, security certificate attacks, undesired access through default settings, compromised cryptosystem, etc. | Firmware content check, device quarantine, firmware recovery, upgrade, patches, ID and certificate attestation |
| External memory | Malicious data, Malware, e.g., Trojan | Comprehensive security breach, e.g., device malfunction, network performance degradation | Device hardening, malicious code and behavior detection, whitelisting, device quarantine, sandboxing, etc. |
| Input/output interfaces | Untrusted data or device connection | Attacks gaining access to IoT device | Unneeded interfaces shutdown, disconnection/connection monitoring |
| Transceivers/Network adapters | Fake node attack, routing attack, etc. | Fake node or data inserted to interfere normal business, energy wasting, undesired routing behavior, DoS attack | Device whitelisting/blacklisting, host attestation, device quarantine |
| Embed expansion slot | Untrusted card installation | Unwanted functionality | Redundant slot deactivation, card verification |
| Operating system (OS) | OS and runtime compromising, malicious coding, etc. | OS crash or lockup, compromised cryptosystem, runtime, etc. | Sandboxing, hardening, lockdown, whitelisting, patches, reset |
| Software utilities | Bugs, backdoors exploitation | Unauthorized device access, resource utilization, compromised cryptosystem, device ID and security certificate modification | Sandboxing, hardening, lockdown, whitelisting, blacklisting |
| Data | Data-in-transit attack, data-in-rest attack, data collection from rejected devices | Password, and other sensitive information loss, privacy leakage, secret loss, data loss, finance loss, reputation loss, etc. | Using protected connection (e.g, virtual private network), user track, cryptosystem, ethical isolation, pentest, etc. |
| User | Social engineering, malicious actors | Sensitive information loss, network misconfiguration, unauthorized access to device, etc. | Enabling situational awareness using a combination of technologies such as big data, machine learning, pentest, etc. |
| Other physical attacks | EMI, RFI attacks, property vandalism, etc. | Losses of device, and connection, etc. | Out of the scope |

and its distributed service in the secure zone as server $S$ and an outside node as client $C$.

---

**Protocol 1** System settings and announcement

1: $S$: Chooses an elliptic curve $E_p(a, b)$ over a large Galois field F and a base point $P$, on that curve as described in Section 3.1. After this, it derives its private/public key pair $(K_S^R, K_S^U)$, where $K_S^U = K_S^R \times P$.

2: $S$: Selects three secure hash functions:
   $H_1(\cdot) : \{0, 1\} \rightarrow G_p$
   $H_2(\cdot) : \{0, 1\} \rightarrow Z_p^*$
   $H_3(\cdot) : \{0, 1\}^* \rightarrow Z_p^*$
   where $G_p$ is generated by $P$ over $E_p(a, b)$ (referring to equation 2)

3: $S \rightarrow C$: Keeps $K_S^R$ in secret and broadcasts clients of $\{E_p(a, b), P, K_S^U, H_1(\cdot), H_2(\cdot), H_3(\cdot)\}$

4: $S$: An SVM creates MSBs for keys distribution service, certificate revocation list (CRL) service, keys computation, storage service, and CA proxy service, and deploys these services on FN as required.

---

The announcement made by $S$ goes along with the cloud–fog–things continuum. Finally, $C$ notices the availability of CA. In this case, $C$ treats CA proxy as $S$, as it is kept back from nodes out of the secure black box. Then, a client starts its registration process through Protocol 2.

---

**Protocol 2** *Child* registration to CA

1: $C \rightarrow S$: Sends $\{ID_C\}$ via CA proxy

2: $S$: Computes the authentication key $A_{ID_C}$ for $C$:
   $A_{ID_C} = K_S^R \times H_1(ID_C) \in G_p$

3: $S \rightarrow C$: Sends $\{A_{ID_C}\}$ via CA proxy in a confidential communication channel.

4: $C$: Computes $A_{ID_C} \times P = K_S^U \times H_1(ID_C)$.
   If true, keeps $A_{ID_C}$ in private.

---

After registration, the *Custodian* and *Child* will authenticate each other and share a session key, using Protocol 3.

Note that a client does not require bilinear-pairings, making the protocol more feasible on resource-constrained nodes. Grounded on the authentication and unique shared session keys, a *Child* can now establish respective session keys with other nodes via any mutually trusted *Custodian*, as demonstrated in Protocol 4.

In Protocol 4, $k_{iS}$ and $k_{jS}$ are symmetric session keys between client nodes $C_i$ and $S$, $C_j$ and $S$, respectively. Node $C_i$ and $C_j$ will be able to perform the negotiation of session keys via trusted $S$ and authenticate each other through a nonce $N$. When $C_i$ is going to communicate with $C_j$, node $C_i$ will initialize a session key $k_{ij}$, and send to $S$, where the key will be forwarded to $C_j$. After this,

**Protocol 3** *Custodian-Child* authentication

1: $C$: Randomly selects a point $R_C = (x_C, y_C) \in E_p(a, b)$, where $x_C$ and $y_C$ are coordinates of the point on x and y axis.
2: $C$: Generates a current timestamp $T_1$, and computes $t_1$, $M_C$, and $R_C^*$.
$$t_1 = H_2(T_1)$$
$$M_C = R_C + t_1 \times A_{ID_C}$$
$$R_C^* = X_C \times P$$
3: $C \rightarrow S$: Sends $\{ID_C, M_C, R_C^*, T_1\}$ via CA proxy.
4: $S$: Computes the coordinates of $Q_{ID_C}$ and $R_C'$.
$$Q_{ID_C} = H_1(ID_C) = (x_Q, y_Q)$$
$$t_1' = H_2(T_1)$$
$$R_C' = M_C - t_1' \times K_S^R \times Q_{ID_C} = (x_C', y_C')$$
5: $S$: Generates the local current timestamp $T_2$, calculates $\Delta T$.
$$\Delta T = T_2 - T_1$$
If $\Delta T$ is accepted, $S$ checks if $R_C^* = x_C' \times P$ holds.
If true, $S$ trusts the validity of $C$, and asserts $x_C' = x_C$.
Else, the protocol terminates.
6: $S$: Randomly chooses a point $R_S$:
$R_S = (x_S, y_S) \in E_p(a, b)$, and computes $t_2$ and $M_S$.
$$t_2 = H_2(T_2)$$
$$M_S = R_S + t_2 \times K_S^R \times Q_{ID_C}$$
And then calculates a session key $k$ and temporary message $M_k$.
$$k = H_3(x_Q, x_C, x_S)$$
$$M_k = (k + x_S) \times P$$
7: $S \rightarrow C$: Sends $\{M_S, M_k, T_2\}$ to C via the proxy.
8: $C$: Computes the coordinates of $Q_{ID_C}$ and $R_S'$.
$$Q_{ID_C} = H_1(ID_C) = (x_Q, y_Q)$$
$$t_2' = H_2(T_2)$$
$$R_S' = M_S - t_2' \times A_{ID_C} = (x_S', y_S')$$
9: $C$: Verifies the delay is allowed by comparing $T_2$ with the current time, if true, $C$ then computes $k'$ and $M_k'$
$$k' = H_3(x_Q, x_C, x_S')$$
$$M_k' = (k' + x_S') \times P$$
If $M_k' = M_k$, then $k = k'$ is true, which means the session key is successfully shared between $S$ and $C$.
Otherwise, the protocol is terminated.

**Table 3**
Key size comparison for the same level of security.

| Symmetric Encryption (key size in bits) | RSA & Diffie–Hellman (module size in bits) | ECC Cryptography (key size in bits) |
|---|---|---|
| 56 | 512 | 112 |
| 80 | 1024 | 160 |
| 112 | 2048 | 224 |
| 128 | 3072 | 256 |
| 192 | 7680 | 384 |
| 256 | 15360 | 512 |

node $C_j$ generates a nonce $N_j$ and encrypts the nonce with key $k_{ij}$ to challenge $C_i$ for authentication.

As can be seen, Protocol 4 supplies the transitory session key negotiation and nonce-based authentication via a *Custodian*. Based on this session key, *Child*) devices can transact their session keys immediately for private communications.

The above four protocols facilitate mutual authentication and key agreement between any two nodes through a mutually trusted *Custodian*. In this process, the VFOG extends cloud-based CA authentication service to network edges. It is significant for delay-sensitive and fast-moving objects, for example, a moving car, to acquire traffic information along the street. In this case, our scheme issues a short-lived session key between the mobile client and a smart traffic system.

**Protocol 4** Inter-*Child* authentication and key management

1: $C_i$: Proposes an encryption key $k_{ij}(i \neq j)$ and a nonce $N_i$ at random (will be used for authentication when the node is requested), initializing the key agreement, and calculates $E_j$,
$$E_j = C_j \times k_{iS}$$
where $C_j$ is the expected communication peer. Then, it encrypts the proposed key with its shared key $K_{iS}$.
$$E_{k_{ij}} = k_{ij} \times k_{iS}$$
2: $C_i \rightarrow S$: Sends $\{E_j, E_{k_{ij}}\}$ to S.
3: $S$: Decrypts the received message from $C_i$ using the shared key $k_{iS}$, computes $C_j$ and $k_{ij}'$.
$$C_j = E_j \times k_{iS}^{-1}$$
$$k_{ij}' = E_{k_{ij}'} \times k_{iS}^{-1}$$
4: $S$: Encrypts the request with shared key $k_{jS}$.
$$E_i = C_i \times k_{jS}$$
$$E_{k_{ij}'} = k_{ij}' \times k_{jS}$$
5: $S \rightarrow C_j$: Forwards $\{E_i, E_{k_{ij}}'\}$ to $C_j$.
6: $C_j$: Decrypts the received message from $S$ using the shared key $k_{jS}$.
$$C_i = E_i \times k_{jS}^{-1},$$
$$k_{ij}'' = E_{k_{ij}'} \times k_{jS}^{-1}$$
7: $C_j$: Generates a nonce $N_j$ and responds to $C_i$'s request. Uses the calculated session key $k_{ij}''$ to encrypt the reply for authentication.
$$E_j = C_j \times k_{ij}''$$
$$E_{N_j} = N_j \times k_{ij}''$$
Meanwhile, $C_j$ memorizes $E_{N_j}$, waiting for authentication.
8: $C_j \rightarrow C_i$: Sends $\{E_j, E_{N_j}\}$ to $C_i$.
9: $C_i$: After receiving the reply, it uses its proposed key at step 1 to decrypt the message. Calculates $C_j$ and $N_j'$.
$$C_j = E_j \times k_{ij} = E_j \times k_{ij}''$$
$$N_j' = E_{N_j} \times k_{ij} = E_{N_j} \times k_{ij}''$$
When it holds, node $C_i$ can now talk to $C_j$ directly for further authentication.
10: $C_i$: Encrypts the nonce with the shared key $K_{ij}$, computes $E_{N_j}'$.
$$E_{N_j}' = N_j' \times k_{ij}$$
11: $C_i \rightarrow C_j$: Sends $\{E_{N_j}'\}$ to $C_j$.
12: $C_j$: Compares the nonce received with stored.
If $E_{N_j}' = E_{N_j}$, the authentication is performed.
Otherwise, the protocol is terminated.

Moreover, we take the *Parent–Child affinity* into consideration for security. With the cognition of working intention and security context, *IVV* contributes to the detection of modifications introduced by physical attacks, as later studied in Section 5. It is of significant importance to generate a symmetric session key for resource-limited devices because it has the smallest key size for the same level of security, as summarized in Table 3.

## 4. The scheme evaluation

As elaborated in Section 2, in the proposed scheme, we decompose computation-intensive CA-related services in a fashion of MSBs, then reassemble them to achieve the same function over multiple FNs. It is thus disseminating micro-services to the proximity of users and devices for authentication along the cloud–fog–device continuum. Since CA services are approaching IoT users and devices, leading to a reduced distance that decreases the registration and authentication delay, meanwhile improves the
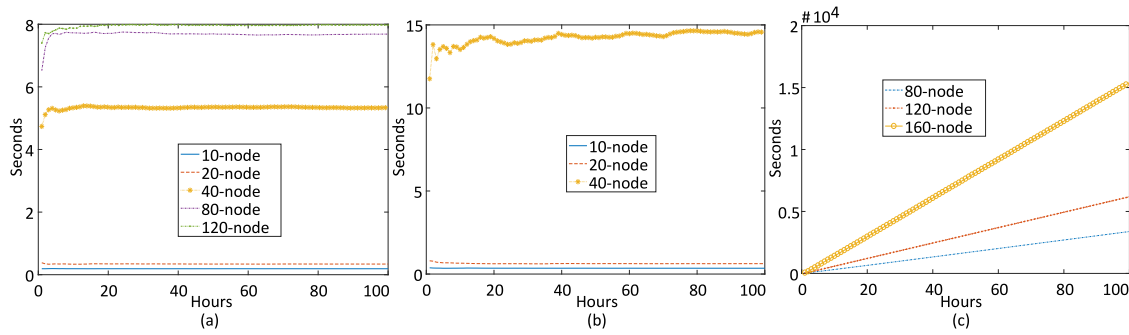
**Fig. 4.** Delay in the cloud-only model: (a) The average registration delay. (b) The authentication delay for a small number of nodes. (c) The authentication delay for a large number of nodes.
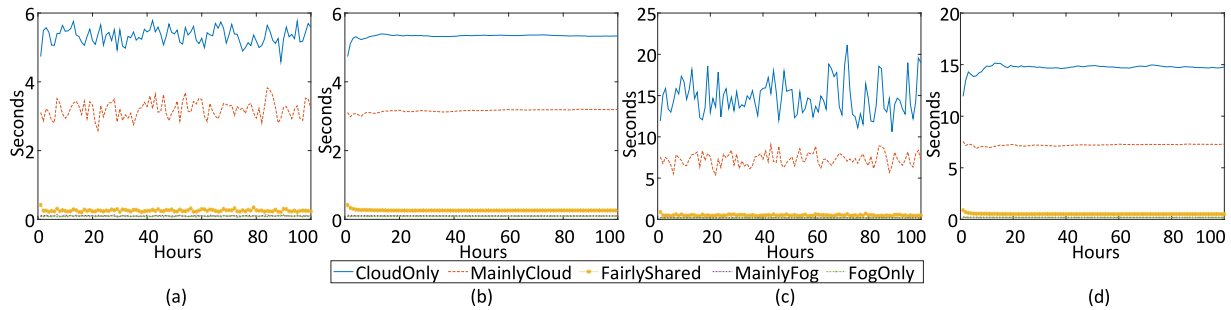


**Fig. 5.** Delay for a fair number of nodes in a shared model: (a) The instantaneous registration delay. (b) The average registration delay. (c) The instantaneous authentication delay. (d) The average authentication delay.

scalability. In this evaluation, we observe the latency depletion along with the cloud–fog–things continuum.

### 4.1. Performance index

In this evaluation, we illustrate the advantage of our scheme, including low-latency of endpoint registration and authentication, as well as high scalability. First, we define the *latency* as the end-to-end delay encompassing nodal processing delay and transmission delay measured in seconds. Second, we interpret the *scalability* as the ability to expand in performing registration and authentication for growing numbers of transactions. The scalability is evaluated based on three parameters, namely, the average delay per transaction, workloads, and CPU utilization on servers. In more detail, the *workload* refers to the number of tasks per second. Accordingly, the more pieces of work, the higher the workloads.

Note that the evaluation of the two indexes are correlative, although their definitions are of fundamental differences. In general, for the same number of endpoints, the lower the average delay is, the higher the scalability it is. Similarly, when a system copes with a specific service, the less number of threads on CPU it is (i.e., fewer workloads), the more scalable it is (implying the potential to handle more tasks), on condition that the processor is not overwhelmed.

It is of the essence to harvest meaningful data in an evaluation process. We, therefore, select mainstream tools with proven effectiveness in network simulation [40]. In this case, the OPNET Modeler and MATLAB are combined in use to collect the required latency and the CPU utilization data due. Particularly, OPNET Modeler is simulating both Cloud and Fog platforms, while the collected data are verified and validated by open-web tools [41] with the aid of MATLAB.

### 4.2. Simulation environment and result

The experimental evaluation conducts in OPNET Modeler 14.5, where several groups of SOs, two routers, the cloud, and two servers make up the testbed. In our testbed, WiFi clients, Wireless routers, and Ethernet servers simulate SOs, routers, and CA servers, respectively. We then reproduce registration with HTTP (heavy browsing) and authentication traffic with DB entry (medium load).

We first observe the average delay of registration and authentication if all traffic is sent and processed in the cloud (traditional cloud model). As outlined in Fig. 4(a), the average registration delay increases from 0.19 s to 7.72 s when the node number varies from 10 to 120 pieces. Likewise, Fig. 4(b) illustrates that the authentication traffic grows 40 times when the entity number jumps from 10 to 40 pieces. This sharp growth is further evidenced by Fig. 4(c), where the authentication delay grows to more than 1000 s linearly. Unfortunately, the high authentication delay will trigger the failure of corresponding applications. To sum up, scalability and availability are poor when all data are sent and processed in the cloud.

Different from the above cloud model that requires all data are sent and processed in a remote cloud, our CAaaS allows us to process the authentication traffic locally, thus decreasing the delay sharply. We define five settings to reflect such transition, i.e., CloudOnly (processed only in the cloud), MainlyCloud (90% processed in the cloud, 10% in the fog), FairlyShared (50% in the cloud and 50% in the fog), MainlyFog (10% in the cloud, others in the fog), and FogOnly (processed only in the fog). As depicted in Figs. 5(a) and 5(b), the registration delay is much reduced from above 5 s (CloudOnly) to under 0.2 s. More interestingly, even fog only processes 10% of the registration traffic, the delay drops by about 42%. Similarly, the CAaaS sharply diminishes authentication delay, as rendered in Figs. 5(c) and 5(d). Indeed, the authentication delay in the FogOnly is about 1.7% of that in the CloudOnly mode.
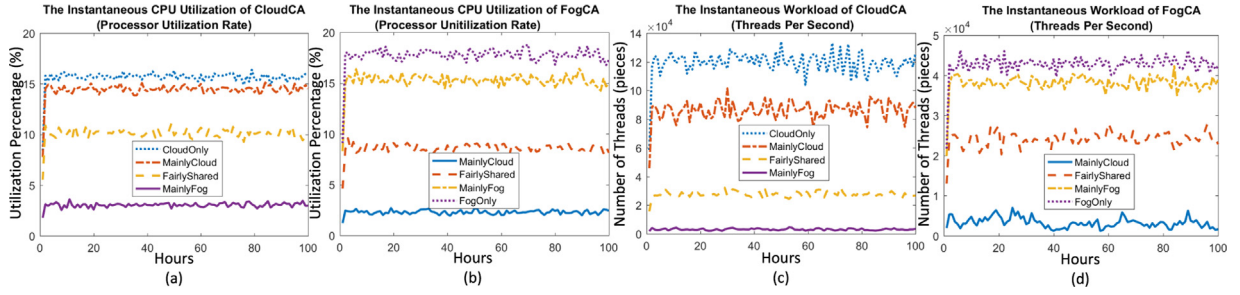
**Fig. 6.** The CPU utilization and workload in the shared model: (a) The instantaneous CPU utilization of CloudCA. (b) The instantaneous CPU utilization of FogCA. (c) The instantaneous workload of CloudCA. (d) The instantaneous workload of FogCA.

Next, we review the CPU utilization and workload on both servers. To this end, Fig. 6(a) details the CPU utilization of the CloudCA while its fog counterpart is described in Fig. 6(b). In general, CPU utilization grows with more pieces of work. Admittedly, the more conducted in the cloud, the fewer remains in the fog, and vice versa. The computing power is, actually, not a bottleneck in either model, as the utilization rate is less than 20% in both servers.

Fig. 6(c) and (d) show the respective workloads on the two servers. Again, the more processed in the cloud, the less is with fog. It is worth noting that the overall tasks on the FogCA are about 45 thousand per second, while on the CloudCA, it is about 140 thousand per second for the same CA services. The difference is due to the excessive data transmission incurred by network congestion. In more detail, when the end-to-end transmission delay is too high, the endpoint has to resend registration and authentication data to remote servers. Such retransmission led to the growth of traffic volume, also causing further delays. Again, the high latency of registration and authentication reduces the scalability and availability of CA services.

In summary, by distributing the required authentication and registration services to a *Custodian* FN, the CAaaS much reduces latency and improves efficiency. Thereby, we showcase the benefits and success of computation movement to the proximity of data sources in the accomplishment of complicated applications.

## 5. The protocol evaluation

In this section, we conduct security analysis and formal verification to confirm that the CAaaS is effective against various attacks.

### 5.1. Security analysis

The security analysis focuses on physical attack, passive attack, replay attack, and impersonation attack.

#### 5.1.1. Physical attack

Since fog cannot provide physical protection (e.g., robbery) for devices, in an insecure zone, we focus on detecting the unregistered modification when an SO joins fog. During the process, fog builds the situation and security awareness of SOs. As scrutinized in [24], fog manages and oversees an SO at its proximity. An SO informs a *Custodian* of its original settings (ID, OS, *IVV*, etc.). Then, the *Custodian* verifies such information by taking advantage of the *Parent–Child affinity*. The SO is trusted when the *IVV* holds.

If a node has any modification after initial registration, the CAaaS can identify the node quickly according to a recalculated *IVV* value. Under this circumstance, it will take further security action (such as device quarantine) to maintain the trustworthiness.

#### 5.1.2. Passive attack

Imagine that an attacker eavesdrops on the session between client and server, trying to derive the secrets in the system. With enough time and endeavors, the attacker may get $\{ID_C, M_C, R_C^*, T_1\}$, and $\{M_S, M_k, T_2\}$, and start to analyze the authentication key $A_{ID_C}$ and the server's private key $K_S^R$. At this instant, it needs to obtain the two values from either $M_C = R_C + t_1 \times A_{ID_C}$ or $M_S = R_S + t_2 \times K_S^R \times Q_{ID_C}$. The attacker would find it too difficult to achieve this because of the difficulties 1) and 3) presented in Section 3.1. For the same reason, the attacker is unable to figure out the session key from $M_k = (k + x_S) \times P$.

#### 5.1.3. Replay attack

Now we suppose that an attacker obtains the data being transferred between client and server, and pretends to be a legitimate client $C$ with $\{M_C, R_C^*\}$. Then, it sends $\{ID_C, M_C, R_C^*, T_1'\}$ to the server, where $T_1'$ is the current time. Since $M_C = R_C + t_1 \times A_{ID_C} = R_C + H_2(T_1) \times A_{ID_C}$, while $T_1' \neq T_1$, the server will find that the verification equation in Step 2 of mutual authentication does not hold, that is, $R_C^* = x_C' \times P$. It is because of $M_C - K_S^R \times t_1' \times Q_{ID_C} \neq M_C - K_S^R \times t_1 \times Q_{ID_C}$, leading to $R_C' \neq R_C$. Vice versa, because of the past timestamp T2 on the client-side, the attacker is unable to claim to be the server by using $\{M_S, M_k\}$.

#### 5.1.4. Impersonation attack

At this moment, we consider that an attacker claims to be a legal client C. It selects a point $R_C'' = (x_C'', y_C'') \in E_p(a, b)$ and $A_{ID_C}''$ to calculate $M_C'' = R_C'' + t_1 \times A_{ID_C}''$ and $R_C^{*''} = x_C'' \times P$. After this, $\{ID_C, M_C'', R_C^{*''}, T_1'\}$ is sent to the server for authentication. Since $M_C''$ is generated from $A_{ID_C}''$, rather than $A_{ID_C}$, the server is unable to get $R_C'' = (x_C'', y_C'')$ from $R_C' = M_C'' - K_S^R \times t_1 \times Q_{ID_C} \neq M_C'' - t_1 \times A_{ID_C}$. Meanwhile, the server finds $R_C'' \neq x_C' \times P$, then determines that the attacker is an illegal user. Likewise, an attacker cannot present as a valid server without knowing the server's private key $K_S^R$.

### 5.2. The formal verification using AVISPA

We use AVISPA to perform the verification, which is a popular evaluation tool for creating and troubleshooting formal models of security protocols [29,42]. In short, we verify the confidentiality of authentication and key agreement message against intruders with a variety of attacking capabilities defined in the threat model. More significantly, AVISPA was developed based on the Dolev–Yao intruder model [43], where the intruder is allowed to intercept, inject, analyze, and change messages in transit at will. On top of this, we assess mutual authentication between clients (*Child*) and servers (*Custodian* and CA). Fig. 7(a) further demonstrates our goal.

During the evaluation, we select the OFMC back-end to perform an automatic diagnosis and generate results. The result confirms that there is no major attack identified in the protocol.

```
Goal

% Confidentiality(Client to Client)
% Clients include IoT nodes and users
Secrecy_of k_ij, k'_ij, k''_ij, k_iS, k_jS

% Mutual authentication (Client to
Client)
% A client i authenticates another client j
based on j's nonce N_j

% Confidentiality(Client to Server)
% A server can be CA, CA proxy, etc.
Secrecy_of ID_C, M_C, R'_C, k, k'

% Mutual authentication (Client to
Server)
% A client and server authenticate each
other on M_S, M_k
```

(a)

```
% OFMC
% Version of 2006/02/13
SUMMARY
    SAFE
DETAILS


BOUNDED_NUMBER_OF_SESSIONS
PROTOCOL
  /home/span/span/testsuite/results/
AuthAndKeyAgr.if
GOAL
    as_specified
BACKEND
    OFMC
COMMENTS
STATISTICS
    parseTime:  0.00s
    searchTime:  12.22s
    visistedNodes:  4600 nodes
    depth:  16 plies
```

(b)

**Fig. 7.** The AVISPA evaluation. (a) The specific goal. (b) The result.

Our protocols are, therefore, secure against the active and passive attacks mentioned above. Fig. 7(b) displays the detailed result.

To summarize, our proposed scheme is efficient (low-latency and scalable) and effective (against various attacks) for authenticating resource-constrained devices in the IoT.

## 6. Related work

Authentication is a cornerstone in the construction of a security pillar [14], identifying the legitimacy of users and devices. To explain, Ali et al. conducted a thorough study of security issues in respective layers of IoT architecture, claiming that a strong device authentication is critical in the perception layer [44]. Likely, Zhou et al. stressed various attacks on pervasive user devices [45]. Reasonably, both academia and industry made impressive efforts in designing and implementing effective schemes for authenticating IoT devices. Silva et al. surveyed main challenges for authentication in the IoT [46], including low computation overhead, protocol deployment on resource-limited nodes, key management, and device heterogeneity. Likewise, Yang et al. in [47] reviewed relevant limitations, possible attacks, and solutions in IoT authentication, reiterating the necessity of a lightweight protocol for overcoming a diversity of resource constraints.

The research community recently proposed a myriad of cutting-edge concepts for IoT authentication, e.g., machine learning (ML), AI, big data [10], 5G [20], SDN, and blockchain. It is worth mentioning that neither ML, AI, big data, nor 5G is an independent technique in providing authentication, although these technologies have witnessed tremendous success in many areas. For example, Kaur et al. applied big data in optimizing energy utilization for the IoT in [30]. When these technologies are appropriately in use, they bring critical benefits in securing the IoT. To enumerate, Neshenko et al. investigated various security issues and current solutions in their comprehensive survey paper [6], pinpointing the future initiative for the combination of technological advancements in addressing various security concerns raised in the IoT ecosystem. Garg et al. developed a deep learning-based model for anomaly detection in [31], improving accuracy and detection rate in securing cloud servers. Next, we briefly explore SDN and blockchain, respectively.

Shu et al. investigated diverse threats and countermeasures for SDN in [13], arguing that SDN is capable of empowering security against various attacks. Nevertheless, the authors inadequately presented a related authentication scheme. Additionally, Salman et al. introduced SDN in their identity-based authentication framework, claiming their approach takes control of various attacks such as replay attack and man-in-the-middle (MITM) attack [48]. Although this may be true, it was merely a conceptual framework without realization protocols. Furthermore, the authors did not study a things-to-things authentication approach in their work. In contrast, we provide a concrete solution for IoT authentications.

Many scholars proposed the blockchain technology in enabling the security for IoT as surveyed in [17]. Almadhoun et al. further proposed a user authentication scheme for fog-based IoT in [8] where the blockchain technique was employed to protect fog nodes, via which, authenticating users to access IoT devices. Similarly, in [49], Li et al. came up with a blockchain-based mechanism to authenticate IoT devices, where a requested node searched the public key of the requesting node in the local blockchain or through consensus nodes recursively. Their scheme offered benefits such as resisting DDoS attacks and preventing malicious accesses. However, the performance would be heavily relying on the blockchain platform. In more detail, it is challenging to design and implement a contiguous blockchain platform in a highly dynamic environment and meanwhile reducing the search time.

Benefiting from a combination of passwords, smart cards, and biometrics, the multi-factor authentication (MFA) approach [50] is becoming popular for mobile IoT devices [51]. Compared with a single-factor system, it provides an additional tier of security in case one factor becomes compromised. Other benefits include ease of use, better compliance with related regulation, and improved user experience. Because such approaches usually need manual intervention, it is infeasible to numerous clientless things.

Situational awareness is the coordinated perception and comprehension of environmental elements in a network, which has played an important role in cybersecurity. In [3], Rapuzzi et al. proposed a three-layer architecture to establish the cognition for security threats, leveraging the technological combination of big data, SDN, NFV, fog, and cloud computing. Xiang et al. developed a situation-aware authentication scheme for smart meters in [52], adjusting security protocol based on assessed security risk levels. When security risk is low, it conducts the low-cost protocol for optimized efficiency. Unfortunately, it is an open issue to assess security levels.

On the other hand, Ammar et al. surveyed the security of mainstream IoT frameworks in the market provided by industrial giants, including Google, Amazon, Apple, and Microsoft. It seems that the industry still prefers CA-based solutions in the IoT authentication [53]. In summary, all industrial leaders scrutinized in the paper employed a CA-based PKI mechanism, which motivated us to focus on the CA-based authentication scheme in our study. Consequently, our solution may be more practical as it is in alignment with the mainstream authentication provisioning in today's market.

Note that a lightweight protocol is effective in addressing the earlier mentioned concerns. Garg et al. put forward a lightweight authentication protocol for IoT in [54], taking advantage of physically unclonable functions and XOR. In [7], Salami et al. designed a lightweight encryption scheme for the smart home, confirmed its efficiency in protecting smart environments. Due to its lightweight feature, ECC, to a large extent, has been proven its feasibility in IoT and sensor domain [55]. He et al. surveyed over thirty ECC-based authentication schemes satisfying the lightweight feature, while pointing out such schemes were insufficient in defending against replay attack, impersonation attack, and MITM attack in [56]. Lee et al. designed a new authentication protocol, named ECDLP-based randomized access control [57], in which the authors minimized the computational workload and the size of required memory. It was, however, incapable of handling impersonation attacks, as identified in [58]. In comparison, our protocol tackles such concerns.

It is as well beneficial to expedite security processing for enhanced security by employing a combination of software and hardware advancements. For example, Kotamsetty et al. proposed an adaptive latency-aware query processing approach over encrypted data in [59]. In this work, the authors showed the latency drop by fine-tuning data size but did not evaluate security in their paper. In contrast, Liu et al. proposed a hardware-accelerated ECC endomorphism, reducing delay up to 50% of its original value as tested on the specially designed hardware platform in [60]. However, products in the current market cannot take advantage of it.

Different from the above methods in authentication acceleration, our scheme takes advantage of virtual fog through which an authentication service is placed closer to users and devices, thus reducing the latency. To this end, computing intelligence moves closer to data instead of data going to remote servers in the cloud. Moreover, benefiting from *Parent–Child affinity*, we minimize the attack surface on endpoints. As a result, our scheme is effective and efficient for IoT nodes, without much modification requirements of hardware and software. Meanwhile, we embrace the cloud–fog–things continuum in the CAaaS, which makes it more suitable for the IoT ecosystem. Last but not least, our solution protects previous investments in a progressive process in the construction of smart environments.

## 7. Discussion

Security is one of the most important characters that an established network must offer. Due to its inborn constraints of computing intelligence, many devices have insufficient power to safeguard themselves [6]. It is fundamental to save incapable nodes with in-network protection intelligence, as a security level lies with an endpoint of the least security. To this end, our CAaaS steps in, just in time, by providing an effective and efficient authentication solution.

Powered by MSB, VNF, VO, and SDN, the VFOG is the foundation of our CAaaS scheme, through which, CA services step closer to data sources, resulting in low latency and high scalability. The VFOG framework brings local situational awareness along the cloud–fog–things continuum, contributing to the improved efficiency and effectiveness for authentication. It is worth noting that the VFOG is open to recent technological advancements. The collaboration of modern technologies will facilitate strengthening situational awareness for *child*, *parent*, and *custodian*. However, it remains an open issue to concretely build situational security-awareness for resource-constrained IoT nodes with the current technique.

In this paper, we evaluate the authentication scheme using OPNET Modeler. In detail, we presented the end-to-end packet delay and workload on the server-side. In future evaluations, it is possible to contain more parameters such as the number of packets sent and received, and packet size information. Such data may aid in studying the simulation accuracy in comparison with data in an established work. As a reference for other scholars, it is also valuable to compare network diagrams and simulation methodology in future work.

The protocol to validate the CAaaS scheme is grounded in ECC. However, there are well-known security issues for the implementation of ECC, including side-channel attacks, possible NSA backdoors, and invalid curve attacks. Fan et al. concluded the leakage of the private key in several cases [61]. It is, therefore, compulsory to perform a thorough security code review, static code evaluation, and stringent penetration testing to prevent improper implementation.

## 8. Conclusion

To conclude, we demonstrate the benefits of computing intelligence movements towards data sources in handling complicated services conducted traditionally in a cloud. In the proposed scheme, a variety of CA services encompassing key generation, storage, distribution, CRL, and CA proxies, separately performed on FN and reassembled to achieve required CA functions for authentication. Compared with a centralized cloud model requiring a large volume of data sent to and processed on a remote server, our scheme leverages fog computing and microservices, extending the intelligence of security computation to data sources with situational awareness. Evaluation results demonstrate the benefits of our CAaaS, including security, low latency, and high scalability. More interestingly, in the implementation of our protocols, computation-intensive workloads such as key generation are performed in fog, while resource-constrained devices only need to perform low-cost computation. We investigate the *Parent–Child affinity* as a valuable input towards end-to-end security. Meanwhile, protocol evaluations confirm the effectiveness of anti-popular attacks. Our scheme is more suitable to resource-limited entities in smart living and other similar environments, grounded in benefits such as efficiency and effectiveness without computation-intensive workload requirements on the client-side.

## CRediT authorship contribution statement

**Jianhua Li:** Conception and design of study, Acquisition of data, Analysis and/or interpretation of data, Writing - original draft, Writing - review & editing. **Jiong Jin:** Conception and design of study, Analysis and/or interpretation of data, Writing - original draft, Writing - review & editing. **Lingjuan Lyu:** Conception and design of study, Analysis and/or interpretation of data, Writing - original draft, Writing - review & editing. **Dong Yuan:** Conception and design of study, Writing - original draft, Writing - review & editing. **Yingying Yang:** Conception and design of study, Acquisition of data, Writing - original draft. **Longxiang Gao:** analysis and/or interpretation of data, Writing - review & editing. **Chao Shen:** Writing - review & editing.

## Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Acknowledgments

## References

[1] K. Noda, Google home: smart speaker as environmental control unit, Disabil. Rehabil. Assist. Technol. 13 (7) (2017) 674–675, http://dx.doi.org/10.1080/17483107.2017.1369589.

[2] J. Li, J. Jin, D. Yuan, M. Palaniswami, K. Moessner, EHOPES: Data-centered Fog platform for smart living, in: Proceedings of the International Telecommunication Networks and Applications Conference, ITNAC 2015, Sydney, Australia, 2015, pp. 308–313, http://dx.doi.org/10.1109/ATNAC.2015.7366831.

[3] R. Rapuzzi, M. Repetto, Building situational awareness for network threats in fog/edge computing: Emerging paradigms beyond the security perimeter model, Future Gener. Comput. Syst. 85 (2018) 235–249, http://dx.doi.org/10.1016/j.future.2018.04.007.

[4] J. Klobas, T. McGill, X. Wang, How perceived security risk affects intention to use smart home devices: A reasoned action explanation, Comput. Secur. 87 (2019) 101571–101583, http://dx.doi.org/10.1016/j.cose.2019.101571.

[5] J. Liranzo, T. Hayajneh, ***security and privacy issues affecting cloud-based IP camera, in: 2017 IEEE 8th Annual Ubiquitous Computing, Electronics and Mobile Communication Conference (UEMCON), IEEE, New York, NY, USA, 2017, pp. 458–465, http://dx.doi.org/10.1109/UEMCON.2017.8249043.

[6] N. Neshenko, E. Bou-Harb, J. Crichigno, G. Kaddoum, N. Ghani, Demystifying IoT security: An exhaustive survey on IoT vulnerabilities and a first empirical look on internet-scale IoT exploitations, IEEE Commun. Surv. Tutor. 21 (3) (2019) 2702–2733, http://dx.doi.org/10.1109/COMST.2019.2910750.

[7] S.A. Salami, J. Baek, K. Salah, E. Damiani, Lightweight encryption for smart home, in: Proceedings of International Conference on Availability, Reliability and Security, ARES 2016, Salzburg, Austria, 2016, pp. 382–388, http://dx.doi.org/10.1109/ARES.2016.40.

[8] R. Almadhoun, M. Kadadha, M. Alhemeiri, M. Alshehhi, K. Salah, A user authentication scheme of IoT devices using blockchain-enabled fog nodes, in: Proceedings of International Conference on Computer Systems and Applications, AICCSA 2018, Aqaba, Jordan, 2018, pp. 1–8, http://dx.doi.org/10.1109/AICCSA.2018.8612856.

[9] Z. Yan, P. Zhang, A. Vasilakos, A security and trust framework for virtualized networks and software-defined networking, Secur. Commun. Netw. 9 (16) (2016) 3059–3069, http://dx.doi.org/10.1002/sec.1243.

[10] E. Ahmed, I. Yaqoob, I. Hashem, I. Khan, A. Ahmed, M. Imran, A. Vasilakos, The role of big data analytics in internet of things, Comput. Netw. 129 (2017) 459–471, http://dx.doi.org/10.1016/j.comnet.2017.06.013.

[11] M. Wazid, A. Das, V. Bha, A. Vasilakos, LAM-CIoT: Lightweight authentication mechanism in cloud-based IoT environment, J. Netw. Comput. Appl. 150 (2020) 102496, http://dx.doi.org/10.1016/j.jnca.2019.102496.

[12] H. Khattak, M. Shah, S. Khan, I. Ali, M. Imran, Perception layer security in internet of things, Future Gener. Comput. Syst. 100 (2019) 144–164, http://dx.doi.org/10.1016/j.future.2019.04.038.

[13] Z. Shu, J. Wan, D. Li, J. Lin, A. Vasilakos, M. Imran, Security in software-defined networking: Threats and countermeasures, Mob. Netw. Appl. 21 (5) (2016) 764–776, http://dx.doi.org/10.1007/s11036-016-0676-x.

[14] Q. Jing, A. Vasilakos, J. Wan, J. Lu, D. Qiu, Security of the internet of things: perspectives and challenges, Wirel. Netw. 20 (8) (2014) 2481–2501, http://dx.doi.org/10.1007/s11276-014-0761-7.

[15] L. Lyu, K. Nandakumar, B. Rubinstein, J. Jin, J. Bedo, M. Palaniswami, PPFA: Privacy preserving fog-enabled aggregation in smart grid, IEEE Trans. Ind. Inf. 14 (8) (2018) 3733–3744, http://dx.doi.org/10.1109/TII.2018.2803782.

[16] Z. Sheng, S. Yang, Y. Yu, A.V. Vasilakos, J.A. Mccann, K.K. Leung, A survey on the ietf protocol suite for the internet of things: standards, challenges, and opportunities, IEEE Wirel. Commun. 20 (6) (2013) 91–98, http://dx.doi.org/10.1109/MWC.2013.6704479.

[17] M. Khan, K. Salah, IoT Security: Review, blockchain solutions, and open challenges, Future Gener. Comput. Syst. 82 (2018) 395–411, http://dx.doi.org/10.1016/j.future.2017.11.022.

[18] M. Anderson, Looking for the key to security in the internet of things, IEEE Spectr. (2014) Available https://spectrum.ieee.org/riskfactor/consumer-electronics/standards/looking-for-the-key-to-security-in-the-internet-of-things (Accessed 26 November 2019).

[19] S. El Kafhali, K. Salah, Efficient and dynamic scaling of fog nodes for IoT devices, J. Supercomput. 73 (12) (2017) 5261–5284, http://dx.doi.org/10.1007/s11227-017-2083-x.

[20] M. Huang, A. Liu, N. Xiong, T. Wang, A. Vasilakos, An effective service-oriented networking management architecture for 5g-enabled internet of things, Comput. Netw. 173 (2020) 1–17, http://dx.doi.org/10.1016/j.comnet.2020.107208.

[21] L. Lyu, S. Chau, N. Wang, Y. Zheng, Cloud-based privacy-preserving collaborative consumption for sharing economy, IEEE Trans. Cloud Comput. (2020) http://dx.doi.org/10.1109/TCC.2020.3010235.

[22] J. Zhou, Z. Cao, X. Dong, A. Vasilakos, Security and privacy for cloud-based IoT: Challenges, IEEE Commun. Mag. 55 (1) (2017) 26–33, http://dx.doi.org/10.1109/MCOM.2017.1600363CM.

[23] E. Fadda, P. Plebani, M. Vitali, Monitoring-aware optimal deployment for applications based on microservices, IEEE Trans. Serv. Comput. (2019) 1–14, http://dx.doi.org/10.1109/TSC.2019.2910069.

[24] J. Li, J. Jin, D. Yuan, H. Zhang, Virtual fog: A virtualization enabled fog computing framework for internet of things, IEEE Internet Things J. 5 (1) (2018) 121–131, http://dx.doi.org/10.1109/JIOT.2017.2774286.

[25] G. Sun, Z. Xu, H. Yu, X. Chen, V. Chang, A.V. Vasilakos, Low-latency and resource-efficient service function chaining orchestration in network function virtualization, IEEE Internet Things J. (2019) 1–13, http://dx.doi.org/10.1109/JIOT.2019.2937110.

[26] G. Sun, R. Zhou, J. Sun, H. Yu, A.V. Vasilakos, Energy-efficient provisioning for service function chains to support delay-sensitive applications in network function virtualization, IEEE Internet Things J. 7 (7) (2020) 6116–6131, http://dx.doi.org/10.1109/JIOT.2020.2970995.

[27] J. Thönes, Microservices, IEEE Softw. 32 (1) (2015) 116, http://dx.doi.org/10.1109/MS.2015.11.

[28] Eplexity, Why the public cloud is more secure than an on-premises data center. Available https://eplexity.com/why-the-public-cloud-is-more-secure-than-an-on-premises-data-center/, (Accessed 26 November 2019).

[29] S. Challa, A.K. Das, P. Gope, N. Kumar, F. Wu, A.V. Vasilakos, Design and analysis of authenticated key agreement scheme in cloud-assisted cyber–physical systems, Future Gener. Comput. Syst. 108 (2020) 1267–1286, http://dx.doi.org/10.1016/j.future.2018.04.019.

[30] K. Kaur, S. Garg, G. Kaddoum, E. Bou-Harb, K.R. Choo, A big data-enabled consolidated framework for energy efficient software defined data centers in IoT setups, IEEE Trans. Ind. Inf. 16 (4) (2020) 2687–2697, http://dx.doi.org/10.1109/TII.2019.2939573.

[31] S. Garg, K. Kaur, N. Kumar, G. Kaddoum, A.Y. Zomaya, R. Ranjan, A hybrid deep learning-based model for anomaly detection in cloud datacenter networks, IEEE Trans. Netw. Serv. Manag. 16 (3) (2019) 924–935, http://dx.doi.org/10.1109/TNSM.2019.2927886.

[32] L. Lyu, J. Bezdek, X. He, J. Jin, Fog-embedded deep learning for the internet of things, IEEE Trans. Ind. Inf. 15 (7) (2019) 4206–4215, http://dx.doi.org/10.1109/TII.2019.2912465.

[33] L. Lyu, J. Bezdek, J. Jin, Y. Yang, FORESEEN: Towards differentially private deep inference for intelligent internet of things, IEEE J. Sel. Areas Commun. (2020) http://dx.doi.org/10.1109/JSAC.2020.3000374.

[34] Q. Zheng, J. Jin, T. Zhang, J. Li, L. Gao, Y. Xiang, Energy-sustainable fog system for mobile web services in infrastructure-less environments, IEEE Access 7 (2019) 161318–161328, http://dx.doi.org/10.1109/ACCESS.2019.2951162.

[35] P. Tonella, Using a concept lattice of decomposition slices for program understanding and impact analysis, IEEE Trans. Softw. Eng. 29 (6) (2003) 495–509, http://dx.doi.org/10.1109/TSE.2003.1205178.

[36] C.D. Newman, T. Sage, M.L. Collard, H.W. Alomari, J.I. Maletic, srcSlice: A tool for efficient static forward slicing, in: 2016 IEEE/ACM 38th International Conference on Software Engineering Companion, ICSE-C, Austin, TX, USA, 2016, pp. 621–624.

[37] F. Chiti, R. Fantacci, F. Paganelli, B. Picano, Virtual functions placement with time constraints in fog computing: A matching theory perspective, IEEE Trans. Netw. Serv. Manag. 16 (3) (2019) 980–989, http://dx.doi.org/10.1109/TNSM.2019.2918637.

[38] W. Tärneberg, A. Mehta, E. Wadbro, J. Tordsson, J. Eker, M. Kihl, E. Elmroth, Dynamic application placement in the mobile cloud network, Future Gener. Comput. Syst. 70 (2017) 163–177, http://dx.doi.org/10.1016/j.future.2016.06.021.

[39] J. Li, T. Zhang, J. Jin, Y. Yang, D. Yuan, L. Gao, Latency estimation for fog-based internet of things, in: Proceedings of the International Telecommunication Networks and Applications Conference, ITNAC 2017, Melbourne, Australia, 2017, pp. 1–6, http://dx.doi.org/10.1109/ATNAC.2017.8215403.

[40] K. Salah, On the deployment of voip in ethernet networks: methodology and case study, Comput. Commun. 29 (8) (2006) 1039–1054, http://dx.doi.org/10.1016/j.comcom.2005.06.004.

[41] M. Leonhard, CloudPing.info. Available https://www.cloudping.info. (Accessed 26 March 2020).

[42] M. Wazid, A. Das, N. Kumar, A. Vasilakos, Design of secure key management and user authentication scheme for fog computing services, Future Gener. Comput. Syst. 91 (2019) 475–492, http://dx.doi.org/10.1016/j.future.2018.09.017.

[43] I. Cervesato, The Dolev-Yao intruder is the most powerful attacker, in: Proceedings of the Annual Symposium on Logic in Computer Science, LICS 2001, vol. 1, Boston, Massachusetts, USA, 2001, pp. 1–2.

[44] I. Ali, S. Sabir, Z. Ullah, Internet of things security, device authentication and access control: a review, 2019, arXiv preprint arXiv:1901.07309.

[45] J. Zhou, X. Dong, Z. Cao, A.V. Vasilakos, Secure and privacy preserving protocol for cloud-based vehicular DTNs, IEEE Trans. Inf. Forensics Secur. 10 (6) (2015) 1299–1314, http://dx.doi.org/10.1109/TIFS.2015.2407326.

[46] E. Silva, W. Lima, F. Ferraz, f. Nascimento Ribeiro, Authentication and the internet of things: A survey based on a systematic mapping, in: Proceedings of the International Conference on Software Engineering Advances, ICSEA 2017, vol. 4, Athens, Greece, 2017, pp. 34–40.

[47] Y. Yang, L. Wu, G. Yin, L. Li, H. Zhao, A survey on security and privacy issues in internet-of-things, IEEE Internet Things J. 4 (5) (2017) 1250–1258, http://dx.doi.org/10.1109/JIOT.2017.2694844.

[48] O. Salman, S. Abdallah, I.H. Elhajj, A. Chehab, A. Kayssi, Identity-based authentication scheme for the Internet of Things, in: Proceedings of the IEEE Symposium on Computers and Communication, ISCC 2016, Messina, Italy, 2016, pp. 1109–1111, http://dx.doi.org/10.1109/ISCC.2016.7543884.

[49] D. Li, W. Peng, W. Deng, F. Gai, A blockchain-based authentication and security mechanism for IoT, in: Proceedings of the International Conference on Computer Communication and Networks, ICCCN 2018, Hangzhou, China, 2018, pp. 1–6, http://dx.doi.org/10.1109/ICCCN.2018.8487449.

[50] A. Das, S. Zeadally, D. He, Taxonomy and analysis of security protocols for internet of things, Future Gener. Comput. Syst. 89 (2018) 110–125, http://dx.doi.org/10.1016/j.future.2018.06.027.

[51] M. Ferrag, L. Maglaras, A. Derhab, Authentication and authorization for mobile IoT devices using biofeatures: Recent advances and future trends, Secur. Commun. Netw. 2019 (2019) 1–20, http://dx.doi.org/10.1155/2019/5452870.

[52] A. Xiang, J. Zheng, A situation-aware scheme for efficient device authentication in smart grid-enabled home area networks, Electronics 9 (6) (2020) 989–1005, http://dx.doi.org/10.3390/electronics9060989.

[53] A. Mahmoud, R. Giovanni, C. Bruno, Internet of things: A survey on the security of IoT frameworks, J. Inf. Secur. Appl. 38 (2018) 8–27, http://dx.doi.org/10.1016/j.jisa.2017.11.002.

[54] S. Garg, K. Kaur, G. Kaddoum, K.R. Choo, Toward secure and provable authentication for Internet of Things: Realizing industry 4.0, IEEE Internet Things J. 7 (5) (2020) 4598–4606, http://dx.doi.org/10.1109/JIOT.2019.2942271.

[55] X. Li, J. Niu, M.Z.A. Bhuiyan, F. Wu, M. Karuppiah, S. Kumari, A robust ECC-based provable secure authentication protocol with privacy preserving for industrial internet of things, IEEE Trans. Ind. Inf. 14 (8) (2018) 3599–3609, http://dx.doi.org/10.1109/TII.2017.2773666.

[56] D. He, S. Zeadally, An analysis of RFID authentication schemes for internet of things in healthcare environment using elliptic curve cryptography, IEEE Internet Things J. 2 (1) (2015) 72–83, http://dx.doi.org/10.1109/JIOT.2014.2360121.

[57] Y.K. Lee, L. Batina, I. Verbauwhede, EC-RAC (ECDLP Based Randomized Access Control): Provably Secure RFID authentication protocol, in: Proceedings of IEEE International Conference on RFID, Las Vegas, NV, USA, 2008, pp. 97–104, http://dx.doi.org/10.1109/RFID.2008.4519370.

[58] J. Bringer, H. Chabanne, T. Icart, Cryptanalysis of EC-RAC, a RFID identification protocol, in: Proceedings of the International Conference on Cryptology and Network Security (CANS 2008), Springer, Hong-Kong, China, 2008, pp. 149–161, http://dx.doi.org/10.1007/978-3-540-89641-8_11.

[59] R. Kotamsetty, M. Govindarasu, Adaptive latency-aware query processing on encrypted data for the internet of things, in: Proceedings of the International Conference on Computer Communication and Networks, ICCCN 2016, Waikoloa, Hawaii, USA, 2016, pp. 1–7, http://dx.doi.org/10.1109/ICCCN.2016.7568488.

[60] Z. Liu, J. Großschädl, Z. Hu, K. Järvinen, H. Wang, I. Verbauwhede, Elliptic curve cryptography with efficiently computable endomorphisms and its hardware implementations for the internet of things, IEEE Trans. Comput. 66 (5) (2017) 773–785, http://dx.doi.org/10.1109/TC.2016.2623609.

[61] J. Fan, X. Guo, E. De Mulder, P. Schaumont, B. Preneel, I. Verbauwhede, State-of-the-art of secure ECC implementations: a survey on known side-channel attacks and countermeasures, in: IEEE International Symposium on Hardware-Oriented Security and Trust, HOST 2010, Anaheim, CA, USA, 2010, pp. 76–87, http://dx.doi.org/10.1109/HST.2010.5513110.

**Jianhua Li** received the Ph.D. degree in Information Engineering from Swinburne University of Technology, Melbourne, VIC, Australia, in 2019. He is currently a Lecturer with the School of Information Technology, Deakin University, Melbourne, VIC, Australia. Meanwhile, he is an ICT professional with extensive hands-on experience on multi-million dollar projects. His current research interests include fog/edge computing, the Internet of Things, cyber-physical systems, quality of experience, artificial intelligence, cybersecurity, and networking softwarization.
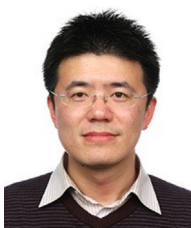
**Jiong Jin** (M'11) received the B.E. degree with First Class Honours in Computer Engineering from Nanyang Technological University, Singapore, in 2006, and the Ph.D. degree in Electrical and Electronic Engineering from the University of Melbourne, Australia, in 2011. From 2011 to 2013, he was a Research Fellow in the Department of Electrical and Electronic Engineering at the University of Melbourne. He is currently a Senior Lecturer in the School of Software and Electrical Engineering, Faculty of Science, Engineering and Technology, Swinburne University of Technology, Melbourne, Australia. His research interests include network design and optimization, edge computing and distributed systems, robotics and automation, and cyber-physical systems and Internet of Things as well as their applications in smart manufacturing, smart transportation and smart cities.

**Lingjuan Lyu** (M'18) is currently a Research Fellow with The Department of Computer Science, National University of Singapore. She received Ph.D. degree from the University of Melbourne. Her current research interests span machine learning, privacy, fairness, and edge intelligence. She has publications in prestigious venues, including: JSAC, TPDS, TDSC, TCC, JIOT, TII, DKE, SIGIR, WWW, CIKM, FL-IJCAI, IJCNN, TrustCom, PerCom, etc. Her work was supported by an IBM Ph.D. Fellowship (50 winners Worldwide) and contributed to various professional activities, including AISTATS, ICML, NIPS, NIPS workshop, S&P, PETS, JIOT, TCC, TII, Neurocomputing, ACM Computing Surveys, TKDE, TDSC, TNSM, IEEE Access etc.

**Dong Yuan** received the B.E. and M.E. degrees from Shandong University, Jinan, China, in 2005 and 2008, respectively, and the Ph.D. degree from the Swinburne University of Technology, Melbourne, VIC, Australia, in 2012. He is a Lecturer with the School of Electrical and Information Engineering, University of Sydney, Sydney, NSW, Australia. His current research interests include cloud computing, data management in parallel and distributed systems, scheduling and resource management, Internet of Things, business process management, and workflow systems.

**Yingying Yang** received the B.E in Harbin Institute of Technology, Harbin, China in 2002 and M.E degrees from Charles Sturt University, Sydney, NSW, Australia in 2013. She is a current Ph.D. student in University Technology Sydney, NSW, Australia. She is a distinguished CISCO Networking Academy Instructor trainer with more than twelve years' experience. Her current research interests include cloud computing, fog/edge computing, physical-cyber systems, Internet of Things, and software-defined networking.

**Longxiang Gao** received his Ph.D. in Computer Science from Deakin University, Australia. He is currently a Senior Lecturer at School of Information Technology, Deakin University. His research interests include fog/edge computing, blockchain, data analysis and privacy protection. Dr. Gao has over 80 publications, including patent, monograph, book chapter, journal and conference papers. Some of his publications have been published in the top venue, such as IEEE TMC, IEEE IoTJ, IEEE TDSC, IEEE TVT, IEEE TCSS, and IEEE TNSE. He is a Senior Member of IEEE.

**Chao Shen** [S'09, M'14, SM'o20] is a full Professor in the faculty of Electronic and Information Engineering, Xi'an Jiaotong University of China. He was a research scholar in Computer Science Department at Carnegie Mellon University from 2011 to 2013. He has published more than 70 research papers in international referred journals and conferences. His current research interests mainly include Data-Driven Network and System Security, AI Security, Cyber–Physical System Security. He currently serves as an Associate Editor for a number of journals, including IEEE Transactions on Dependable Secure Computing, Journal of Franklin Institute, and Frontiers of Computer Science, etc.